VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Steganografie v IP telefonii
Steganography in IP telephony

2015 Bc. Ivo Zbranek

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra telekomunikační techniky

Zadání diplomové práce

Student: **Bc. Ivo Zbranek**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T059 Mobilní technologie

Téma: Steganografie v IP telefonii
Steganography in IP telephony

Zásady pro vypracování:

Dnešní IDS systémy jsou schopny odhalit ukrytý kanál jako anomálii např. pomocí Holt-Winters modelu a především jeho rozšíření v Brutlag algoritmu. Výsledky výzkumu, které v oblasti steganografie v IP telefonii publikoval tým prof. Szczypiorského z Varšavské univerzity, ukazují způsob, jak vytvořit utajený kanál v SIP signalizaci. Cílem diplomové práce je rozvést myšlenky prof. Szczypiorského v oblasti využití specifických polí SIP hlavičky a zaměřit se i na využití vložení utajených dat do RTP toku.

1. Úvod do VoIP a steganografie.
2. Metody pro detekci anomálií.
3. Realizace výměny informací pomocí vloženého pole v SIP signalizaci.
4. Využití RTP pro přenos textových informací.
5. Zhodnocení dosažených výsledků.

Seznam doporučené odborné literatury:

W. Mazurczyk, K. Szczypiorski.Covert Channels in SIP for VoIP signaling, In *Proc. of 4th International Conference on Global E-security 2008* . London, United Kingdom, 23-25 June 2008, pp. 65-72.
M. Szmit, A. Szmit. Usage of Modified Holt-Winters Method in the Anomaly Detection of Network Traffic: Case Studies. In *Journal of Computer Networks and Communications*. Volume 2012 (2012), Article ID 192913, 5 pages. http://dx.doi.org/10.1155/2012/192913.
V. Berk, A. Giani, G. Cybenko. *Detection of covert channel encoding in network packet delays.* Department of Computer Science, Dartmouth College, Tech. Rep. TR2005-536, November 2005.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **doc. Ing. Miroslav Vozňák, Ph.D.**

Datum zadání: 01.09.2014

Datum odevzdání: 07.05.2015

doc. Ing. Miroslav Vozňák, Ph.D.
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 12.dubna 2015 .....

Tímto bych rád poděkoval doc. Ing. Miroslavu Vozňákovi, Ph.D. za předání jeho cenných zkušeností, vědomostí a v neposlední řadě také za vstřícnost a trpělivost.

Abstrakt

Tato diplomová práce se zabývá steganografickými technikami a jejich aplikací v oblasti technologie IP telefonie. Obsahuje popis jednotlivých steganografických metod a technologií, které se používají v rámci IP telefonie. V další části jsou popsány a rozebrány metody pro detekci anomálií, na základě algoritmu Holt-Winters, Holt-Winters:Brutlag a Naive Bayes. Současně se zabývá umístěním steganogramu do hlavičky SIP protokolu, schopnosti jej detekovat pomocí systému pro detekci anomálií a využitím aplikace SIPp pro generování injektovaných zpráv SIP protokolu. Další část práce popisuje steganografickou metodu využívající datový tok RTP protokolu, pro přenos textových informací. Následně jsou zhodnoceny dosažené výsledky.

Klíčová slova: steganografie, IP telefonie, VoIP, bezpečnost, SIP, RTP, SIPp, SNORT AD, anomálie, detekce

Abstract

This master thesis concerns with steganographic techniques and their application in the field of IP telephony. It encompasses description of individual steganographic methods and technologies which are used within the scope of IP telephony. The next part analyzes and describes anomaly detection methods based on Holt-Winters, Holt-Winters:Brutlag and Naive Bayes. Simultaneously, concerns with embedding of a steganogram into an SIP protocol header, means of its detection with help of the anomaly detection system and the use of SIPp application for generating injected SIP protocol messages. The next part of thesis describes a steganographic method which uses an RTP protocol data flow for the transfer of text-based informations. The achieved results are evaluated afterwards.

Keywords: steganography, IP telephony, VoIP, security, SIP, RTP, SIPp, SNORT AD, anomaly, detection

Seznam použitých zkratek a symbolů

3GPP	–	3rd Generation Partner- ship Project
ACELP	–	Algebraic Code Excited Linear Prediction
AD	–	Anomaly Detection
ADPCM	–	Adaptive Differential Pulse Code Modulation
ADS	–	Anomaly Detection System
AKA	–	Authentication and Key Agreement
AMPS	–	Advanced Mobile Phone System
APP	–	Application-Specific Message
ARP	–	Address Resolution Protocol
ASCII	–	American Standard Code for Information Interchange
ATM	–	Asynchronous Transfer Mode
AVG	–	Moving average
AVT	–	Audio-Video Transport Working Group
B2BUA	–	Back-to-Back User Agent
BTS	–	Base Transceiver Station
CC	–	CSRC count
CDMA	–	Code Division Multiple Access
CDMA2000	–	Code Division Multiple Access 2000
CELP	–	Code Excited Linear Prediction
CNAME	–	Canonical Name
CNG	–	Comfort Noise Generator
CoS	–	Class of Service
CRLF	–	Carriage Return Line Feed
CS-ACELP	–	Conjugate-Structured Algebraic Code-Excited Linear Prediction
CSRC	–	Contributing SouRCe
DDoS	–	Distributed Denial-of-Service
DLP	–	Data Loss Prevention
DNS	–	Domain Name System
DoS	–	Denial-of-Service
DSCP	–	Differentiated Services Code Point
DSP	–	Digital Signal Processor
DSSS	–	Direct Sequence Spread Spectrum

DTX	–	Discontinuous Transmission
EDGE	–	Enhanced Data rates for GSM Evolution
ETSI	–	European Telecommunications Standards Institute
FDDI	–	Fiber Distributed Data Interface
FDMA	–	Frequency Division Multiple Access
FHSS	–	Frequency-Hopping Spread Spectrum
GPRS	–	General Packet Radio Service
GSM	–	Global System for Mobile Communications / Groupe Spécial Mobile
H-ARQ	–	Hybrid-Automatic Re- peat Request
HFA	–	Hipath Feature Access
HICCUPS	–	Hidden Communication System for Corrupted Networks
HSDPA	–	High-Speed Downlink Packet Access
HTTP	–	HyperText Transfer Protocol
IAX/IAX2	–	Inter-Asterisk eXchange / Inter-Asterisk eXchange 2
ICE	–	Interactive Connectivity Establishment
ICMP	–	Internet Control Message Protocol
IETF	–	Internet Engineering Task Force
iLBC	–	Internet Low Bitrate Codec
ILP	–	Intelligent Loss Prevention
IMP	–	InterMessage Processor
IMS	–	P Multimedia Subsystem
IP	–	Internet Protocol
IPP	–	IP precedence
IPv6	–	Internet Protocol version 6
IS	–	Information System
ISDN	–	Integrated Services Digital Network
ISIM	–	IP Multimedia Services Identity Module
ISO/OSI	–	International Organization for Standardization / Open Systems Interconnection model
ITU-T	–	ITU Telecommunication Standardization Sector
LACK	–	Lost Audio paCKets steganography
LAN	–	Local Area Network
LD-CELP	–	Low Delay-Code Excited Linear Prediction
LOC	–	Geographic User Location (RTCP)
LP	–	Linear Prediction
LPC	–	Linear Predictive Coding
LSB	–	Least Significant Bit
LTE	–	Long Term Evolution

LTE-A	–	Long Term Evolution-Advanced
MAC	–	Media Access Control
MD5	–	Message-Digest algorithm 5
MGCP	–	Media Gateway Control Protocol
MMUSIC	–	Multiparty Multimedia Session Control
MOS	–	Mean Opinion Score
MP-MLQ	–	Multi-Pulse Maximum Likelihood Quantization
NAT	–	Network Address Translation
NBAD	–	Network Behavior Anomaly Detection
NGN	–	Next Generation Network
NIDS	–	Network Intrusion Detection Systems
NIPS	–	Network Intrusion Prevention Systems
NMT	–	Nordic Mobile Telephony
P2P	–	Peer-To-Peer
PBX	–	Private Branch Exchange
PCM	–	Pulse-Code Modulation
PRIV	–	Private extensions SDES item
PSTN	–	Public Switched Telephone Network
PT	–	Payload Type
QoS	–	Quality of Service
RR	–	Receiver Report
RSA	–	Rivest-Shamir-Adleman cryptosystem
RSU	–	Remote Subscriber Unit
RSVP	–	Resource Reservation Protocol
RTCP	–	Real-Time Control Protocol
RTP	–	Real-time Transport Protocol
SB-ADPCM	–	Sub-Band Adaptive Differential Pulse Code Modulation
SBC	–	Session Border Controller
SCCP	–	Skinny Call Control Protocol
SCTP	–	Stream Control Transmission Protocol
SDES	–	Source Description
SDP	–	Session Description Protocol
SID	–	Silence Insertion Description
SIP	–	Session Initiation Protocol
SMS	–	Short Message Service
SMTP	–	Simple Mail Transfer Protocol
SN	–	Sequence Number
SONET/SDH	–	Synchronous Optical Networking and Synchronous Digital Hierarchy

SPIT	–	SPam over Internet Telephony
SR	–	Sender Report
SSRC	–	Synchronization Source
TCP	–	Transmission Control Protocol
TLS	–	Transport Layer Security
ToS	–	Type of Service
TS	–	TimeStamp
UA	–	User Agent
UAC	–	User Agent Client
UAS	–	User Agent Server
UDP	–	User Datagram Protocol
UMTS	–	Universal Mobile Telecommunications System
URI	–	Uniform Resource Identifier
USIM	–	Universal Subscriber Identity Module
UTM	–	Unified Threat Management
VAD	–	Voice Activity Detection
VDID	–	Voice and Data Integration Device
VoATM	–	Voice over Asynchronous Transfer Mode
VoFR	–	Voice over Frame Relay
VoIP	–	Voice over IP
VoLTE	–	Voice over LTE
VoWiFi	–	Voice over WiFi
VoWLAN	–	Voice over WLAN
WAN	–	Wide Area Network
WLAN	–	Wireless Local Area Network
WWW	–	World Wide Web
xDSL	–	Digital subscriber line
XMPP	–	eXtensible Messaging and Presence Protocol

Obsah

Seznam tabulek

Seznam obrázků

Seznam výpisů zdrojového kódu

1 Úvod

V současném rozmachu datových sítí a služeb s nimi spojenými, hraje stále důležitější roli bezpečenost přenášených dat. Jednou z nejvíce progresivních služeb, které tyto sítě nabízejí je IP telefonie, využívající protokoly používané v rámci sítě Internet. V souvislosti s tím se objevují reálné hrozby jako odcizení citlivých osobních dat, úniky interních informací, firemních dat či přetížení serveru v jehož důsledku je způsobena nedostupnost služby, apod. Jednou z využívaných možností pro přenos citlivých či tajných dat je využití skrytého kanálu v rámci komunikace. Vytvářením skrytých kanálů se zabývá vědní disciplína nazývaná steganografie, která se dá také využít v rámci IP telefonie. Výsledky výzkumu v této oblasti, publikované týmem prof. Szczypiorského z Varšavské univerzity ukazují způsob, jak vytvořit skrytý kanál v SIP signalizaci. Proti technikám založených na výše zmíněném principu, se používají systémy IDS, které jsou schopny detekovat anomálii, ovšem ne za všech podmínek. Jedním z přístupů pro detekování anomálií je behaviorální analýza síťového provozu, která je schopna na základě statistických algoritmů víče či méně danou anomálii detekovat.

Cílem diplomové práce je rozvést myšlenky prof. Szczypiorského v oblasti využití specifických polí v hlavičce protokolu SIP a také se zaměřit na využití vložených utajených dat do RTP toku.

Tato práce pojednává o problematice internetové IP telefonie a jejímu využití pro steganografické účely. V úvodní části je probrána problematika IP telefonie a technologií, na kterých je postavená, problematika steganografie s popisem vývoje, technik využívaných v oblasti IP telefonie a shrnutí dosavadních úspěchů v této oblasti. V následující části jsou popsány metody pro detekci anomálií, systémy IDS, jejich klasifikace, popis využitých algoritmů Holt-Winters, Holt-Winters:Brutlag, Naive Bayes, využití softwarové sondy SNORT s preprocessorem Anomaly Detection a implementace. V části zvané Realizace výměny informací pomocí vloženého pole v SIP signalizaci se probírá samotný protokol SIP a injektování tajných dat do hlavičky, za pomocí generátoru SIPp, který slouží pro generování SIP zpráv. V další části práce je popsána metoda, která umožňuje přenos textových informací v rámci toku RTP protokolu. V poslední části se zabývám zhodnocením dosažených výsledků. V závěru je zhodnocen přínos této práce.

2 Úvod do VoIP a steganografie

V této kapitole se zabývám historií, vývojem a současným stavem technologie Voice over Internet Protocol a metodou ukrývání tajných informací, která se nazývá steganografie. Podrobněji jsou popsány principy digitalizace, komprese, paktizace a přenosu dat, které se využívají v rámci internetové telefonie. Dále je probrána problematika steganografie. Její vznik, princip, klasifikace, techniky, analýza. Také jsou probrány omezení lidského vnímání, které plynou z vlastností, které naše orgány mají. V poslední časti této kapitoly se zabývám průnikem těchto technologii, kde jsou popsány dosavadní úspěchy steganografie v oblasti VoIP.

2.1 VoIP

Voice over Internet Protocol (VoIP) je sada technologií, které zajišťují převod analogového audio signálu do binárního kódu, kdy je výsledný digitalizovaný signál komprimován pomocí psychoaktustických algoritmů, rozdělen na jednotlivé pakety a přenášen v těle paketů (jako souvislý proud) rodiny protokolů UDP/TCP/IP prostřednictvím počítačové sítě nebo jiného média, využívající protokol IP.

V první fázi (digitalizace) dochází k převodu analogového signálu na digitalní (A/D konverze) pomocí analogově-digitálního převodníku (může být součástí kodéru).

Ve fázi kódování a komprese již digitálních dat, se v oblasti telekomunikací využívá vzorkování s frekvencí 8 KHz a kvantování na 8 bitů, čímž vzniká bitrate (datový tok) 64kbps. Nicméně v dnešní době broadbandové konektivity lze využít i kodeky, které zpracovávají vyšší kvalitu vstupního signálu. V rámci převodu analogového signálu na digitální se zároveň přenášejí i nežádoucí a neslyšitelné části spektra (pro lidské ucho). Tyto části spektra se po aplikaci psychoakustického modelu při kódování signálu eliminují nebo se sníží jejich výskyt, což v důsledku sníží i objem přenášených dat (sníží se jejich datová velikost).

Další fází je proces paketizace, kdy jsou data rozdělena za pomocí signalizačních a média protokolů, na jednotlivé pakety (nejčastěji konstantní velikosti).

Následuje fáze přenosu dat, rozdělených na jednotlivé pakety pomocí IP sítě, za využití protokolu UDP/TCP/IP.

Poté, co jsou data přenesena po IP sítí následuje fáze depaketizace, která je přesně opačná od fáze paketizace. Dochází v ní ke sloučování paketů a získání původních dat.

Po získání původních dat dochází k jejich dekódování (dekomprimace), po které se nám sestaví původní digitální signál. Tato fáze je opět přesně opačná od fáze kódování (komprimace).

Finální fází je převod digitálního signálu zpět na analogový (D/A konverze), které se říká dedigitalizace. Po vykonání této fáze dostáváme původní analogový signál (výstupní analogový signál není absolutně totožný s původním vstupním analogovým signálem).

I přes použití pokročilých technologií se VoIP nevyhýbají problémy při komunikaci, dané původními technologiemi, na kterých jsou dnes postaveny základy VoIP. Nutnou podmínkou pro srozumitelné a spolehlivé VoIP telefonní spojení je zajištění tzv. kvality

služby, zkráceně označované QoS. VoIP technologie se využívá pro telefonování prostřednictvím Internetu, intranetu nebo jakéhokoliv jiného datového spojení.

2.1.1 Historie

Současné telekomunikační technologie prošly za přibližně 140 let od vynálezu telefonního přístroje obrovským vývojem. Telekomunikační vývoj pokračoval po vynálezu telefonního přístroje přes pevné analogové sítě, pevné digitální sítě, mobilní sítě, VoIP, VoWiFi, VoLTE. V průběhu tohoto vývoje probíhal i vývoj datových sítí, které reprezentuje globalní síť Internet. Během tohoto paralelního vývoje došlo ke konvergenci těchto sítí.

2.1.1.1 Výnález telefonu Když Alexander Graham Bell v roce 1875 vynalezl první telefonní přístroj, který umožňoval přenos lidského hlasu pomocí elektrické energie, znamenalo to převrat v mezilidské komunikaci. Patent na telefonní přístroj mu byl udělen v roce 1876. Než byl telefon vynalezen, předcházela mu řada neúspěšných pokusů o elektrický přenos zvuku. Primárně se pokoušel Bell vynalézt telegraf pro přenos více různých zpráv v jednom okamžiku, tzv. harmonický telegraf. V mikrofonu a sluchátku byla umístěna elektromagneticky prohýbaná membrána, která měla za úkol rozkmitávat cívku navinutou na ocelovém magnetu. Tímto způsobem byla poprvé přenesena lidská řeč. Všeobecně je s vynálezem telefonního přístroje spojován právě A.G. Bell, nicméně výzkumem přenosu lidské řeči ze zabývalo mnoho vědců. Ve stejný den, o pár hodin později, přišel se stejným vynálezem Elisha Gray, který ho objevil v roce 1874 nezávisle na Bellovi, ale nejdříve ho postupně zdokonaloval. V patentovém sporu s Bellem však prohrál. V současné době je považován za objevitele telefonního přístoje Antonio Meucii, který ho poprvé prezentoval v New Yorku v roce 1860. Zpráva byla zveřejněna v lokálním italsky psaném tisku. V principu se jednalo o elektromagnetický mikrofon/sluchátko. Membrána s permanentním magnetem byla rozkmitaná zvukem, která převedla pohyb na elektrický proud. Poté byl po drátech signál přenesen do stejného zařízení a elektrický signál přeměnen na akustický. Spor s Bellem o prvenství vynálezu byl odkládán až do Meucciho smrti, kdy byla kauza zrušena. V roce 2002 byl Meucci rezolucí kongresu USA uznán prvním vynálezcem telefonu.

V roce 1877 Bell založil první telefonní společnost s názvem Bell Telephone Company, která je dnes známá jako American Telephone & Telegraph Company (AT&T).

O rok později (1878) byla v Paříži patentována první telefonní síť na světě.

Do třicátých let 20. století, bylo možné telefonovat pouze mezi dvěma koncovými zařízeními přes spojovací ústřednu. Hovor byl přepojován ručně pomocí spojovatelky.

2.1.1.2 Pevné analogové telefonní sítě Pevná analogová telefonní síť, anglicky PSTN (Public Switched Telecommunication Network), je telefonní síť, která přenáší audio signál převedený na analogový elektrický signál. Po vedení je dále přenášen v podobě střídavého napětí. Je zde využit princip přepínaní okruhů. To znamená, že mezi koncovými zařízeními se vytvářelo přímé spojení (analogový přenosový okruh). Kvůli optimalizaci

přenosové kapacity sítě, měly hovory omezené frekvenční spektrum od 300 Hz - 3400 Hz, které neomezuje srozumitelnost. Omezené frekvenční spektrum bylo realizované pomocí pásmové propusti. V důsledku těchto frekvenčních omezení, umožňovalo realizovat více hovorů za použití frekvenčního multiplexu. Koncová zařízení byla připojena k pobočkové ústředně (PBX), ta je dále připojena k tranzitní ústředně, která je propojena s jinou tranzitní ústřednou. Vzhledem k omezenému dosahu PBX, kvůli fyzikálním zákonům, zejména velkému útlumu, je efektní rádius 3-5 Km. Aby se zvýšil dosah sítě resp. vzdálenost od ústředny, připojovalo se mezi koncové zařízení a PBX jednotky RSU (Remote Subscriber Unit).

2.1.1.3 Pevné digitalní telefonní sítě Další evoluční stupeň telefonních sítí byla jejich digitalizace. Přechod na pevnou digitalní telefonní síť byl plynulý. Nejdříve prošly modernizací ústředny, poté samotná síť a nakonec i koncové zařízení. Díky těmto modernizacím byly uživatelům nabídnuty nové služby. Začala se používat nová technologie ISDN (Integrated Services Digital Network), která umožňuje plně digitalní přenos až k účastníkovi. Oproti původní analogové sítí je ISDN spolehlivější, kvalitnější a umožňuje vyšší přenosovou rychlost. Jedná se o multimediální komunikaci. Standardní ISDN přípojka obsahovala 2 nezávislé B kanály s rychlostí 64 kbps a jednoho D kanálu s rychlostí 16 kbps určeného pro přenos signalizace. V dnešní době je již tato technologie zastaralá. V současnosti se stává nejdůležitějším připojením připojení do sítě Internet. K tomu se využívají technologie xDSL, které mají připojení symetrické i asymetrické. V důsledku konvergence začínají klasické telefonní sítě pozbývat svůj primární účel, kterým je přenosu hlasu. V současné době je stále zřetelněji vidět, že dominantní je poskytování datových služeb (připojení do Internetu).

2.1.1.4 Mobilní telefonní sítě Počátky mobilních sítí (celulární / buňková rádiová síť) se datují od 70. let 20. století. Systém pro svou funkci využívá radiofrekvenční spektrum. Systém základových stanic (BTS - Base Transceiver Station) tvoří síť vzájemně překrývajících se malých buněk, které jako celek ukrývají určité území. Celulární sítě pracují ve frekvenčním spektru od 300 MHz do 3 GHz. Mobilní sítě prošly několika generacemi vývoje. Bezdrátové mobilní sítě se vyvíjely paralelně s klasickými pevnými telefonními sítěmi. Na počátku devadesátých let 20. století se sítě začaly rozšiřovat do povědomí širší veřejnosti, čímž se mobilní telefonní sítě staly hlavním konkurentem pevných telefonních sítí. Díky levnější realizaci infrastruktury pomocí BTS, je rozvoj těchto sítí progresivnější než rozvoj pevných telefonních sítí.

2.1.1.4.1 1G - NMT, AMPS, TACS 80. letech 20. století se dostala na trh první generace mobilní sítě kterou reprezentovaly systémy NMT (Nordic Mobile Telephony), americký systém AMPS(Advanced Mobile Phone System) a britský TACS(Total Access Communication System). Tyto sítě byly čistě analogové, využívající přístupovou metodu FDMA (Frequency Divison Multiple Access). Tyto systémy však nebyly mezi sebou kompatibilní. S rostoucím zájmem uživatelů bylo zřejmé, že kapacitní možnosti sítí 1G budu brzy

vyčerpány. To vedlo k potřebě rozšířit nějaký stávající systém, nebo vytvořit nový jednotný globální systém, vyhovující kapacitním požadavkům. Tímto způsobem se zrodil evropský standard druhé generace GSM.

2.1.1.4.2 2G - GSM, CDMA Druhou generací byl v roce 1989 evropskou telekomunikační standardizační institucí (ETSI) definován standard GSM (Global System for Mobile Communications) jako nový mezinárodní digitální komunikační buňkový standard. GSM přináší změnu metody přístupu a to na komplexnější kombinaci TDMA (Time Divison Multiple Access) a FDMA (Frequency Divison Multiple Access). Ve Spojených státech v této době začal dominovat systém CDMA (Code Division Multiple Access). Systémy druhé generace používají digitální přenos dat, nižší vysílací výkon a tudíž menší velikost buněk, lepší odolnost vůči chybám a zvýšení bezpečnosti. Primárním účelem těchto sítí je stále přenos hlasových služeb, ačkoli s novými systémy byly představeny nové služby jako SMS (Short Message Service) nebo email.

2.1.1.4.3 2.5G - GPRS Mezi druhou a třetí generací jsou tzv. 2.5G a 2.75G. Tato kategorie mezigeneračních systémů rozšiřuje stávající systémy 2G o nové komponenty a služby. 2.5G rozšiřuje GSM o paketově orientované datové přenosy prostřednictvím nadstavbové technologie GPRS (General Packet Radio Service), která také rozšířila stávající systém o nové komponenty.

2.1.1.4.4 2.75G - EDGE, CDMA2000 2.75G je označována technologie EDGE (Enhanced Data Rates for GSM Evolution), zvyšující přenosovou rychlost použitím techniky modulace (8-PSK). Tak je pro americký CDMAOne byla vytvořená nadstavba v podobě systému CDMA2000 1xRTT.

2.1.1.4.5 3G - UMTS, CDMA2000 (3X) Představitelem třetí generace je systém UMTS (Universal Mobile Telecommunication System). Tento evoluční stupeň vývoje GSM přináší vysokorychlostní a vysokokapacitní přenos dat, efektivnější využití přenosového spektra a nové multimediální služby. Maximální přenosová rychlost je 384 kbps. Americkým představitelem je CDMA2000 (3X) pracující s metodu přístupu CDMA.

2.1.1.4.6 3.5G - HSDPA 3.5G je označována technologie HSDPA (High-Speed Downlink Packet Access), která podstatně zvyšuje přenosovou rychlost pro downlink. Může dosahovat maximální teoretické rychlosti 14,4 Mbps. Této rychlosti bylo možné dosáhnout několika inovacemi architektury sítě, díky čemuž se snížila latence, zvýšila se rychlost reakce na změnu kvality kanálu a zpracování H-ARQ (Hybrid-Automatic Repeat Request).

2.1.1.4.7 3.9G - LTE 3.9G představuje technologie LTE (3GPP Long Term Evolution), která je určena pro vysokorychlostní datové přenosy mobilních sítích. Maximální teoretická rychlost pro downlink je 172,8 Mbps a uplink 57,6 Mbps.

2.1.1.4.8 4G - LTE Advanced Poslední generací (4G) mobilních sítí je LTE Advanced (LTE-A), která byla standardizovaná v roce 2011. Maximální teoretická přenosová rychlost pro downlink je 3 Gbps a uplink 1,5 Gbps (LTE-Advanced - LTE Release10)[21]. Oproti LTE se zvýšila datová rychlost v downlink i uplink pomocí agregace více pásem a vícecestného přenos signálu (MIMO 8x Tx), efektivnější využití frekvenčního pásma a snadnější pokrytí signálem uvnitř budov pomocí Femtocell se sofistikovanou koordinaci buněk.

2.1.1.4.9 VoLTE VoLTE (Voice over LTE) je technologie pro přenos audio dat pomocí sítí Long Term Evolution (LTE). Pro přenos je využívána infrastruktura IP Multimedia Subsystem (IMS) se zvláštními profily pro řídící a mediální rovinu hlasové služby v LTE. Cílem LTE je přenos audio dat jako datové streamy, bez závislosti na starších sítích s přepínáním okruhů. Výhodu je lepší využití kapacity přenosového pásma, v důsledku menší hlavičky VoLTE paketu než jakou disponuje VoIP/LTE. První VoLTE služby byly spušteny v Jížní Koreji a ve Spojených státech v roce 2012. První komerční VoLTE na světě spustila firma SingTel v Singapuru 31.května 2014.

2.1.1.5 Datové sítě Datovou sítí (data network) je dnes chápána síť, která slouží k přenosu digitálních dat, rozdělených a přenášených ve formě paketů, tedy na principu přepojování paketů (packet switching). Narozdíl od toho, přenos dat po telefonní sítí probíha po jednotlivých bitech (stream) na principu přepínání okruhů.

2.1.1.5.1 VoFR VoFR (Voice over Frame Relay) je protokol linkové vrstvy pro přenos audio signálu pomocí sítě Frame Relay. Frame Relay je standartizovaná WAN (Wide Area Network) technologie, která specifikuje fyzickou a linkovou vrstvu digitálních telekomunikačních kanálů sítě na principu přepínání datových jednotek - rámců (paketový přenos). Používá se v případech, kdy je potřeba propojit telefonní ústředny nebo připojit telefonní přístroj na vzdálenou ústřednu. Frame Relay vytváří virtuální spojení point-point v rámci virtuálního okruhu, což umožňuje spojení pouze s pevně definovaným počtem koncových bodů. Výhodou oproti VoIP je ve efektivnějším využití přenosové kapacity, protože hlasové samply se vkládají přímo do rámce Frame Relay bez účasti IP, UDP, RTP hlaviček. Dále je výhodou garance minimalní a maximální přenosové rychlosti a optimalizace pro přenos audio dat. Problémy v rámci přenosu hlasu můžou způsobit přenosy dlouhých rámců v síti. Pro eliminaci tohoto problému má Frame Relay pro podporu přenosu hlasu zařízení VDID (Voice and Data Integration Device), které zajišťuje fragmentaci dlouhých datových rámců a prioritizaci rámců přenášející hlas. VoFR zařízení pro přenos hlasu nejsou detailně standardizovaná, což není problémem, protože v rámci VoFR mezi sebou komunikují dvě zařízení, které jsou v síti jednoho providera, v důsledku toho není třeba řešit kompatabilitu s jinými zařízeními.

2.1.1.5.2 VoATM VoATM (Voice over Asynchronous Transfer Mode) je standard pro přenos audio signálu pomocí vysokorychlostní síťové architektury. Je základní protokol používaný v rámci SONET/SDH páteřních linek pevné veřejné telefonní sítě (PSTN) a

Integrated Services Digital Network (ISDN). Byl navrhován telekomunikačními společnostmi, v důsledku toho tento standard implementoval pokročilé technologie pro přenos hlasu pomocí datové sítě, pro které byl primárně určen. Obsahuje technologii kontroly kvality služby (QoS - Quality of Service) pro přenos audia a videa. Využívá asynchronní časový multiplex. Umožňuje přenos IP datagramů. Pracuje na principu přepojování paketu fixní velikosti (cell) užitím virtuálních okruhů. Přenášená data jsou rozdělena na buňky (cell), které mají narozdíl od paketů pevnou délku (53 Byte; 48 Byte data a 5 Byte záhlaví). ATM technologie realizuje spojení mezi dvěma body ještě před samotnou výměnou dat. Využívá se u technologie DSL. ATM je složitá a drahá technologie, což vede k jejímu postupnému zániku.

2.1.1.5.3 Přepínání okruhů Přepínání okruhů (circuit-switched networks) je princip přenosu dat v síti, kdy se pro komunikaci alokuje kanál od jednoho koncového účastníka až ke druhému koncovému účastníkovi, po kterém probíhá daná komunikace po celou dobu spojení. Každá ústředna, přes kterou komunikace prochází, musí tento kanál pro hovor alokovat. Pokud jakákoliv ústředna nemůže tento kanál alokovat, volajícímu je odeslán signál obsazení linky. Výhodou tohoto řešení je stabilní kvalita hovoru, která nekolísá v závislosti na šířce pásma a nižší nároky na režii přenosu. Z toho důvodu se některé sítě ve svých nižších vrstvách vracejí k tomuto řešení. Naopak negativní vlatností je malá efektivita využití přenosového pásma. Princip přepínání okruhů se využívá u telefonních sítí i u mobilních sítí. V současnosti je dominatní princip přepojování paketů.

2.1.1.5.4 Přepojování paketů Přepojování paketů (packet-switched networks) narozdíl od přepínání okruhů, princip přepojování paketů rozděluje jednotlivá data na segmenty stejné velikosti, které se jmenují pakety. Každý paket v hlavičce nese informaci o adrese odesílatele, adrese cíle, kontrolní součet a je přenášen, doručován samostatně tzn. že každý paket může být doručován jinou cestou. Směrování v uzlech sítě zajišťují přepínače (IMP - InterMessage Processor) jako např. switch, router. Router obsahuje interní routovací tabulku, kde má přehled o okolních zařízeních (routerech). Z nich vybere to, ke kterému má nejkratší (nejlevnější) cestu pro doručení paketu k cíli, kde jsou pakety sestaveny do původních dat. V sítí Internet zajišťuje přepravu paketů protokol IP umístěný v síťové vrstvě a sestavení původní zprávy zajišťuje protokol TCP umístěný v transportní síťové vrstvě, který zajišťuje správu virtuálního okruhu. Výhodou tohoto typu přenosu je efektivnější využití sířky přenosového pásma v důsledku toho, že není nijak kontrolováno, zda byl daný paket doručen v pořádku. Další neméně podstatnou výhodou je robustnost tohoto typu přenosu. Za předpokladu, že síť obsahuje redundantní spoje, jsou v případě výpadku uzlu, pakety přepravované alternativní cestou. Mezi nevýhody patří kolísání kvality hovoru, které je dáno tím ze jednotlivé pakety mohou k cíli dorazit v jiném pořadí něž v jakém byly odeslány. Řešením tohoto problému je technologie QoS, která bude probrána podrobněji dále.

2.1.1.6 IP sítě Termínem IP sítě označujeme počítačové (datové) sítě s principem přepojování paketů, které pro přenos dat užívají protokol IP (Internet Protocol). IP protokol

je routovací protokol, který přenáší data z vyšších vrstev síťového modelu. Data z protokolů vyšších vrstev jsou zapouzdřena do IP paketu. Každý IP paket má svoji cílovou zdrojovou IP adresu, zároveň každé zařízení připojené do sítě má svoji unikátní IP adresu. Vzhledem k abstrakci a zapouzdření poskytované TCP/IP a ISO OSI modelu, může být IP protokol transparentně používán na 1.(fyzické) a 2.(linkové) vrsvě. To umožňuje IP protokolu být použit s technologiemi Ethernet, FDDI, Token Ring, ATM, Frame Relay a ostatními technologiemi pracujících na 2.linkové vrstvě. Tento protokol vyniká svojí jednoduchostí a možnostmi jeho implementace. To vedlo k využití IP protokolu pro přenos audio dat po datové sítí. Tím ale vyvstaly problémy spojené v případě přetížení sítě. Pokud je IP síť přetížená, nedokáže garantovat postupně doručování paketu v pořadí, v jakém byly odeslány, což je pro IP telefonii klíčová vlastnost. IP protokol funguje na principu best-effort service, tzn. že negarantuje odesílateli doručení dat k cíli ve správném pořadí, včas a v pořádku. Z tohoto důvodu se pro IP telefonii využívají další protokoly, které zajišťují spolehlivost přenosu dat. Mezi nejrozšířenější komunikační protokoly patří např. H.323, SIP, IAX, MGCP.

2.1.1.7 VoIP Za počátek internetové telefonie je všeobecně považován rok 1995. V únoru tohoto roku uvedla izraelská firma Vocaltec Ltd. revoluční produkt s názvem "Internet Phone". Jednalo se o vůbec první čistě softwarové řešení, které umožňovalo volat mezi počítačovými stanicemi připojenými do sítě Internet. Jednalo se o komerční software. K provozu aplikace (klienta) byla potřeba vytáčená linka s rychlosti minimálně 14,4 kbps. Doporučné hardwarové nároky, které byly nezbytné pro plynulý chod aplikace byly: procesor architektury x86, 486SX o frekvenci 25 MHz, 8 MB RAM.

2.1.2 Digitalizace - převod A/D

Abychom mohli přenášet audio konverzaci pomocí datové sítě, je potřeba převést hlas do digitální podoby. Tomuto procesu se říká digitalizace nebo analogově-digitální konverze (A/D převod). Zvuk je z fyzikálního hlediska vlnění (vibrace) o určité frekvenci, která rozkmitává molekuly vzduchu. Pokud je tato frekvence v rozsahu 20 Hz - 20 kHz (rozsah se s narůstajícím věkem zmenšuje), a jsme dostatečně blízko zdroje, který produkuje zvuk, můžeme jej pomocí sluchového orgánu slyšet. Datové sítě používají pro přenos informace elektrické nebo optické impulsy, které vyjadřují danou binární hodnotu. Převod analogového signálu na digitalní zajišťuje tzv. digitální signální procesor (DSP - Digital Signal Processor), který je na zpracování takových signálů specializovaný. Princip převodu spojitého signálu na diskrétní je složen ze dvou fází. V první fázi se provede vzorkování signálu, ve druhé poté následuje kvantování.

Při vzorkování analogového signálu se rozdělí vodorovná osa, která reprezentuje čas, na rovnoměrné úseky, a z každého úseku je odebrán jeden vzorek. V závislosti navzorkovací frekvenci se mění kvalita digitalizovaného signálu. Čím vyšší je vzorkovací frekvence tím, je kvalitnější výsledný signál. Je zřejmé, že z původního signálu ztratíme mnoho detailů, protože místo spojitého signálu, který lze donekonečna zvětšovat získáváme pouze množinu diskrétních bodů s intervalem odpovídajícímu použité vzorkovací

frekvenci. V rámci vzorkování může dojít k nevratnému zkreslení signálu díky jevu, kterému se říká aliasing. Pokud je frekvence analogového signálu vyšší než polovina vzorkovací frekvence (Nyguistova frekvence), jak říká Shannonův teorém, dojde k aliasingu. Tomu se dá zabránit tzv. antialiasing filtrem realizovaným dolní propustí zařazenou před konvertor.

Počítače a ostatní zařízení zpracovávající digitální signál dokážou vyjádřit pouze čísla s omezenou přesností, vzniká potřeba navzorkované hodnoty upravit i na svislé ose. Hodnota vzorku lze vyjádřit pouze o určitých kvantech, odtud název fáze kvantování. Na svislé ose může veličina nabývat pouze celočíselné hodnoty. Aby bylo možné určit výslednou hodnotu daného vzorku je prostor kolem jednotlivých hodnot rozdělen na toleranční pásy. Jakmile se vzorek dostane do tolerančního pásu, je mu při kvantování přiřazena daná hodnota. Kvantované hodnoty se většinou liší od skutečných navzorkovaných hodnot. Velikost této chyby se pohybuje intervalu v +1/2 až -1/2 kvantizační úrovně. Počet kvantizačních úrovní A/D převodníku je roven 2^N, protože je digitalní signál zpracováván pomocí binární číselné soustavy. Opět platí, že čím více kvantizačních úrovní máme, tím je výsledný digitální signál přesnější, ale také má větší datovou náročnost. S kvantizací nám vzniká nežadoucí efekt ve formě tzv. kvantizačního šumu, což je náhodný signál, který nám říká, jaká je velikost chyby v jednotlivých vzorcích. Vyjadřuje se jako poměrné číslo v dB, jako poměr užitečného signálu / kvantizačnímu šumu (1).

$$SNR_{A/D} = 20 \cdot log2^N[dB] \tag{1}$$

Standardní analogové telefonní systémy využívají frekvenční rozsah 300Hz - 3400Hz, který je ověřen jako dostatečně kvalitní pro přenos hlasového spektra a je šetrný na čerpání systémových zdrojů.

2.1.3 Kódování / Komprese

Kódování je proces, při kterém dochází aplikováním daného algoritmu ke kompresi signálu podle daných specifik jednotlivých kodeků. Hlavním účelem kódování signálu je snížení přenosové rychlosti (bitrate), v jejímž důsledku se sníží nároky na přenosovou šířku pásma datové linky. Kodeky mají různou úroveň komprese od níž je závislý datový tok, nesmíme však opomenout i čas, který daná komprese zabere. Kódování s minimální kompresí je rychlejší, než kódování s vysokým kompresním poměrem. Tento fakt hraje významnou roli v kvalitě hovoru a jeho nárocích na datovou infrastrukturu. Základní parametry u kodeků jsou: míra komprese, algoritmus, přenosová rychlost (kbps), kvalita hlasu, míra zpoždění při kódování/dekódovaní. Existuje mnoho různých typů kodeků, které se dají rozdělit do tří základních skupin. První skupinou jsou kodéry tvarového průběhu (waveforms coders), které kódují zdrojový tvar vlny. Tento typ kodeku vytváří velmi kvalitní hlasový signál. Druhou skupinou jsou vocodéry (vocoders, voice coders), kde se jedná o parametrické zdrojové kódování. Kodeky této skupiny vynikají svým nízkým datovým tokem, ale hlasový signál je méně kvalitní, než u waveform kodeků. Třetí skupinou jsou hybridní kodéry (hybrid coders), kde jde o hybridní zdrojové kódování s využitím předností obou předchozích skupin kodeků. K hodnocení kvality kodérů se vy-

užívá stupnice MOS (Mean Opinion Score) (Tabulka 1). Jako výchozí hodnota zdrojového kódování se používá PCM (Pulse Code Modulation) - pulzně kódová modulace.

MOS	Kvalita	Zhoršení
5	Excelentní	Nepostřehnutelné
4	Dobrá	Znatelné, ale nerušivé
3	Uspokojivá	Mírně nepříjemné
2	Špatná	Nepříjemné
1	Nedostatečná	Velmi nepříjemné

Tabulka 1: MOS (Mean Opinion Score) - stupnice hodnocení kvality koderů

Důležitou hodnotou je také míra zpoždění algoritmu[ms] a míra zpoždění kodeku[ms], které významně ovlivňují výslednou kvalitou hovoru. Tyto hodnoty udávají o kolik milisekund se zpozdí signál při aplikovaní daného algoritmu a využití daného kodeku. Kodeky, které mají vyšší míru komprese původního signálu, jsou náročnější na hardwarové zdroje.

2.1.3.1 G.711 - PCM (Pulse Code Modulation) - pulzně kódová modulace Je modulační metoda převodu analogového audio signálu na digitalní signál, která byla vytvořena v roce 1937 Alecem Reevsem. Převádí analogový audio signál ve frekvenčním rozsahu 300 Hz - 3400 Hz, který je na digitální signál převáděn ve třech krocích. Vzorkovaním, kvantováním a kódováním. Použitý vzorkovací kmitočet je 8khz a kvantování na 8 bitů, při kterém se vytvoří digitální signál o přenosové rychlosti 64 kbps. Komerční název tohoto kodeku je G.711.

V roce 2009 byla schválena nová verze kodeku, která poskytuje bezeztrátovou kompresi (lossless compression) pod označním Lossless compression of G.711 pulse code modulation. Bývá také označován jako G.711 LLC nebo G.711.0.

V roce 2008 byl schválen standard G.711.1, který umožňuje rozšíření G.711 o možnost použít vzorkovací frekvenci 16 kHz a vyšší kvalitu pomocí tří vrstev. Využívá datový tok 64, 80 nebo 96 kbps. Frekvenční pásmo je 50 - 7000 Hz. V roce 2010 došlo k rozšíření frekvenčního pásma na 50 - 14000 Hz (Více v části: Využití RTP pro přenos textových informací).

2.1.3.2 G.722/G.722.1 - SB-ADPCM (Sub-Band Adaptive Differential Pulse Code Modulation) Tento kodek bývá nazýván širokopásmovým, protože využívá dvojnásobnou vzorkovací frekvenci (16 kHz), frekvenční pásmo je rozšířeno na 7kHz. Díky tomuto rozšíření je dosaženo mnohem vyšší kvality hovoru, a tím i srozumitelnější přenos hlasu, který je důležitý u audiokonferencí. Dále přináší výhodu z hlediska bezpečnosti (ochrana před falešným volajícím). Jde o embedded kodér, u kterého se muže volně přepínat přenosová rychlost mezi 48, 56 a 64 kbps bez předchodího oznámení kodéru. Vzhledem k tomu je algoritmické zpoždění jen 1,5 ms, je vhodný pro profesionální použití. Pro kódování je použit algoritmus SB-ADPCM (Sub-Band Adaptive Differential Pulse Code Modulation).

Nejnovější verze širokopásmového kodeku nese označení G.722.1, který pracuje s přenosovou rychlostí 24 kbps nebo 32 kbps. Navrhla jej společnost PictureTel, kterou později koupila společnost Polycom, která komerčně prodává kodek s přenosovou rychlostí 16 kbps (ver. kodeku Siréna). Rámce mají velikost 20 ms s predikcí na 20 ms. Verze s přenosovou rychlostí 16kbps je podporována ve Windows Messenger.

2.1.3.3 G.723.1 - MP-MLQ (Multi-Pulse Maximum Likelihood Quantization) / ACELP (Algebraic Code Excited Linear Prediction) V počatcích VoIP byl tento kodek zvolen jako základní pro použití v protokolu H.323. Je také používán mobilními videotelefony UMTS 99 (standard H.324M). Jedná se o jeden z nejoblíbenějších kodeků spolu s kodekem G.729 pro VoIP. Používá rámce délky 30ms s predikcí na 7,5ms. Vyznačuje se velmi nízkým bitovým tokem. Obsahuje dva provozní režimy - 6,3 kbps (r63 - 24B rámec) a 5,3 kbps (r53 - 20B rámec), mezi kterými lze přepínat. Je zatížen 18-ti patenty.

Kodek G.723.1 podporuje detekci hlasové aktivity (VAD - Voice Activity Detection), diskontinuální přenos (DTX – Discontinuous Transmission) a generování výplňového šumu (CNG – Comfort Noise Generator).

Pro tento kodek jsou definovány dva kodeky:

- MP-MLQ (Multi-Pulse Maximum Likelihood Quantization) s bitovým tokem 6,3 kbps a MOS 3,9.
- ACELP (Algebraic Code Excited Linear Prediction) s bitovým tokem 5,3 kbps a MOS 3,62.

ACELP algoritmus vychází z CELP algoritmu, používá však specifickou algebraickou strukturu. Použitá algebraická knihovna může být velmi velká (větší než 50 b) a to bez nepřiměřeně velkých nároků na výpočetní výkon (RAM/ROM a CPU).

2.1.3.4 DPCM (Differential Pulse Code Modulation) Diferenciální pulzní kódová modulace, je modifikací PCM kódování, která se používá za účelem snížení datového toku. Rozdíl oproti PCM kódování je, že se nekódují navzorkovaná data, ale jejich rozíl oproti odhadnutému průběhu vývoje signálu. Díky vlastnostem hlasového ústrojí a traktu je možné vývoj signálu částečně odhadnout. Jelikož je navzorkovaný a odhadnutý průběh signálu podobný, má výsledný rozdíl mnohem menší dynamický rozsah, a tím jej lze zakódovat pomocí menšího počtu bitů. Výsledkem je snížení množství přenášených dat.

2.1.3.5 G.726 - ADPCM (Adaptive Differential Pulse Code Modulation) Adaptivní diferenciální pulzní kódová modulace je dalším evolučním krokem vycházejícím z DPCM. Generátor srovnávacího průběhu je adaptivní tzn., že se dokáže přizpůsobovat konkretní řeči, která se kóduje. Tento algoritmus je schopen snížit dynamický rozsah ještě více než DPCM a v důsledku je možné snížit počet bitů nutných k zakódování. Dále se adaptivně mění i vlastnosti kvantifikace pro danou charakteristiku řeči. Dostupné přenosové rychlosti jsou 16, 24, 32 a 40 kbps. Komerční název tohoto kodeku je G.726.

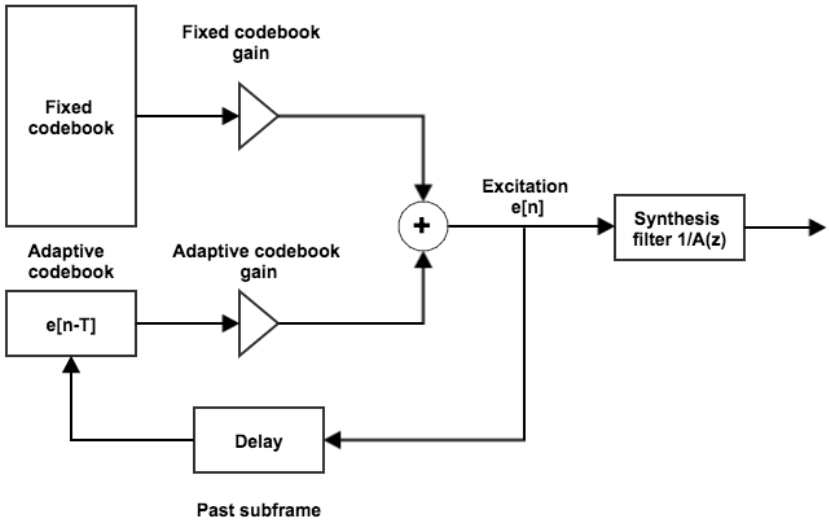
2.1.3.6 LPC (Linear Predictive Coding) Je kódovací algoritmus ztrátové komprese, který zajišťuje reprezentaci spektrální obálky digitalizovaného audio signálu v komprimované formě. Narozdíl od PCM a ADPCM, které vycházejí z kvantifikace průběhu signálu je metoda LPC jednou z nejefektivnějších způsobů analýzy a zpětné rekonstrukce audio signálu použitelného pro kodeky s nízkým datovým tokem (low-bit-rate codecs). Na vysílací straně probíhá analýza hlasového signálu a jeho komprese. Na přijímací straně dochází k obnově původního signálu s co nejmenší deformací, která je možná. LPC při provádění analýzy vychází ze znalostí hlasového ústrojí a hlasového traktu, kde je hlas generován hlasivkami, díky nímž je ovlivňován frekvenční průběh a intenzita generovaných vokálů. Pomocí krku a úst, které představují tubus, který způsobuje různé rezonance nazývané formanty. Charakteristická barva hlasu je dána počtem harmonických frekvencí. Při analýze hlasu se nejdříve oddělí rezonanční frekvence (formanty) pomocí filtrů (inverzní filtrace) a poté se provede analýza zbytku hlasu. Vzhledem k tomu, že se lidská řeč sestavuje z opakujících se zvukových elementů, je možné sestavit jejich databázi (slovník). Zbytek hlasu je poté asociován se záznamy v této databázi. Na stranu příjemce se přenášejí pouze odkazy, které směřují do této databáze a informace o charakteristice hlasu. Jejich zpětnou syntézou vzniká vysoce věrná podoba výchozího hlasu. Protože se vstupní signál mění v závislosti na čase, je nutné tento proces opakovat pro kratší časové intervaly, které se nazývají rámce. Pro zachovaní dostatečné kvality výsledné řeci se používá 30 - 50 rámců za sekundu. Audio signál kódovaný pomocí LPC je srozumitelný již při šířce pásma 2400bps.

2.1.3.7 CELP (Code Excited Linear Prediction) Je lineárně predikční kodek s vektorovým kvantováním budícího signálu. V paměti kodéru jsou uloženy soubory možných posloupností vstupního signálu. Je-li v kódové knize (Codebook) nalezena posloupnost odpovídající budícímu signálu, je příslušná adresa budící posloupnosti (která je uložena v kódové knize) binárně přenesena do přijímače. Přijímač podle přijaté adresy generuje na základě shodné kódové knihy příslušnou budící posloupnost pro LP (Linear Prediction) hlasový syntetizátor. Optimální vektor je určen užitím kritéria minima střední kvadratické chyby. CELP algoritmus poskytuje dobrou kvalitu přenášeného audio signálu již při šířce pásma 4,8kbps.

Díky tomuto algoritmu lze eliminovat problémy s přenosem tzv. sykavek, které vznikají při použití metody LPC.

Velkou nevýhodou toho algoritmu jsou jeho nároky na výpočetní výkon, při prohledávání kódových knih. CELP algoritmus vyžaduje procesor schopný zpracovat alespoň 20MIPS a prostor 40kB pro kódovou knihu. Kódovací latence je až 35ms, tím je omezeno použití CELP algoritmu pro přenosy na delší vzdálenosti a jejich vícenásobné použití. Z toho důvodu vznikly různé modifikace tohoto algoritmu (ACELP, CS-ACELP, LD-CELP), které eliminují vysokou míru latence.

2.1.3.8 G.728 - LD-CELP (Low Delay-Code Excited Linear Prediction) Kodek G.728 využívá algoritmus LD-CELP (Low Delay-Code Excited Linear Prediction), který je modifikací CELP algoritmu tím, že snižuje jeho latenci. Při šířce pásma 16 kbps má MOS



Obrázek 1: Model CELP syntézy řeči (dekodér)

skóre srovnatelné s G.726 při šířce pásma 32kbps. Tento kodek díky tomu, že vychází z CELP algoritmu není vhodný pro přenos hudby. V rámci H.320 je používán pro videokonference a H.323 videokonferenční systémy. Zde se využívá jeho nízké latence (625 μs - 2,5 ms).

Variantu s nižší přenosovou rychlostí (15,2 kbps) je přehrávač RealAudio vyvinutý v roce 1995 firmou RealNetworks.

Způsob kódování s nízkou latencí LD-CELP je založen na zpětnovazebním řízení adaptace s kódovacím zpožděním do 2ms a přenosovou rychlostí 16kbps. LD-CELP se od algoritmu CELP liší v řešení vlastního hlasového syntetizátoru a způsobu jeho řízení. U algoritmu CELP se přenášejí v časovém multiplexu lineárně predikční koeficienty vyčíslené v kodéru krátkodobou (STP - Short Term Predictor) a dlouhodobou (LTP - Long Term Predictor) analýzou, které slouží v příjímači k periodickému nastavování LPC (Linear Prediction Coefficient) filtru hlasového syntetizátoru. Narozdíl od toho mezi kodérem a dekodérem LD-CELP se tato data nepřenášejí a kodér i dekodér jsou řízeni zpětnovazebně.

2.1.3.9 G.729/G.729A - CS-ACELP (Conjugate-Structured Algebraic Code Excited Linear Prediction) Jedná se o velmi populární kodek, který se používá pro přenos hlasu pomocí Frame Relay, který byl schválen v roce 1996. Jedná se o oblíbený audio kodek pro VoIP spolu s G.723. Kodek pracuje se vzorky o délce 10ms, a pro kompresi audio signálu využívá algoritmus CS-ACELP (Conjugate-Structured Algebraic-Code Excited Linear Prediction). Predikce je 5ms. Dosahovaný bitový tok je 80-ti bitových rámců má hodnotu 8kbps, k dispozici je i rozšíření pro 6,4kbps a 11,8kbps, které umožňují větší kvalitu zvuku. Tento kodek není určen pro přenos hudby, nedokáže spolehlivě přenést ani DTMF tóny. Přílohy A a B G.729 definují VAD (Vocie Activity Detection), CNG (Com-

fort Noise Generator), a DTX (Discontinuous Transmission) schémata pro G.729. Rámce poslané k aktualizaci pozadí hluku jsou 15 bitů dlouhé a zasílány pouze v případě změny pozadí. Tento kodek je opět zatížen patenty (20-ti), které vlastní společnost SiproLabs.

CS-ACELP kodér pracuje s rámci řeči o délce 10 ms, což při vzorkovací frekvenci 8000 Hz odpovídá 80 vzorkům za sekundu. Každý 10 ms je rámec řečového signálu analyzován pro získání parametrů CELP modelu. Tyto parametry jsou kódovány a přenášeny komunikačním kanálem. V dekodéru jsou tyto parametry použity k obnovení excitačního signálu a koeficientů syntetického filtru. Řečový signál je pak rekonstruován filtrací excitačního vektoru přes syntetický filtr. Kvalita řečového signálu je pak ještě zvýšena postfiltrem.

Celkové zpoždění nutné pro výpočet algoritmu je 15 ms, další přídavná zpoždění mohou vzniknout během přenosu v komunikačním kanálu nebo při multipexování dat. Velkou nevýhodou je zde (stejně jako u G.728 LD-CELP) velký vypočetní výkon potřebný k prohledání celé kódové knihy a nalezení nejlepšího exitačního vektoru.

G.729A je zpětně kompatibilní s G.729. Jeho výhodou jsou menší nároky na výpočetní výkon.

2.1.3.10 iLBC (Internet Low Bit Rate Codec) Jedná se o otevřený úzkopásmový zvukový kodek využívající lineární predikci specifikovaný v RFC 3951. Vyvinula jej v roce 2004 firma Global IP Solutions a je určený pro real-time aplikace jako je VoIP a videokonference. Rozdíl oproti starším standardům založených na PCM je, že pro lineární predikci používá velmi kvalitní blokově nezávislý algoritmus. Vzorkovací frekvence je 8 kHz/16 bit (160 vzorků pro 20 ms rámce, 240 vzorků pro 30 ms rámce). Má fixní bitovou rychlost 15,2 kbps pro 20 ms rámce a 13,33 kbps pro 30ms rámce a fixní velikost rámce (304 bit/20 ms blok rámce, 400 bit/30 ms blok rámce). Systémové nároky na výpočetní výkon jsou srovnatelné s G.729A, ale oproti němu má vyšší odolnost vůči ztrátě paketů.

Mezi aplikace, které tento kodek využívají patří Google Talk, Yahoo! Messenger, Polycom IP Phone, Gizmo5, QuteCom, Maemo Recorder a další.

2.1.3.11 Speex Je otevřený zvukový kodek pro kompresi zvuku, který není zatížený patenty ani licencemi. Pro kompresi využívá CELP algoritmus. Používá šířku pásma od 2 - 44 kbps. Vzorkovací frekvence, které lze využít jsou 8,16 a 32 kHz. Jeho specifikace umožňuje dosahovat velmi dobrých kompresních poměrů při zachování dobré srozumitelnosti. Narozdíl od ostatních podobných kodeků není primárně určen ke kompresi telefonních hovorů, ale pro VoIP.

2.1.4 Paketizace - signalizační protokoly

2.1.4.1 H.323 V roce 1996 byl vydán organizací ITU-T, studijní skupinou 16 (SG-16) standard H.323, který byl původně určený pro podporu videokonferencí v rámci lokálních sítí (LAN). Nicméně brzy se začal používat i pro přenos hlasu v síti Internet, čemuž se začaly přizpůsobovat pozdější revize tohoto standardu (poslední je verze H.323v7 z roku 2009). Hlavním úkolem standardu bylo zajistit kompatibilitu mezi různými druhy

Coder	Type	Rate [kbps]	Packetization period [ms]	Frame size [ms]	Algorithmic delay [ms]	Codec delay [ms]
G.711	PCM	64	20	0,125	0	0,125
G.723.1	MP-MLQ	5,33	30	30	7,5	37,5
G.723.1	ACELP	6,4	30	30	7,5	37,5
G.726	ADPCM	32	20	10	0	10
G.728	LD-CELP	16	30	0,625	0	0,625
G.729A	CS-ACELP	8	20	10	5	15

Tabulka 2: Srovnání nejpoužívanějších kodeků používaných v IP telefonii. Typ kódování, rychlost, typická paketizace, rámec a zpoždění. Převzato z [8].

zařízení sloužících k přenosu audia a videa. Jedná se o protokol pro přenos audia, videa a dat v reálném čase přes paketově přepojované sítě (sítě obsahující IP - Internet Protokol), bez garance kvality služby (QoS) vynikající svou robustností, skládající se z několika podstandardů. Obsahuje H.225 (Q.931, RAS), H.245 (popis multimediální relace), RTP,RTCP (přenos hovoru). Ke svému provozu otevíra několik portů TCP (Transmission Control Protocol) i UDP (User Datagram Protocol). K sestavení spojení je potřeba výměny až 8 zpráv. Lze využít v jakýchkoliv topologiích, počínaje point-to-point přes sběrnicové až po hvězdicové topologie. Protokol může být použit pro multicastovou multimediální komunikaci, což znamená přenos od jednoho zdroje k více příjemcům.

V roce 1997 vyvinula americká firma Level 3 první komerční softswitch (softwarová ústředna) na standardu H.323.

2.1.4.2 SIP V roce 1999 vydala IETF (Internet Engeneering Task Force) doporučení RFC 2543, popisující první verzi protokolu SIP. Byl vyvíjen již od roku 1996 pracovní skupinou MMUSIC (Multiparty Multimedia Session Control) v rámci IETF. SIP (Session Initiation Protocol) je (signalizační) protokol určený pro sestavení, modifikaci a ukončování multimediálních relací v IP sítích. Nejčastěji je využíván pro audio. Nejvíce se používá v kombinaci s protokoly RTP (Real-Time Transport Protocol) pro přenos vlastního obsahu a SDP (Session Description Protocol - RFC-2327), pro popis přenášeného obsahu. Typ přenášených dat (audio, video) protokol SIP přímo nedefinuje, ale využívá se k tomu již zmiňovaný SDP. SIP je textově orientovaný a vychází z osvědčených protokolů HTTP a SMTP, které jsou nejpoužívanějšími protokoly na Internetu. Protokolu HTTP je velmi podobný. Díky této volbě při návrhu je protokolu SIP zajištěna nadčasovost a robustnost. Primárně komunikuje na UDP/5060 (zabezpečený: UDP/5061) protokolu, ale pro komunikaci může použít i TCP. SIP je end-to-end (P2P) orientovaný protokol tzn., že veškerá komunikační logika je uložena v samotných koncových zařízeních, které znají i jednotlivé stavy komunikace. Díky této architekruře je zvýšená odolnost komunikace proti chybám.

SIP entity jsou identifikovány za použití SIP URI (Uniform Resource Identifier), které se skládají z username, password (volitelně), host (doména) a z parametrů hlavičky (vo-

litelně), které se uvádějí za znak ? (Výpis.1).

```
sip:user:password@host:port;uri-parameters?headers
```

Výpis 1: Syntaxe SIP URI. Převzato z [8].

Architektura SIP protokolu se skládá se z účastnické stanice (UA - User Agent), což jsou zařízení, které implemnetují SIP protokol a jsou používaná především pro uskutečnění a příjem hovoru, umístěná na konci datové sítě a ze zástupného serveru (SIP Proxy Server), který zodpovídá za příjem požadavků od učastnických stanic a ostatních SIP proxy, přes které směruje hovor k cílové stanici přes SIP proxy, u které je cílová stanice registrovaná.

Učastnické stanice (UA) zároveň můžou vystupovat v roli klienta (UAC) a serveru (UAS), které jsou implementovány zárověň.

Více se věnuji protokolu SIP v kapitole Realizace výměny informací pomocí vloženého pole v SIP signalizaci.

2.1.4.3 IAX, IAX2 V rámci projektu Asterisk byly také vyvinuty protokoly IAX (Inter-Asterisk eXchange) a IAX2 vyvinuté firmou Digium Inc. Jako primarní účel vzniku těchto protokolů byla komunikace s ostatními Asterisk servery. Tyto protokoly jsou otevřené, a nejsou limitovány pouze na Asterisk, díky čemuž jsou podporovány mnoha open source telekomunikačními projekty i výrobci hardware. Protokoly dodnes nejsou součástí standardu IETF. IAX a IAX2 jsou transportní protokoly, které využívají jeden UDP port 4569 jak pro signalizaci, tak pro data. Jejich výjímečnou vlastností je seskupování několika relací do jednoho datového streamu, což je velmi efektivní vzhledem k využití šířky pásma, v případě posílání mnoha simultánních kanálů. Tato schopnost se nazývá trunking, umožňuje více různým streamům být reprezentovány jednou datagramovou hlavičkou (multiplexace), což vede k menším nárokům na řežiji spojenou s jednotlivými kanály. Důsledkem toho se snižuje latence a výrazně redukují nároky na šířku pásma.

Zabezpečení IAX protokolu je možné třemi způsoby: plain text, MD5 hashování, RSA výměna klíčů. Dále umožňuje použít šifrovaný provoz využitím dynamické výměny klíčů během sestavování spojení (call setup), nastavením AES128.

Protokol IAX2 byl navržen za účelem spolupráce se zařízeními využívající NAT (Native Address Translation) a snadnějšímu překonání firewallu. Díky využití jednoho UDP portu, jak pro signalizaci tak pro data, se snižuje počet potenciálních průniku přes firewall na minimum. Z hlediska implementace se jedná o jeden z nejjednodušších protokolů.

2.1.4.4 IMS (Internet Multimedia Subsystem) Je globální architektura založena na protokolu IP, která je nezávislá na přístupové síti, určená k poskytování různých druhů multimediálních služeb koncovým uživatelům. Vzhledem k dnešnímu trendu konvergence sítí, a tím sjednocování služeb, vznikla potřeba architektury, která by všechny poskytované služby zastřešovala. Příkladem můžou být dnešní smartphony, které nám dokážou poskytnout veškeré komunikační a mutlimediální služby.

IMS byl definován v rámci standardizační organizace 3GPP (3rd Generation Partnership Project) v roce 1999 jako součást rozvoje technologií pro mobilní sítě. Původně sloužil pro podporu multimediálních služeb v rámci GPRS (General Packet Radio Service). Postupem času vycházely nové verze (release), které přidávaly podporu současných i budoucích technologií jako např. CDMA2000 (Code Division Multiple Access 2000), Cablelabs (sdružení operátorů sítí kabelových televizí), EDGE (Enhanced Data for GSM Evolution), UMTS (Universal Mobile Telecommunications System), LTE (Long Term Evolution), NGN (Next Generation Network) - konvergence pevných a mobilních služeb.

Jako jeden ze základních protokolů byl zvolen SIP, obsahující jednoduché textové zprávy, které zjednodušují vývoj, ladění kódu a jeho flexibility z hlediska rozšiřitelnosti, optimalizace hlaviček a vytváření různých topologií. SIP implementace v rámci IMS má od čistého SIP následující odlišnosti:

- optimalizace pro bezdrátové sítě - komprese SIP zpráv (SigComp), implicitní registrace více identit,
- nové autentizační mechanismy - mechanismus AKA (Authentication and Key, Agreement), ISIM (IP Multimedia Services Identity Module) a USIM (Universal Subscriber Identity Module),
- accounting - Online (předplacené služby) / Offline (dodatečně placené služby),
- policies (politiky) - aplikované na data.

2.1.4.5 MGCP (Media Gateway Control Protocol) Je protokol, který se liší od předchozích signalizačních protokolů. MGCP neslouží pro signalizaci mezi koncovými zařízeními např. IP telefony, ale zajišťuje signalizaci mezi media gateway (MG) v prostoru IP sítí a veřejných PSTN (Public Switched Telephone Network) sítí. V roce 1999 vydala IETF (Internet Engeneering Task Force) doporučení RFC 2705, popisující první verzi protokolu. V RFC 2805 byla popsána architektura a programovací rozhraní protokolu. Aktuální specifikace protokolu je nyní popsána v dokumentu RFC 3435, který nahradil starší RFC 2705. Stejně jako protokol SIP využívá protokol SDP (Session Descritption Protocol) k vyjednání o podrobnostech přenášeného obsahu (protokoly, kodeky, čísla portů) a RTP protokol pro přenos média streamů.

2.1.4.6 SCCP / SKINNY (Skinny Client Control Protocol) Je proprietární ovládací terminál Cisco a signalizační protokol typu klient-server. Původně jej vyvinula společnost Selsius Systems Inc. založená v roce 1997, kterou následně v roce 1998 získala společnost Cisco Systems Inc.. Tento protokol využívá Cisco Unified Comunications Solution. Obecně platí, že SCCP se používá k zajištění kontrolního kanálu mezi Cisco Unified IP telefony a CUCM - Cisco Unified Communitations Manager (dříve Cisco Call Manager). Cisco Unified Communitations Manager představuje správce komunikace nebo IP PBX (Private Branch eXchange - pobočková ústředna) pro IP telefonii a VoIP řešení od společnosti Cisco Systems. V defaultním nastavení SCCP používá TCP protokol a port 2000. Jedná se o binární protokol.

2.1.4.7 HFA (Hipath Feature Access) Je vlastním (proprietárním) protokolem firmy Siemens. Podstatou tohoto protokolu je tunelování pokročilých služeb telefonních systémů řady HiPath do protokolu IP a jejich přenos datovou sítí. Hlavní důvody, které vedly k zavedení HFA protokolu, byly velmi omezené možnosti telefonních služeb poskytovaných prvními verzemi mezinárodně standardizovaných VoIP protokolů (např. H.323) a nutnost zajištění kompatibility klasických a VoIP částí systému HiPath.

2.1.4.8 Skype Skype je software vyvinutý firmou Skype Technologies S.A (nyní patřící společnosti Microsoft), který umožňuje provozovat VoIP, videohovory a instant messaging. Velkou výhodou je bezproblémová průchodnost přes NAT (Native Address Translation) a Firewall. Protokol není veřejně dostupný. Pracuje na principu peer-to-peer (P2P). Po aplikaci deduktivní analýzy se zjistilo, že Skype využívá hybridní řešení. Architektura obsahuje Login servery, které uchovávájí uživatelská data (loginy, kontakty), dále SuperNodes (ústředny), což jsou uzly, které běží v privilegovaném módu (disponují rychlým připojením, a veřejnou IP adresou), a které propůjčují své systémové zdroje k uskutečnení hovoru (komunikaci) mezi ostatními uživateli. Node (uzel) je bežný koncový uživatel, který je schovaný za NAT nebo Firewallem a není nijak využíván ostatními uživateli. Může mít i veřejnou IP adresu. Hovor je pak spojován po P2P síti nejkratší cestou. Pokud nemá žadný z Node (uzlů) veřejnou IP adresu, je spojení vedeno přes SuperNode. V případě, že jeden z Node tuto veřejnou IP adresu má, je spojení vedeno přímo.

2.1.4.9 Jingle (Google Talk) Jingle je rozšíření pro XMPP (eXtensible Messaging and Presence Protocol), které přidává princip P2P signalizace pro multimediální služby jako je VoIP nebo videokonference. Byl navržen společností Gooole a XMPP Standards Fundations. Multimediální streamy jsou dodávány pomocí RTP. Podpora NAT je zajištěna za použití ICE (Interactive Connectivity Establishment). Má širokou podporu platforem a aplikací pro VoIP komunikaci jako např. Asterisk, iChat, Google Talk, Yate, QIP, FreeSWITCH a další.

2.1.5 Přenos dat - transportní (média) protokoly

Zatímco signalizační protokoly zajišťují informace nezbytné pro sestavení hovoru, transportní (média) protokoly zajišťují samotný přenos hlasu, videa či jiného multimediálního obsahu přes IP sítě. Předtím než se analogový audio záznam začne přenášet přes datovou síť jako paket IP, musí se nejdříve digitalizovat a paketizovat. V rámci Internetu se nám nabízí transportní spojově orientovaný protokol TCP (Transport Control protokol), který zajišťuje spolehlivou službu s potvrzením doručených dat, ale pro real-time přenos není vhodný, z důvodů pozbývání platnosti přenášených dat časem.

Jako další lze využít vhodnější UDP (User Datagram Protocol), který se řadí do kategorie nespojově orientovaných transportních protokolů, tudíž neobsahuje potvrzení o doručených datech a také nezaručuje jejich doručení ve správném pořadí, což může způsobovat problémy u některých aplikací. Do IP záhlaví přidává pole: zdrojový a cílový port služby, délka přenášených dat (včetně záhlaví) a kontrolní součet pseudozáhlaví.

Vzhledem k tomu že i UDP protokol má své nedostatky vznikl protokol RTP, který je umístěn nad UDP protokolem.

2.1.5.1 RTP (Real-time Transport Protocol) Je protokol, který je umístěn na aplikační vrstvě modelu ISO/OSI. Byl vytvořen pro přenos multimediálích dat ve formě paketů v reálném čase (audio, video). RTP vyvinula AVT (Audio-Video Transport Working Group) v rámci IEFT (Internet Engineering Task Force) v roce 1996, kdy byl poprvé publikován jako standard RFC 1889, později (v roce 2003) nahrazený RFC 3550 a poslední verzí je standard RFC 3711 (SRTP) z roku 2004.

Je postaven na protokolu UDP, a navíc obsahuje nové vlastnosti pro zajištění lepšího přenosu multimediálních dat. Zajišťuje identifikaci začátku a konce rámce, rekonstrukci správného pořadí paketů na základě sekvenčních čísel (sequence number), jejich časové značkování (timestamp - reprezentuje vzorkovací značka prvního oktetu v paketu), na jejíchž základě je určen správný okamžik přehrávání dat (synchronizace), dále zajišťuje multiplexování a demultiplexování. Podporuje přenos mezi dvěma i více účastníky. Hlavička má velikost obvykle 12B. RTP nezajišťuje rezervaci kanálu a negarantuje QoS (Quality of Service).

Více se věnuji protokolu RTP věnovat v části: Využití RTP pro přenos textových informací(4.).

2.1.5.2 RTCP (Real-time Transport Control Protocol) Je řídící protokol, který doplňuje protokol RTP o poskytování zpětné vazby určující kvalitu služeb (QoS) poskytovanou RTP. Je definovanán v RFC 3550, které nahradilo RFC 1889. Nepřenáší žádná multimediální data, ale pouze řídící pakety k těmto relacím. RTCP shromažďuje statistiky přenosu:

- počet odeslaných Byte,
- počet odeslaných paketů,
- počet ztracených paketů,
- jitter (rozptyl zpoždění),
- feedback (zpětnou vazbu),
- latence (doba odezvy).

Více se věnuji protokolu RTCP věnovat v části: Využití RTP pro přenos textových informací(4.).

2.1.6 QoS (Quality of Service)

Jelikož je IP telefonie založena na technologii přepojování paketů, kde všechny pakety mají stejnou prioritu při průchodu sítí, může docházet k přehlcení datové linky, kdy

linka již nedokáže pokrýt množství datového provozu. Toto je problém především pro realtimové služby jako je VoIP, videokonference a online hry.

Z toho důvodu byla zavedena technologie, která se nazývá kvalita služby (QoS - Quality of Service), která kontroluje a zajišťuje, aby kvalita provozu odpovídala dané službě. Cílem QoS je zajistit určitou garanci kvality dané služby. Pomocí QoS můžeme nastavit minimální a maximální přenosové pásmo pro určitý typ dat, přidělovat jim různou prioritu, rozdělovat je do kategorii apod. Mezi hlavní problémy VoIP, které dokáže QoS úspěšně eliminovat či významně snížit jejich hodnoty patří:

- jitter (rozptyl zpoždění) - kdy dochází ke kolísání latence vlivem různého vytížení sítě v čase. K eliminaci tohoto efektu se používá tzv. jitter buffer, který je schopen vyrovnat rozdíly v latenci,
- latence (doba odezvy) - je celkové zpoždění dat, při přenosu hovoru po datové síti. Do tohoto zpoždení se započítává i latence kodeku, latence samotného algoritmu a všechny aspekty, které způsobují zpoždění v čase. Kvalitní hovor vyžaduje latenci co nejnižší (nejlépe do 100ms), pokud je latence vyšší dochází k rozpoznatelnému zpoždění a neplynulosti hovoru,
- ztráta paketů - představuje četnost zahazování paketů zařízeními v síti nebo příjímačem. Toto zahazování může způsobovat řada věcí např. přetížení sítě, velký provoz na routerech apod..

Mechanismy QoS pracují s určitou alokovanou šířkou pásma, kterou sice nejsou schopné v případě nutnosti zvětšit, ale umí s ní efektivně nakládat, aby byla kvalita služeb v maximální možné kvalitě.

Prinip Best-effort zajišťuje, že je s daty nakládáno tak, aby byla zachována co možná nejlepší snaha o doručení, avšak bez jakýchkoliv mechanismů garantujících spolehlivost komunikace, v závislosti na aktuálním vytížení sítě.

Metoda DiffServ (Differentiated services) je jednou z nejvyužívanějších implementací pro QoS. Hlavním atributem určujícím prioritu je hodnota DSCP (Differentiated Services Code Point), která se nachází v hlavičce IP paketu v poli ToS (Type of Service). DSCP má 6b hodnotu, z toho 3b s nejvyšším významem reprezentují položku IPP (IP precedence). Čím je hodnota atributu vyšší, tím se zvyšuje priorita paketu. Dále určuje prioritu provozu hodnota atributu CoS (Class of Service), která je umístěna v hlavičce ethernetového rámce. CoS může nabývat hodnoty 0-7, kdy 0 představuje nejnižší prioritu.

Metoda IntServ (Integrated services) pracuje na principu rezervace pásma. Rámec IntServ se skládá ze čtyř částí: plánovač paketů, kontrola přístupu, klasifikátor a rezervační protokol RSVP (resource ReSerVation Protocol). Rezervování síťových prvků je iniciováno aplikací a rezervaci potvrzují všechny síťové prvky na trase mezi zdrojem a cílem určení dat.

2.1.7 Zabezpečení

Postupem času, jak se IP telefonie stává stále popularnější mezi běžnými uživateli, stává se předmětem zájmu hackerů a jiných útočníků. V důsledku toho vznikla potřeba odolá-

vat nežádoucím útokům jejich eliminací, či snížení bezpečnostních rizik v rámci IP telefonie. Jako každá jiná technologie není ani VoIP naprosto bezpečná, jelikož je používána v tak různorodém prostředí jako je Internet. Do jisté míry je to způsobené oblibou SIP protokolu ve VoIP, avšak tento protokol není šifrovaný. Pro zvýšení bezpečnosti, alespoň na základní úroveň, je třeba dodržovat bezpečnostní pravidla sítě, operačního systému a aplikačního software. Další prvky, které dokážou zvýšit bezpečnost jsou firewally s podporou internetové telefonie, šifrování telefonního hovoru a případně použít další bezpečnostní mechanismy. Dále je nutné mít na paměti, že není vhodné šířit citlivé osobní údaje, jako jsou čísla platebních karet, sdělovaní loginů a hesel apod.

Mezi hlavní techniky útoku, které se aplikují na VoIP patří: odposlouchávání, modifikace dat (Man-in-the-middle), DoS, DDoS, Vishing, SPIT.

2.1.7.1 DoS / DDos - Odepření služby DoS útok (Denial of Service) je nejrozšířenější útok tohoto typu. Při útoku dochází ke kompletní nedostupnosti VoIP služeb. Účelem DoS útoku je znepřístupnit službu, či omezit její funkčnost, zahlcením serveru požadavky, které není schopen v rámci svého výpočetního výkonu obsloužit. V praxi pak tento útok degraduje kvalitu daného telefonního hovoru.

DDos(Distributed Denial of service) - jedná se distribuovanou o variantu DoS, kdy se nejčastěji, díky počítačům infikovaných škodlivým kódem, vytvoří tzv. botnet. Takový botnet je pak útočník schopen řídit z jednoho místa. Ve chvíli, kdy je zahájen útok, všechny stanice v botnetu začínají zasílat požadavky na vybraný server, kterým chce útočník zamezit poskytování standardních služeb běžících na serveru.

2.1.7.2 Krádež identity Jedná se o metodu, kdy se útočník zmocní přihlašovacích údajů změnou signalizace při navazování spojení, krádeží přímo koncového zařízení či prolomením registrace. Útočník je schopen prolomit registraci zahlcením systému podvrženými žádostmi a dotazy, na které systém „zmateně“ odesílá svoje odpovědi, díky čemuž si útočník skenuje strukturu sítě (IP adresy koncových zařízení, ústředny, atd.).

2.1.7.3 Vishing (Voice phishing) Je kriminální praktika, která využívá sociálního inženýrství aplikovaného na IP telefonii, za účelem získání přístupu k soukromým osobním a bankovním informacím. Vychází z klasického phishingu. Jedná se o falešné telefonáty, kdy je uživatel vyzván, aby si změnil své přístupové údaje k dané službě, a poté zavolal na tzv. zákaznickou linku a potvrdil své údaje. Hovor je ovšem presměrován a citlivá osobní data jsou zaznamenána. To je zapříčiněno i díky popularitě protokolu SIP a jeho absenci šifrování.

2.1.7.4 VoIP Spam / SPIT (SPam over Internet Telephony) Jsou hromadné nevyžádané automatické volání, nahrané pro telefonní hovory, uskutečněné pomocí VoIP. Stejně jako jiné služby Internetu se dají zneužít, pokud má dotyčná strana zlé úmysly. Používá se pro doručování reklamy (telemarketing), žertovné hovory apod. Tento druh obtěžování není zatím tolik rozšířen, nicméně s roustoucí oblibou VoIP se bude vyskytovat stále častěji.

2.2 Steganografie

Steganografie (steganography) je vědní obor zabývající se utajením komunikace prostřednictvím ukrytí zprávy. Zpráva je ukryta tak, aby si pozorovatel neuvědomil, že komunikace vůbec probíhá. Síla této komunikace je závislá na schopnosti utajení zprávy (jedná se o takzvanou bezpečnost skrze utajení – security through obscurity), a proto zachycení skryté zprávy prakticky znamená její prozrazení. Aby ani v tom případě nedošlo k prozrazení obsahu zprávy, kombinuje se zpravidla s dalšími metodami šifrování.

Termíny steganografie a šífrování pocházejí ze starověkých řeckých slov Steganos, což znamená chráněné (kryté) a Kryptos má význam skrýtý (tajný).

Rozdil mezi kryptografií a steganografií spočívá v rozdílné aplikaci metod. Kryptograficé metody využívají aplikace šifrovacích algoritmů na přenášené informace. To má za následek, že jsou informace pro neoprávněné osoby nečitelné, avšak uživatel ví, že tato „tajná“ komunikace probíhá. Narozdíl od toho steganografie aplikuje metody, které se zaměřují na to, aby byla veškerá tajná komunikace utajená pro neoprávněné osoby.

V současné době se steganografie považuje za disciplínu informatiky či matematiky, ovšem v historii tomu tak nebylo.

2.2.1 Historie

První zdokumentovaný případ steganografie pochází z 5.století před našim letopočtem, kdy Řek Demaratus, žijící v Sůsách, poslal varování o perských přípravách na invazi do Řecka vyryté do voskové psací tabulky, z níž nejprve seškrábal vosk a po vyškrábání zprávy do dřevěného podkladu ji voskem opětovně zakryl. Destička se jevila navenek jako klasické zboží, které mohlo být bezpecně dopraveno do Řecka.

Dalším případem využití steganografie bylo zaslání tajné zprávy v břiše zajíce, médským generálem Harpagusem. Zpráva byla zaslána do Persie, kterou aktuálně založil král Kýros II. To zapříčinilo pořážku Řecka Perskou říší.

V jiném případě, který se odehrál jen několik desítek let po této události, chtěli Řekové přesvědčit milétského vůdce Aristagora, aby se připojil k povstaní proti Persii. Zpráva byla poslána způsobem, kdy byly otrokovi oholeny vlasy, a na jeho lebku byla vytetována zpráva. Ve chvíli, kdy otrokovi dorostly vlasy, byl vyslán s poselstvým do Milétu. Tato technika se ovšem neuplatnila v širším měřítku, protože příprava otroka pro přenos zprávy trvala řádově měsíce. Dalším aspektem bylo omezené množství otroků, kteří měli nepotetovanou lebku.

V Číně se využívala steganografická technika, při které byla zpráva zapsána na hedvábný papír malých rozměrů, který byl složen a zalit do vosku, který sloužil jako obal. Posel poté tento zakonzervovaný papír spolkl a vydal se na cestu k příjemci. Jak byla zpráva získána ze zažívacího traktu zdroje neuvádí.

V období prvního století našeho letopočtu, filosof Pilinius využil a aplikoval látku připomínající dnešní neviditelný inkoust. Jednalo se o mléčnou esenci rostliny thithymallus. Z tenké vrstvy látky nanesené na papír se v krátkém čase vyprařila pigmetová složka, která způsobila zdánlivé zmizení textu. Pro přečtení textu stačilo papír dostatečně zahřát, čímž uhlík přítomný v látce zuhelnatěl a text se objevil v hnedé barvě.

V časech raného novověku se používala steganografická technika tzv. Cardanova mřížka, která nese pojmenovaní po Gerolamu Cardanovi. Jednalo se o destičku pevného materiálu obsahující nepravidelně umístěné pravoúhlé otvory. Po přiložení papíru na destičku byla do jednotlivých otvorů vepsána jednotlivá šifrovaná (tajná) písmena. Do zbylých volných pozic se doplnil další nosný text. Doplněný text je závislý na tajných písmenech, což může od jisté míry budit podezření. Tato technika je vpodstatě nejstarší kryptografická transpoziční šifra.

V Americe se během občanské války využívala technika neviditelného inkoustu. Neviditelný inkoust je technika, která využívala mléčné šťávy pampelišky nebo citronové šťávy, které se zviditelňovaly po zahřátí. Později se vytvářely inkousty ze složitějších syntetických sloučenin, viditelných jen pod speciálním světelným zdrojem, nebo po přetření jinou chemikálií.

V oblasti moderních dějin přispělo k rozvoji steganografických technik období světových válek. V období první světové války se využívaly techniky používání synonym z určité terminologické oblasti pro označování pojmů z jiné oblasti. Například zprávy obsahovali text z oblasti chemie či přírody, které ve skutečnosti popisovaly geografické oblasti, počty válečných jednotek a další taktické informace.

Ve druhé světové válce vynalezli nacisté technologii tzv. microdots. Jednalo se o technologii miniaturizace textu či grafiky do prostoru velikosti tečky psané na klasickém psacím stroji. Do tohoto miniaturního bodu bylo možné vložit celou stránku formátu A4 (zmenšení až 1:200). Miniaturizovaná informace byla vytištěna na materiálu podobném filmu a k jejímu přečtení postačil malý mikroskop. Odesilatel nalepil tyto bodové nosiče tajných zpráv přes interpunkční znaménka v dopisech, jejichž obsah byl na první pohled zcela nevinný. Protože se filmový materiál oproti matnému papíru výrazně leskl, byly používány různé dodatečné kamuflážní metody, např. samoúčelné gumování prostoru s přelepenou tečkou. Přesto časem právě tento rozdíl v lesklosti materiálu spojencům existenci microdots prozradil.

2.2.2 Současnost

V současném informačním věku a s rozmachem výpočetní techniky je steganografie používaná zejména v oblasti digitalních dat a Internetu. Nutností v této oblasti je, aby byla tajná zpráva snadno přenositelná a prezentovatelná v heterogenním prostředí digitálních dat. Většinou se kombinuje s kryptovacími algoritmy. Tajné zprávy jsou přenášeny v různých typech dat(audio, video, obrázek, text, metadata komunikačních protokolů.)

2.2.3 Princip

Principy moderní steganografie využívají k přenosu tajných zpráv jiné datové soubory, které kamuflují daný přenos či skrývají přenášní dat. Data která slouží ke kamuflaci se nazývají nosič (cover medium). Jako nosič můžou sloužit následující prostředky:

- textové soubory,
- spustitelné soubory (executable files),

Obrázek 2: Steganogram

- binární soubory (binary files),
- obrazové soubory,
- audio soubory,
- video soubory,
- komunikační (síťové) protokoly,
- diskové oddíly, souborové systémy (file system),
- operační systémy.

Výstupem dané steganografické techniky je tzv. steganogram.

Pro skrytí informací musí být k nošiči přidána redundantní data, nebo musí být část dat nosiče modikovaných, v přijatelné míře, aby nedošlo ke znehodnocení informace, kterou nese nosič nebo k detekovatelným změnám vedoucích k odhalení přenosu tajné informace. Výhodou redundantního přídaní tajných dat je, že nijak neovlivňuje ani nemodifikuje původní informace nosiče. Nevýhodou je poměrně snadná detekce při analýze steganogramu.

Všeobecně se považují za vhodné nosiče datové segmenty s proměnnou velikostí, jejich vnitřní struktura je částečně modifikovatelná a jejich interpretace záleží na vnějším subjektu.

Maximální kapacita, která udává, kolik tajných informací je daný nosič schopen pojmout se nazývá steganografická kapacita (nosiče).

V současné době se k vetší bezpečnosti resp. k menší pravděpodobnosti odhalení tajných informací využívají spolu se steganografickými technikami i techniky kryptografické.

2.2.4 Rozdělení steganografie

2.2.4.1 Injekční steganografie (Injection steganography) Spočívá v tom, že přidává tajné data do nosiče na vhodné místo tak, aby se nenarušil původní obsah nosiče. K tomu se využívají injekční algoritmy. Příkladem takové techniky utajení může být např. do skriptu interpretovaného jazyka přidán komentář s tajnou informací (textem).

Výhodou injekčních algoritmů je malá nebo žádná náročnost na výpočetní výkon - počet bitů infomace, kterou chceme skrýt společně s tajnými daty. Naopak nevýhodou

injekčních algoritmů je modifikace velikosti souboru. To vede ke zvýšení objemu dat nosiče. To může vzbudit podezření u uživatelů v případě, že dva soubory s totožným obsahem mají rozdílnou datovou velikost nebo soubor jehož datová velikost je neadekvátní vzhledem k povaze dat (textový soubor).

Injekční techniky jsou aplikovány u mnoha počítačových virů např. trojský kůň.

2.2.4.2 Substituční steganografie (Substitution steganography) Je založená na nahrazení části původních dat nosiče tajnými daty, přičemž je nutné, aby nedošlo ke zřetelné modifikaci nosiče. Substituční algoritmy využívají nedokonalosti lidských smyslů zejména sluchu a zraku (více: v části Omezení vnímání člověka). Do této kategorie patří technika LSB (Least Significant Bit).

2.2.4.3 Propagační steganografie (Propagation steganography) Se zásadně liší od injekčního a substitučního algoritmu tím, že nevyužívá nosič. U propagačního (generujícího) algoritmu figuruje jediný vstup a tím jsou tajné informace. Jejím výstupem je steganogram syntetizovaný stegosystémem.

Výhodou propagačního algoritmu je eliminace omezení, které vznikají při použití nosiče a nemožnost porovnání výsledného steganogramu.

Na principu propagačního algoritmu jsou generovány např. fraktály.

2.2.5 Omezené vnímání člověka

Je dáno fyzikálními vlastnostmi orgánu zpracovávající informace z okolního prostředí a následnnou interpretací a zpracováním mozku, při kterém dochází k částečné modifikaci reality, na základě předchozích vjemů a následné domýšlivosti mozku.

V rámci algoritmů, které jsou založeny na principu ztrátové komprese dochází právě na základě omezení lidského vnímání k odstranění redundantních nebo méně významných informací. Tohoto aspektu se využívá ve steganografii k nahrazení těchto informací informacemi tajnými.

2.2.5.1 Sluch Bylo zjištěno, že relativně tichý zvuk, který bezprostředně následuje po relativně hlastitějším zvuku, je pro lidský sluch neslyšitelný. Lidský sluch nedokáže zaznamenat krátké ozvěny (do 20 ms). V rámci této vlastnosti lidského sluchu lze využít pro zakódování utajené informace do podoby krátké (binární 0) a dlouhé (binární 1) ozvěny [29]. Udávaný frekvenční rozsah, které je lidské ucho schopné zpracovat je 20 Hz - 20 kHz. Tento frekvenční rozsah se v průběhu života mění. S narůstajícím věkem se horní hranice slyšitelného spektra snižuje. To má za následek omezené vnímání vysokých tónů. Této vlastnosti lze využít pro umístění tajné informace tak, že bude umístěna mimo hranici slyšitelného spektra.

2.2.5.2 Zrak Lidský zrak je schopen rozlišit od sto tísíc do několika desítek miliónů barev a v průběhu života se tento rozsah mění. Tento velký rozsah hodnot je dán tím, že

přesné množství barev, které je lidské oko schopné rozpoznat, není zatím exaktně zjištěno.

Lidský zrak má nedostatky v oblasti malé změny jasu v různých častech obrázku, jsou-li rozprostřeny relativně náhodně. Intenzivněji rozlišuje změnu jasu oproti změně barevného odstínu, obzvláště v různobarevné a členité struktury. Dále je mozek schopný si domýšlet (na základě předchozích zkušeností) zdánlivé informace, které v obraze vůbec nejsou tzv. optický klam. Této vlastnosti lze využít v rámci steganografie pro skrytí tajné informace. V rámci této úpravy je potřeba mít na paměti, že zrak se zaměřuje na hlavní objekty na obrázku, čímž je nežadoucí, aby byla tato oblast nosiče modifikována. Případná modifikace, by mohl vést k odhalení utajované informace.

2.2.6 Techniky

Nejčastěji používané techniky ve steganografii jsou z kategorie substitučních algoritmů. Existuje velké množství technik, které dokážou ukrýt, přenést a následně zobrazit tajnou informaci. V této části se zaměřím zejména na steganografii v oblasti síťových komunikačních sítích a audio signálu.

2.2.6.1 Network steganography (Síťová steganografie) Je velmi progresivně se rozvíjející oblast, v závislosti na rychlém vývoji komunikačních technologií. S tím souvisí také rostoucí rizika při komunikaci a přenosu prakticky jakýchkoliv informací pomocí komunikačních sítí. Jednou z nejrychleji rozrůstajících se služeb v těchto sítích je služba VoIP. Tím ovšem vzrůstá i potenciální riziko zneužití, pro přenos tajných informací pomocí staganografie. V následující části budou popsány steganografické techniky, které využívají vlastosti komunikačních protokolů jako je IP/UDP/TCP/RTP/RTCP/SIP.

Steganografická metoda	Přenosová šířka pásma skrytého kanálu
IP/UDP protokol	32b/paket
RTP protokol	16b/paket
RTCP	192b/paket
LACK	1280b/paket (použití 0.1% ze všech RTP paketů)
QIM (audio watermarking)	0.6b/paket

Tabulka 3: Steganografické metody a jejich přenosová šířka pásma. Převzato z [1].

2.2.6.1.1 IP/UDP/TCP/RTP Vzhledem k tomu, že TCP/IP jsou hlavní protokoly určené pro komunikaci v datových sítích, je tato oblast značně prozkoumána. Mezi používané metody patří zejména modifikace (polí) IP hlavičky (header). Vzhledem k tomu že jsou určitá (pole) hlavičky nevyužité nebo je můžeme modifikovat a tím využít k vytvoření skrytého kanálu. Nicméně analogický postup lze aplikovat na protokoly UDP/TCP/RTP/RTCP a na všechny protokoly založené na IP protokolu jako je např. SIP protokol.

2.2.6.1.2 LACK - Lost Audio paCKets steganography Je steganografická metoda, která využívá pro přenos steganogramu záměrného zpoždění audio paketů v rámci VoIP komunikace. Využívá poznatku, že audio pakety, které jsou zpožděny nad určitou hodnotu času, tak jsou systémem zahozeny (resp. odloženy). Totoho principu se využívá právě pro přenos skrytých dat.

2.2.6.1.3 Speech Codec SID (Silence Insertion Description) frame Kodeky řeči implementují technologie DTX (Discontinous Transmission) - přerušované vysílání, VAD (Voice Activity Detection) - detekce hlasové aktivity, CNG (Comfort Noise Generation) - generátor šumu. Tyto mechanismy jsou schopné detekovat ve vstupním signálu přitomnost hlasu. Je-li detekován hlas začne jej kódovat kodér. Pokud není hlas detekován, pošle se speciální rámec s názvem SID. Tento rámec neobsahuje konverzaci, což znamená, že se přenáší pouze malé množství dat (bitů). SID rámce nemusí být posílány periodicky, ale pouze při změně hladiny hluku v pozadí. Velikost rámce je závislá na použitém kodeku. Např. pro G.729AB to je 10bits/frame a pro G.723.1 24bits/frame. Když jsou využívány DTX/VAD/CNG, lze během intervalu ticha SID rámce použít pro přenos skrytých dat.

Ovšem přenosová šířka skrytého kanálu je poměrně nízká. Navíc aktivní správce (active warden) je schopen modifikovat některé bity v SID rámci a tím může eliminovat nebo omezit šířku pásma této metody [1].

2.2.6.1.4 HICCUPS - Hidden Communication System for Corrupted Networks (Skrytý komunikační systém pro narušené sítě) Jedná se o steganografickou techniku aplikovanou na službu VoWLAN v rámci WLAN sítí. V rámci této techniky dochází k zaměrnému vytváření špatných kontrolních součtů (checksum) rámců. Pokud nepatří stanice do skryté skupiny (je tzv. normální), jsou rámce se špatným kontrolním součtem vyřazeny, nicméně právě tyto rámce nesou steganogramy (tedy tajnou zprávu) a je tak vytvořen skrytý kanál. Tato technika umožňuje na požádání (on-demand) vytvořit dotatečnou šířku pásma pro steganografické účely.

Nevýhodou této metody je nutnost modifikace síťových karet.

2.2.6.2 Audio watermarking Primárním účelem této techniky je zabezpečit autorská práva nebo známky duševního vlastnictví tzv. DRM (Digital Right Management). Audio watermarking se dá využít i pro přenos tajných dat v reálném čase. Tato technika obsahuje řadu metod, které je možné použít k digitalnímu označení dat. Mezi nejpoužívanější metody patří LSB (Least Significant Bit), DSSS (Direct Sequence Spread Spectrum), FHSS (Frequency Hopping Spread Spectrum), Echo Hiding, QIM (Quantization Index Modulation). Dostupná šířka pásma pro skrytou komunikaci (kanál) v rámci těchto algoritmů závisí především na vzorkovací frekvenci a typu audio materiálu, který je kódován [1].

2.2.6.2.1 LSB - Least Significant Bit (Nejméně významný bit) Jedná se o substituční metodu, která využívá k ukrytí tajných dat nejméně vyznamných bitů daného nosiče. Mezi hlavní výhody této metody patří jednoduché vkládání a extrahování tajných dat (což je zároveň i nevýhodou), z tohoto důvodu není nutné mít dostupný vysoký

Audio watermarking algoritmus	Přenosová šířka pásma skrytého kanálu RBR (Skype)	Přenosová šířka pásma skrytého kanálu RBR (VoIP hovor)
LSB	1kbps/1kHz (ze vzor. fr.)	4kbps
DSSS	4bps	22.5bps
FHSS	-	20.2bps
Echo Hiding	16bps	22.3bps

Tabulka 4: Audio watermarking algoritmy a jejich experimentálně vypočítané RBR. Převzato z [1].

výpočetní výkon. Mezi další přednosti patří vysoká steganografická kapacita. Mezi nevýhody patří špatná robustnost této metody. Např. při manipulaci v rámci obrázku jako je otočení, ořezání či aplikování filtru jsou tajná data většinou znehodnocena. Tajné informace se taktéž mohou znehodnotit pokud je použit na nosič ztrátový kompresní algoritmus, který odstraňuje redundantní data, které se využívají právě pro uchování tajných informací.

LSB Random je vylepšení modifikační techniky LSB. Oproti LSB využívá LSB Random náhodně vybraných bitů nosiče, čímž se zvyšuje zabezpečení tajných informací proti detekci. K náhodnému rozložení a ukrytí tajné informace se využívá pseudonáhodný generátor čísel. Pro přečtení tajných informací obě komunikující strany používají klíč (stego-key), který představuje tzv. seed, což je počáteční hodnota pro pseudonáhodný generátor čísel. Výstupem je náhodná sekvence hodnot, které představují délku bitů zpráv [30].

Tato technika se využívá pro obrazové nosiče, audio a video nosiče.

2.2.6.2.2 Spread Spectrum Method (Metoda rozprostřeného spektra) Je metoda, která se původně používala v radiovém spojení k přenosu více nezávislých informačních signálů pomocí jednoho kanálu. Výhodu této metody je její odolnost vůči širokopásmovému a úzkopásmovému rušení. V důsledku těchto vlastností, se tato metoda začala využívat v oblasti digitalního watermarkingu. Dále je v současné době využívána v bezdrátových sítích standardu 802.11, Bluetooth a GPS [6].

Podle způsobu rozprostření ve spektru rozlišujeme metody:

- DSSS - Direct Sequence Spread Spectrum (Přímé sekvenční rozprostření spektra) - Pomocí této metody je úzkopásmový signál násoben pseudonáhodnou posloupností,
- FHSS - Frequency Hopping Spread Spectrum (Rozprostření spektra pomocí frekvenčního skákání) - Tato metoda využívá princip přeskakování mezi několika frekvencemi.

2.2.6.2.3 Echo Hiding Tato metoda vkládá do audio signálu ozvěnu (echo), s velmi krátkou dobou zpoždění od původního signálu. Délka (resp. zpoždění) ozvěny je v rozmezí 10 - 20ms a tudíž je pro lidský sluch nedetekovatelná. Výhodou této metody je robustnost a neslyšitelnost. Naopak nevýhodou je snadná detekce takto označeného signálu třetí osobou.

2.2.7 Stegoanalýza (Steganalysis)

Je disciplína, která se zabývá odhalováním skrytých dat (steganogramů) v stegosystémech. Lze zde nalézt analogii s kryptoanalýzou aplikovanou na kryptografii.

Skutečnost, že stegoanalýza je vetšinou založena na odhadu než na obecně platném matematickém prolomení vlastností stegosystému naznačuje i členění analytických postupů zavedených Peterem Waynerem [30] :

2.2.7.1 Visual or aural attack Jedná se o metodu detekování steganogramu za pomoci lidských smyslů (sluch, zrak). Ve snaze co nejvíce zvětšit steganografickou kapacitu nosiče, může docházet k velké modifikaci nosiče a tím i detekování skrýtých dat např. v audiosouboru, obrázku i v IP hlavičce.

2.2.7.2 Structural attack Je čistě strojová metoda pro detekci steganogramů. Ve vnitřní struktuře nosiče detekuje anomálie této struktury, které vetšinou reprezentují redundantní data (tzn. steganogram), která by nebyla nijak interpretována či nějak využita ve standardní aplikaci. V úvahu připadají zejmená typicky strukturované nosiče, které obsahují nějaké volitelné segmenty (které jsou standardní aplikací ignorovány) nebo segmenty bez jasně zadané sémantiky. Příkladem můžou být síťové datagramy nebo spustitelné binární soubory.

Slabým místem této metody je okamžik strojového rozhodování, zda byla detekovaná data injektovaná stegosystémem či nikoliv [7].

2.2.7.3 Statistical attack Tato metoda shromažďuje množinu vytipovaných kvantitativně vyjádřitelných charakteristik. Poté jsou získaná data porovnávána s neinjektovaným (čistym) profilem objektu. Po aplikaci metody získáme dvě číselné hodnoty: pravděpodobnost úspěchu vykonání samotného statistického testu a pravděpodobnost správnosti verdiktu (čisty objekt/steganogram) [7].

2.3 Současný stav řešení problému

Steganografií aplikovanou na oblast internetové telefonie se zabývají výzkumy provedené týmem prof. Szczypiorského z Varšavské univerzity. Jejich výzkumy popisují způsoby jakými lze vytvořit skrytý kanál nejen v SIP signalizaci, ale v rámci všech nejpoužívanějších protokolů pro internetovou telefonii (SIP, SDP, RTP, RTCP).

Ve vědeckém článku [1] jsou popisovány techniky vytvoření skrytého kanálu pomocí modifikace hlavičky RTP, RTCP protokolu a aplikace steganografické techniky LACK v

rámci RTP protokolu. Dále jsou definovány principy hodnocení skrytých kanálů, navrženy bezpečnostní mechanismy steganografických polí pro RTP / RTCP protokoly, popis (Speech Codec SID, LACK) a srovnání jednotlivých watermarkingových algoritmů (LSB, DSSS, FHSS, Echo Hiding), experimentální ohodnocení šířky přenosového pásma skrytého kanálu v rámci VoIP streamů.

Výsledkem toho výzkumu je popis dvou nových steganografických technik (RTP / RTCP a LACK). Dále bylo zjištěno, že lze v průběhu typického VoIP hovoru zaslat v rámci skrytého kanálu 1,3Mbps dat (v jednom směru).

Kromě toho, z dalšího závěru vyplývá, že nejdůležitější steganografické techniky ve VoIP komunikaci v rámci experimentu je staganografie IP / UDP / RTP protokolů, která poskytuje 96% přenosové kapacity z celkového přenosového pásma skrytého kanálu. Jiné techniky jako LACK tvoří 2,6% a audio watermarking 1,2% z celkového přenosového pásma skrytého kanálu.

Je konstatováno, že celková šířka přenosového pásma skrytého kanálu typického VoIP hovoru je vysoká.

V oblasti stegoanalýzy je konstatováno, že lze dosáhnout omezení šířky přenosového pásma skrytého kanálu jen do určité míry. Dále jsou zdůrazněné dvě věci. Za prvé, že v současné době neexistuje žádná zdokumentovaná implementace aktivního strážce (active warden), který by měřil kritická data v IP sítích, takže lze použít všechny dostupné steganografické techniky. Za druhé, analyzování každého VoIP paketu aktivním strážcem pro každou steganografickou techniku popsanou v tomto výzkumu, může potenciálně vést ke ztrátě kvality hovoru v důsledku zvýšeného zpoždění.

Konečným závěrem je konstatování, že skrytý kanál v rámci VoIP hovoru je potenciální hrozbou pro bezpečnost sítí a množství informací, které je možné přenést pomocí skrytého kanálu je významné.

Vědecký článek [2] popisuje steganografickou techniku v signalizační fázi hovoru a to modifikaci SIP / SDP hlavičky. Také je zde popsán princip odhadnutí možství přenesených dat v rámci skrytého kanálu. Dále připomíná jaké jsou možnosti steganografie v oblasti IP / TCP / UDP. Steganografická kapacita IP je poměrně velká (může být větší než 32 b/paket). Tento důležitý poznatek je podstatný zejména z toho důvodu, že výše zmíněné protokoly jsou přítomny v každém VoIP paketu (bez ohledu na to, zda se jedná o zprávu signalizační, audio paket nebo zprávu kontrolní).

Výsledkem tohoto výzkumu je poznatek, že lze pomocí modifikace SIP / SDP přenést v rámci skrytého kanálu 2000 bitů dat v jednom směru během jednoho hovoru. I když toto množsví informací může být považováno za nízké, může i tak způsobovat vážné informační úniky.

Ve vědeckém článku [5] popisuje tým docenta Vozňáka schopnost algoritmu AVG (klouzavý průměr) a NAIVE detekovat anomálii v rámci internetové telefonie. Bylo zjištěno, že při použití algoritmu NAIVE, lze navíc vložit maximálně 1 SIP zprávu za sekundu, bez vyvolání výstrahy. V případě algoritmu AVG, lze navíc vložit maximálně 4 zprávy za sekundu. Dále jsou definována pravidla pro detekci anomálií v rámci SNORT pro metodu INVITE, kdy je počet zpráv větší než 100 za minutu a pravidlo, pokud je počet všech zpráv, které využívají protokol TCP, větší než 300 za minutu. Dále je popsáno,

že v rámci RFC jsou definována pole pro SIP hlavičku. Jejich počet je 115, z toho je 109 polí volitelných. V rámci SDP protokolu je celkem 15 polí, z toho je pouze 5 volitelných, což představuje velký potenciál pro využití steganografických metod. V neposlední řadě je zde zníněno, že je různé typy SIP zpráv generují další zprávy jako např. metoda INVITE, která je následována zprávou 200 OK a zprávou ACK. Pro steganografické účely je výhodnější použit metody INFO nebo OPTIONS, které generují jen jednu zprávu 200 OK.

Vypočítaná přenosová kapacita skrytého kanálu je asi 464 kbps při použití 1 přidané SIP zprávy za sekundu a 1856 kbps při použití 4 přidaných SIP zpráv za sekundu.

USA DoD (Department of Defence) je americké ministerstvo obrany, které považuje každý skrytý kanál s šířkou pásma větší než 100 bitů za sekundu za nejisté z hlediska bezpečnosti.

3 Metody pro detekci anomálií

V této kapitole jsou rozebrány systémy pro detekci anomálií v síťovém provozu a jejich varianty. Podrobně jsou popsány systémy IDS, NBAD, UTM. Subsystémy, které tvoří UTM jako DLP a ILP. Následně je popsána klasifikace u metod pro detekci průniků. Dále jsou rozebrány používané algoritmy Naive Bayes, Holt-Winters a jeji modifikace Holt-Winters Brutalg, které využívám pro predikci síťového provozu v rámci této práce. V další podkapitole se věnuji programu SNORT, který reprezentuje softwarovou síťovou sondu. Tato sonda se využívá na Unixových systémech pro analýzu a aktivní či pasivní zásahy do komunikace. Jsou popsány komponenty, operační módy, pravidla a samotná implementace v rámci mé práce. V poslední části této kapitoly se věnuji preprocesoru pro SNORT, který se nazývá AD (Anomaly Detection). Popíšu parametry, které Anomaly Detection preprocesor analyzuje, strukturu systému a konfiguraci. Závěr této kapitoly obsahuje implementaci Anomaly Detection preprocesoru v rámci této práce.

3.1 Obecně

Detekce anomálií v komunikačních (datových) sítích je díky stále se zvětšujícímu množství datových toků a počítačových útoku velmi obtížný úkol. Nicméně uživatelé v sítích, vyžadují jejich vysoké zabezpečení. Monitorování každého paketu přeneseného v rámci sítě by bylo velmi náročné z hlediska výpočetního výkonu a případného zpomalení přenosu dat. Z tohoto důvodu se používají systémy pro monitorování síťového provozu, které jsou schopny detekovat anomálii v síti, přijmout vhodná opatření a informovat při detekování nebezpečí či případně zasáhnout proti nezbezpečí.

Mezi takové patří systémy (N)IDS, IPS, NBAD/ADS, UTM[1] [3].

3.2 Detekční systémy

3.2.1 (N)IDS - (Network) Intrusion Detection Systems

(N)IDS (Systém odhalení průniku) je monitorovací softwarové nebo hardwarové řešení zaměřené na detekci pokusu o průnik do chráněné sítě a dalších podezřelých aktivit. Také se zabývá aktivitami, které předcházejí samotnému finalnímu útoku jako např. skenování portů a dalšímu sběru informací potřebných k útoku.

Hlavním prvkem IDS je senzor, který implementuje mechanismy pro detekci anomálií v síti. Jelikož se jedná o systém pasivní, tak v případě detekce anomálie vygeneruje upozornění tzv. alert a zapíše záznam do logu.

[1] Rozdělení jednotlivých systému a jejich funkcí není fixní. V současné době se funkce jednotlivých systému prolínají.

3.2.2 (N)IPS - (Network) Intrusion Prevention Systems

(N)IPS[2] při detekovaní anomálie v sítí vygeneruje upozornění (alert) a poté je aktivně zasaženo např. zablokováním služby. Případně je resetováno spojení a přeprogramována nastavení firewallu tak, aby měl nebezpečný zdroj zablokován přístup do sítě.

Systém (N)IPS dokáže zaznamenat i útok vedený zevnitř sítě a zasáhnout proti němu narozdíl od firewallu, který se snaží zabránit proniknutí omezením či zablokováním přístupu mezi sítěmi, ale neumožňuje signalizaci útoku zevnitř sítě.

3.2.3 NBAD - Network Behavioral Anomaly Detection / ADS - Anomaly Detection System

Jsou systémy pracující na podobném principu jako systémy IDS. Narozdíl od IDS nevyužívají pro detekci anomálií detekční pravidla, ale vnitřní model sítě, který popisuje chování sítě. Detekce hrozeb je založena na principu porovnávání aktuálního provozu s provozem, který vychází ze standardního modelu. Standardní model se v průběhu času mění spolu se změnou chování modelu sítě. V případě že systém při porovnávání obou modelů detekuje změnu, která přesahuje určité hranice provozu, vyvolá systém varování (alert). Systém je schopný detekovat i malé odchylky datového toku, využití legitimní šířky přenosového pásma.

Výhodou tohoto systému je shopnost detekovat nežádoucí kód, který se ukrývá v běžně používaných protokolech, jako je např. HTTP či DNS.

Nevýhodou toho systému je vyšší míra falešných poplachů, z čehož vyplývá nutnost další manuální analýzy a rozhodnutí zda jde o falešný poplach nebo o reálnou hrozbu [10].

Další podstatnou nevýhodou tohoto systému je, že v momentě, kdy se vytváří referenční model chování sítě a v sítí probíhá tajná komunikace, tak je tato komunikace zaznamenaná do referenčního modelu, čímž je výrazně omezena až znemožněna jeho budoucí detekce.

3.2.4 UTM - Unified Threat Management

UTM (jednotná správa hrozeb) je jednotný systém pro správu hrozeb počítačových sítí. Tento systém vytváří komplexní ochranu před bezpečnostími riziky, které v rámci síťové komunikace vznikají.

UTM systém tvoří řada subsytému, které plní specializované funkce jako je např. Firewall Gateway, Antispam Gateway, Content Filtering, Parental Control, Load Balancing, Bandwidth Management, DLP (Data Loss Prevention) a On-Appliance (využití aplikací) reporty.

[2] V literatuře lze též nalézt označení IDPS (Intrusion Detection and Prevention Systems), což označuje vzájemný průnik systému IDS a IPS.

3.3 Metody detekce průniků

Detekce průniků do informačního systému (IS) je založena na myšlence, že každý útok má určité atributy a formát, který lze definovat a poté strojově číst. Další možností je zaznamenávat chování sítě, a poté porovnávat jeho chovaní s predikovaným modelem. V neposlední řadě existuje metoda analýzy protokolů, kde jejich pravidla vycházejí z parametru definovaných v RFC dokumentech.

3.3.1 Signature-Based Detection / Knowleadge-Based Detection

Signature-Based Detection (porovnávání signatur) je metoda využívána pro detekci průniku v systémech IDS / IPS. Systém sleduje pakety v síti a porovnává je s databází signatur nebo atributů již známých hrozeb. Vzhledem k tomu, že je nejprve potřeba zavést hrozbu do databáze, aby mohla být později detekováná. Tento problém popisují, tzv. zero-day útoky.

3.3.2 Behavioral Analysis / Statistical Anomaly-Based

Behavioral Analysis (Behaviorální analýza) metoda založena na vytvořeném standardním (referenčním) modelu sítě (tzv. baseline), který reprezentuje normalní stav sítě a jeho následném porovnání s aktuálním provozem. Standarní model definuje šířku přenosového pásma v závislosti na portu. V případě, že se chování sítě výrazně odlišuje od standardního modelu a přesáhne určité meze, dojde k vyvolání poplachu (alert). Pro tuto metodu je typický zvýšený výskyt falešných poplachů. Odchylky v chování IS vzhledem ke standardnímu modelu (stavu) sítě resp. k predikovanému modelu se zaznámenavají jako anomálie systému.

Výhodou této metody je, že dokáže oproti signature-based metodě detekovat i nové, zatím nezaznamenané útoky.

3.3.2.1 Fáze učení Ve fázi učení se vytváří referenční model, který obsahuje velké množství informací o síťovém provozu. Časový interval zachyceného referenčního modelu musí být dostatečné velký, aby dostatečně kvalitně reprezentoval reálné chování systému. Statistické algoritmy vyžadují různou délku časového intervalu referenčního modelu. Většinou se jedná o období od 2 dní až po 3 týdny.

Zásadním problémem této metody je, že během fáze učení není systém IDS použitelný a může generovat více falešných poplachů. Zároveň vzniká riziko, kdy jsou během této fáze zaznamenány potenciální útoky a zahrnuty do referenčního modelu. V tom případě by IDS později vyhodnotil potenciální útok jako běžnou událost.

3.3.2.2 Fáze detekce Tato fáze vychází z porovnávání dat referenčního a predikovaného modelu. Predikovaný model je generován pomocí statistických algoritmů, jejichž výstupem jsou hodnoty síťového provozu pro následující období. Zároveň jsou definovány meze, které pokud jsou překročeny v porovnání s aktuálním provozem, tak jsou vyhodnoceny jako anomálie. Predikované modely se pravidelně aktualizují.

Míra vyvolaných poplachů je závislá na použitém algoritmu, který definuje *minimální* a *maximální meze* tzv. confidence bands v daném čase, pro daný atribut predikovaného modelu.

Pro predikci modelu využívají statistické algoritmy: AVG, NAIVE, Autoregressive, Holt-Winters a Holt-Winters:Brutlag.

3.3.3 Stateful Protocol Analysis Detection

Je metoda detekce anomálií, která vychází z přesných stavů protokolů provozovaných v sítích na základě definic v dokumentech RFC a jiných standardech. Činnost protokolu je dána a přechody mezi jednotlivými stavy řídí stavový automat. Jestliže se automat dostane do nedefinovaného stavu protokolu, je tento stav považován za anomálii [9]. IDS SNORT obsahuje SIP preprocesor, který je představitelem tohoto přístupu.

3.4 Algoritmy

Mezi statistické algoritmy, které se využívají pro výpočet predikovaného síťového modelu patří algorigmus Naive Bayes, Holt-Winters a jeho modifikace Holt-Winters:Brutlag, které využívám v rámci mé práce. Existují i další algoritmy, které jsou implementovány v rámci SNORT AD jako např. klouzavý průměr nebo autoregrese.

3.4.1 Naive Bayes

Naive Bayes algoritmus je klasifikační algoritmus, který je založený na Bayesově teorému. Při klasifikaci se využívá tzv. naivního přístupu, který je založen na silných (naivních) předpokladech nezávislosti mezi charakterisktikami prvku tj. že prvky v dané množině jsou na sobě nezávislé, což ale ve skutečnosti nemusí být pravda. Výpočet vychází z počtu podmíněných pravděpodobností jednotlivých prvků pro různé třídy.

$$\widehat{y}_t = y_{t-T} \tag{2}$$

kde $\widehat{y}$ je predikováná hodnota proměnné v čase t, y_t je reálná (naměřená) hodnota proměnné v čase t a T je denní nebo tydenní periodicita.

3.4.2 Holt-Winters

Holt-Wintersova metoda také nazývána *Triple Exponential smoothing* je adaptivní model využívaný k predikci vývoje časové řady na základě historických dat. V roce 1957 ji navrhnul Charles C. Holt a poté ji v 60. letech 20. století vylepšil student Peter R. Winters.

Metoda Holt-Winters existuje ve dvou variantách: aditivní a multiplikativní [14]. Aditivní varianta, kterou využívám ve své práci, se vyznačuje proměnlivostí hodnot časové řady, která je přibližně konstatní v čase a je tak vhodnější pro data s neměnnou sezónosti. Sezónní složka je v rámci aditivní varianty vyjádřena v absolutní hodnotě, řada je očistěna o sezónní složku jejím odečtením. Multiplikativní varianta je charakteristická tím, že variabilita časové řady roste v čase nebo se v čase mění. Sezónní složka je vyjádřena

v relativní hodnotě, řadu očistíme jejím vydělením [15]. Ve své práci využívám aditivní variantu H-W metody.

Tuto metodu lze popsat čtyřmi charakteristikami:

- **trend** - je vzrůst (uptrend) nebo pokles (downtrend) křivky v dlouhodobém horizontu,
- **sezónní trend** - je oscilace křivky k minimálním a maximálním hodnotám v rámci denního cyklu,
- **sezónní proměnlivost** - je míra fluktuace křivky, zapřičiněná rozdílem maximálních a minimimálních hodnot v čase,
- **celkový vývoj** - je pohled na křivku, kdy je zachycena progrese vývoje časové řady.

Metoda Holt-Winters vytváří predikci na základě historických dat, přičemž váha jednotlivých složek, kterou ovlivňují výsledek, exponenciálně klesá směrem do minulosti.

V rámci trojitého vyhlazování pracujeme s časovou řadou, kterou rozdělujeme na tři koeficienty (složky) [31]:

- **L - Level (báze)**:

$$L_t = \alpha(y_t - S_{t-T}) + (1 - \alpha)(L_{t-1} + P_{t-1}) \tag{3}$$

- **P - Trend**:

$$P_t = \beta(L_t - L_{t-1}) + (1 - \beta)P_{t-1} \tag{4}$$

- **S - Sezónnost**:

$$S_t = \gamma(y_t - L_t) + (1 - \gamma)S_{t-T} \tag{5}$$

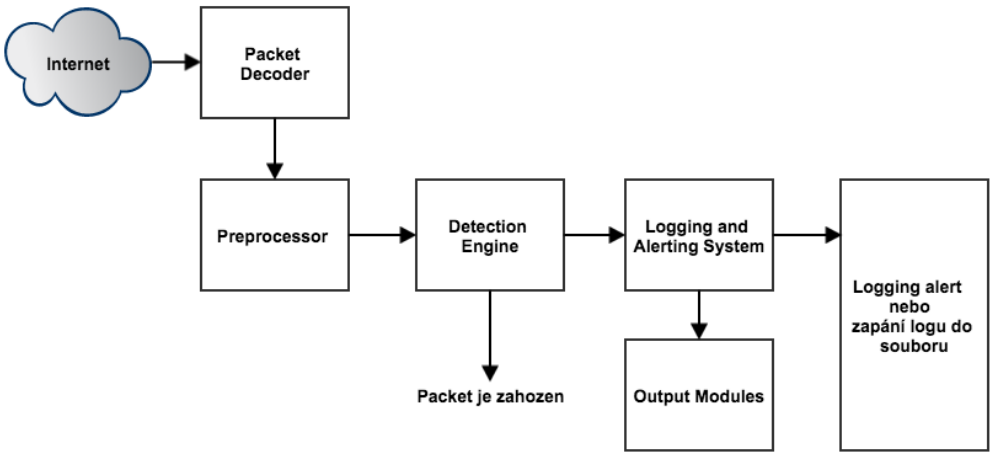
kde α je faktor vyhlazování dat (data smoothing factor), β je faktor vyhlazovaní trendu (trend smoothing factor), γ je faktor vyhlazování sezónní proměnlivosti (seasonal change smoothing factor), y_t je reálná (naměřená) hodnota proměnné v čase t, t je okamžik v čase a T je období časové řady,

Suma těchto tří složek reprezentuje predikci:

$$\widehat{y}_t = L_{t-1} + P_{t-1} + S_{t-T} \tag{6}$$

kde $\widehat{y}$ je predikováná hodnota proměnné v čase t, t je okamžik v čase, T je období časové řady, L je složka reprezentující Level (báze), P je složka reprezentující Trend a S je složka reprezentující Sezónnost.

Adaptační parametry α, β, γ mohou nabývat hodnoty: $0 < \alpha, \beta, \gamma < 1$. Tyto hodnoty adaptačních parametrů určují rychlost adaptace algoritmu na změny vstupních hodnot. Čím se hodnoty v intervalu $\langle 0, 1 \rangle$ blíží více k 0 je rychlost adaptace nižší, a výsledné predikované hodnoty budou vycházet více z historických hodnot. Naopak pokud se hodnoty blíží k 1, rychlost adaptace bude vysoká, a tím výsledné predikované hodnoty budou vycházet více z aktuálních trendů.



Obrázek 3: Komponenty SNORTu.

3.4.3 Holt-Winters-Brutlag

Tento algoritmus je modifikací algoritmu Holt-Winters. V roce 2000 jej představil Jake D. Brutlag.

$$\widehat{y}_t^{max} = L_{t-1} + P_{t-1} + S_{t-T} + md_{t-T} \tag{7}$$

$$\widehat{y}_t^{min} = L_{t-1} + P_{t-1} + S_{t-T} - md_{t-T} \tag{8}$$

kde $\widehat{y}$ je predikováná hodnota proměnné v čase t, t je okamžik v čase, T je období časové řady, L je složka reprezentující level (báze), P je složka reprezentující trend a S je složka reprezentující sezónnost, m je faktor změny hranice důvěry pro Brutlag a d je predikovaná odchylka daná rovnicí:

$$d = \gamma \mid y_t - \widehat{y}_t \mid +(1-\gamma)d_{t-1} \tag{9}$$

3.5 SNORT

SNORT [18] je softwarová síťová sonda dostupná pro UNIX a Windows systémy patřící do kategorie NIDS, založená na definování pravidel v rámci síťového provozu a jeho analýze. Původní koncepce umožňovala použití v tzv. packet sniffer módu. V současné době již umožňuje pracovat v módech packet logger, NIDS a inline.

3.5.1 Komponenty

Snort je logicky rozdělen do několika komponent (Obrázek 3). Tyto komponenty pracující společně a jsou schopny detekovat útoky a generovat výstup v požadovaném formátu. Komponenty využívají Libpcap a WinPcap knihovny pro zachytávání a filtrování paketů.

3.5.1.1 Zachytávání bloků paketů Tato komponenta slouží k zachytávání paketů. Tyto pakety jsou zachytávány ze síťových rozhraní a předávány dekodéru pro další zpracování.

3.5.1.2 Dekodér Je komponenta odpovědná za provedení analýzy syntaxe na MAC, IP, TCP/UDP vrstvě IP paketu.

3.5.1.3 Preprocessor Komponenta, která umožňuje načítat preprocesory, které jsou schopny analyzovat, uspořádávat a upravovat pakety TCP/UDP vrstvy předtím, něž detekční jednotka provede analýzu, zda se jedná o paket, který je potenciálně nebezpečný či nikoliv. Některé používané preprocesory, vyhledávají anomálie v hlavičkách paketů. Preprocesory jsou napsány v jazyce C++. Ve své práci využívám preprocesor AD (Anomaly Detection) (popsán samostatně níže).

3.5.1.4 Detekční engine Jedná se o nejdůležitější komponentu SNORTu. Jejím úkolem je systematicky analyzovat data uvnitř každého packetu, a porovnávat je s řetezci nebo hodnotami definovanými v pravidlech. Jestliže paket odpovídá některému z pravidel, pak je vyvolána odpovídající akce. Může se jednat o vytvoření alertu nebo zápis do logu.

Detekční engine je náročná komponenta z hlediska výpočetního výkonu. Výkon komponenty může ovlivňovat počet definovaných pravidel, výpočetní výkon počítače, na kterém je SNORT spuštěn, rychlost vnitřní sběrnice počítače a zatížení sítě.

3.5.1.5 Generování alertu a logovaní Tato komponenta se stará v případě, že detekční engine zjistí anomálii o to, aby byla tato událost zaznamenána do log souboru nebo byl vyvolán alert. Logy mají strukturu textového souboru v tcp-dump tvaru nebo jiných formátech. Všechny logovací soubory se implicitně ukládají do adresáře */var/log/snort*.

3.5.1.6 Výstupní zásuvné moduly Tato komponenta provádí operace nad výstupy vytvořenými komponentou generování alertu a logování. Podle nastavení jednotlivých modulů mohou vykonávat akce: zaznamenávaní logu do jiných formátů, zasílaní SNMP trapů, zasílání zprávy do syslogu, zapisování do databáze MySQL nebo Oracle, generování XML výstupů či modifikovat konfiguraci routerů a firewallů [16].

3.5.2 Operační módy

SNORT je možné spustit v několika módech, které se liší svými primárními úkoly. K dispozici jsou módy: sniffer, packet logger, NIDS a inline.

3.5.2.1 Sniffer Mód, který zachytává pakety procházející sítí a zobrazuje je v nepřetržitém proudu na konzoli.

```
./snort -v -[option]
```

Výpis 2: Příkaz pro spuštění SNORT v módu sniffer.

3.5.2.2 Packet logger Tento mód zapisuje log soubory na disk. Pro spuštění stačí zadat parametr *-l* a specifikace adresáře pro logování a SNORT automaticky přejde do tohoto módu.

```
./snort -l /var/log/snort -[option]
```

Výpis 3: Příkaz pro spuštění SNORT v módu packet logger.

3.5.2.3 NIDS Je nejvíce komplexní a konfigurovatelný mód, který umožňuje analyzovat provoz v rámci uživatelem definovaných pravidel a vykonává případné opatření. Pro spuštění tohoto módu je nutné zadat pomocí parametru *-c* cestu ke konfiguračnímu souboru *snort.conf*.

```
./snort -c /etc/snort/snort.conf -[option]
```

Výpis 4: Příkaz pro spuštění SNORT v módu NIDS.

Po spuštění SNORT se zobrazí na konzoli, nastavení, pravidla a preprocesory, které budou použity pro detekci anomálií.

3.5.2.4 Inline Tento mód získává pakety z iptables místo využití knihoven Libpcap a WinPcap. Na základě pravidel je rozhodováno o zahození či povolení paketů. Pokud je SNORT spuštěný v tomto režimu vystupuje jako IPS.

3.5.3 Pravidla

Pravidla ve SNORTu jsou popsány jednoduchým a silným jazykem. Je velmi jednoduché je vytvářet a modifikovat. Každé pravidlo se skládá z *hlavičky (head)* a *volby (options)*.

Hlavičku pravidla reprezentuje volba akce, která se má vykonat protokol, ke kterému se pravidlo váže, zdrojovou adresu s portem a cílovou adresu s portem. Ve volbách je možné nastavit velké množství atributů jako např. hodnoty a řetězce obsažené v paketu.

```
alert udp any any -> 158.196.244.197 5060:5064 ( msg: "AD␣HIGH␣VALUE␣OF␣DOWNLOAD␣
    UDP␣DATA␣SPEED";
sid:1000152; gid:1000100; rev: 1; metadata: rule-type preproc; classtype:bad-unknown; )
```

Výpis 5: Nastavení pravidla ve SNORT.

3.5.3.1 Hlavička (head) Podrobněji rozeberu atributy, které obsahuje hlavička v následujícím textu. Detailní popis je dostupný v dokumentaci SNORT [17].

- **Akce** - Atribut *alert* vyvolá výstrahu a zaznamenává ji do log souboru. Další hodnoty, které může atribut nabývat jsou: log, pass, active, dynamic, drop, reject a sdrop.
- **Protokol** - Atribut *udp* popisuje protokol, na který se váže dané pravidlo. Další hodnoty, které může atribut nabývat jsou: TCP, UDP, ICMP a IP.
- **IP adresa** - Atribut *any* popisuje IP adresu, která se váže danému pravidlu. V tomto případě se jedná o jakoukoliv adresu, která komunikuje se stanicí reprezentovanou adresou 158.196.244.197 na portu 5060 až 5064.
- **Port** - Atribut *any* popisuje port, na který se váže dané pravidlo. V tomto případě se jedná o jakýkoliv port, který komunikuje se stanicí reprezentovanou adresou 158.196.244.197 na portu 5060 až 5064.
- **Směr komunikace** - Atribut „->“ reprezentuje směr komunikace, na které se pravidlo vztahuje. Další hodnoty, které může atribut nabývat jsou: „<-“ nebo „<>“.
- **IP adresa** - Atribut *158.196.244.197* popisuje IP adresu, která se váže k danému pravidlu. V tomto případě s cílovou adresu 158.196.244.197 na portu 5060 až 5064.
- **Port** - Atribut *5060:5064* popisuje port, na který se váže dané pravidlo. V tomto případě se jedná o cílové porty v rozsahu 5060 až 5064, na kterých probíha monitorování.

3.5.3.2 Volby (options) Volby nastavení tvoří nejdůležitější část z hlediska detekce anomálií. Všechny možnosti volby nastavení jsou od sebe odděleny pomocí znaku středníku (;). Klíčové slovo volby nastavení (atribut) je od svých argumentů oddělen pomocí znaku dvojtečka (:). Podrobněji rozeberu atributy, které je možné využít v rámci volby nastavení. Detailní popis je dostupný v dokumentaci SNORT [17].

- **Zpráva** - Atribut *msg* reprezentuje zobrazovaný text či text zaznamenaný do log souboru při vyvolání alertu. V konkretním případě se zobrazí text "AD HIGH VALUE OF DOWNLOAD UDP DATA SPEED".
- **SNORT ID** - Atribut *sid* je unikátní ID SNORT pravidla. Je reprezentováno numerickou hodnotou. Konkétně: 1000152.
- **Generator ID** - Atribut *gid* je unikátní ID, které se váže na subsystémy jako jsou preprocesory a generátory. Je reprezentováno numerickou hodnotou. Konkétně: 1000100.
- **Revisions** - Atribut *rev* popisuje hodnotu revize daného pravidla. Konkrétně se jedná o hodnotu: 1.

- **Metadata** - Atribut *„Metadata“* blíže specifikuje resp. popisuje dané pravidlo. Konkrétně se jedná o hodnotu: rule-type preproc.
- **Classtype** - Atribut *classtype* slouží ke kategorizaci pravidel resp. útoků. SNORT poskytuje základní sadu tříd pro rozdělení úrovně nebezpečnosti v případě vyvolání anomálie. Konkrétně se jedná o hodnotu: classtype:bad-unknown.

3.5.4 Implementace

V této části popisuji detailní proces implementace SNORT pro potřeby této práce. Pro aplikování steganografikých technik a prozkoumání jejich limitů v rámci modifikace SIP hlavičky je potřeba využít technik pro jejich detekci.

Ve své práci využívám linuxovou distribuci Ubuntu 12.04.3 LTS, která byla virtualizována pomocí VM WARE. Také využívám čtyři virtualní servery v sítí LIPTEL, VŠB-TU Ostrava, kde je taktéž nainstalován Linux Ubuntu 12.04.3 LTS. Před samotnou instalací IDS SNORT je potřeba nainstalovat několik balíčků pro správnou funkčnost SNORTu, nicméně tato problematika přesahuje rozsah této práce.

3.5.4.1 Vytváření profilu síťového provozu Pro vytvoření referenčního síťového provozu jsem vytvořil komunikaci mezi dvěma servery *.196* a *.197*. Na serveru *.196* byly spuštěny dvě instance UAC klienta se software SIPp (více: Realizace výměny informací pomocí vloženého pole v SIP signalizaci) pro generování SIP komunikace. Na serveru *.197* byly spuštěny dvě instance UAS klieta se software SIPp a SNORT.

Pro správné spuštění SNORT je potřeba upravit konfigurační soubor, který se nachází v adresáři */etc/snort/snort.conf*, kde je potřeba zadat cestu, kam se bude ukladát log soubor. Dále je nutné nastavit časový interval, který je v tomto případě v délce 60 sekund.

```
preprocessor AnomalyDetection: LogPath /var/log/snort log time 60
```

Výpis 6: Konfigurace snort.conf pro vytvoření log souboru.

V další fázi se spustí samotné zaznamenávání síťového provozu následujícím příkazem, kde parametr *-c* určuje cestu ke konfiguračnímu souboru a parametr *-h* určuje, z jaké IP adresy se má provoz zaznamenávat,

```
snort -c /etc/snort/snort.conf -h 158.196.244.197
```

Výpis 7: Příkaz pro zaznamenání síťového provozu do log souboru.

Po spuštění příkazu vidíme na konzoli, kontinuální výpis zápisu do log souboru s frekvencí 60 sekund.

```
Loged transfer between 23-03-15 00:07:01 - 23-03-15 00:08:01
Loged transfer between 23-03-15 00:08:01 - 23-03-15 00:09:01
Loged transfer between 23-03-15 00:09:01 - 23-03-15 00:10:01
Loged transfer between 23-03-15 00:10:01 - 23-03-15 00:11:01
Loged transfer between 23-03-15 00:11:01 - 23-03-15 00:12:01
Loged transfer between 23-03-15 00:12:01 - 23-03-15 00:13:01
Loged transfer between 23-03-15 00:13:01 - 23-03-15 00:14:01
```

```
Loged transfer between 23−03−15 00:14:01 − 23−03−15 00:15:01
Loged transfer between 23−03−15 00:15:01 − 23−03−15 00:16:01
Loged transfer between 23−03−15 00:16:01 − 23−03−15 00:17:01
```

Výpis 8: Kontinuální výpis zápisu do log souboru s frekvencí 60 sekund.

Zápis dat díky AD preprocesoru probíha do souboru *ADLogx.txt,* kde *x* reprezentuje frekvenci záznamu v sekundách.

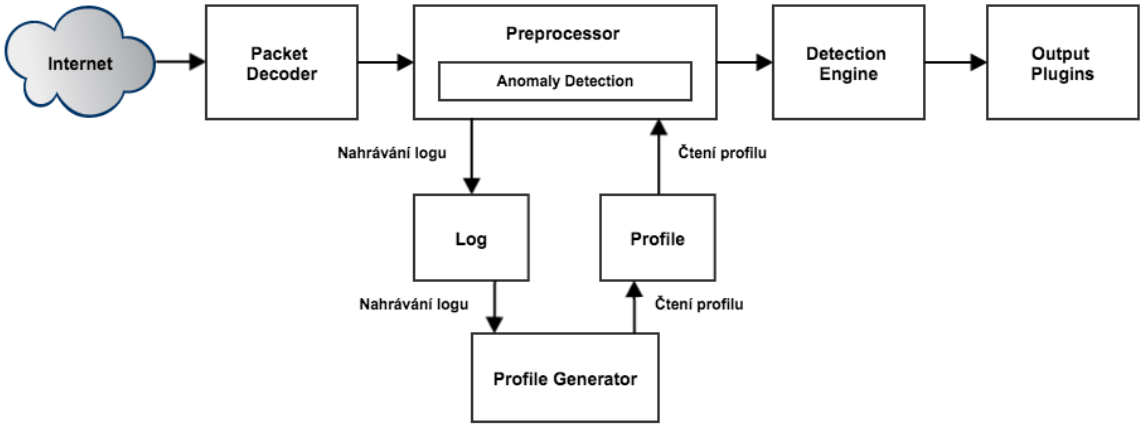
3.5.4.2 Výsledný log soubor referenčního síťového provozu

Log soubor obsahuje 29 parametrů, které popisují síťový provoz (více: Parametry síťového provozu). Pro účely mé práce využiji hodnoty počtu přenesených paketů a rycholost přenosu v rámci UDP protokolu.

```
24−03−15,18:04:01,Tue,60,508,301,207,0,7261,3960,3300,0,248,120,120,
0,0,0,0,0,0,45,0,0,8123,3.21,0.18,0.00,0.00,19.94,18.76,0.00,0.00

24−03−15,18:05:01,Tue,60,511,302,209,0,7260,3960,3300,0,262,120,120,
0,0,0,0,0,0,49,0,0,8143,3.21,0.18,0.00,0.00,19.94,18.76,0.00,0.00

24−03−15,18:06:01,Tue,60,500,301,199,0,7260,3960,3300,0,246,120,120,
0,0,0,0,0,0,48,0,0,8114,3.21,0.17,0.00,0.00,19.94,18.76,0.00,0.00

24−03−15,18:07:01,Tue,60,512,302,210,0,7262,3960,3300,0,266,120,120,
0,0,0,0,0,0,49,0,0,8150,3.23,0.18,0.00,0.00,19.94,18.76,0.00,0.00

24−03−15,18:08:01,Tue,60,507,300,207,0,7262,3961,3301,0,247,120,120,
0,0,0,0,0,0,46,0,0,8123,3.19,0.18,0.00,0.00,19.94,18.76,0.00,0.00

24−03−15,18:09:01,Tue,60,509,301,208,0,7260,3960,3300,0,263,120,120,
0,0,0,0,0,0,48,0,0,8141,3.21,0.18,0.00,0.00,19.94,18.76,0.00,0.00

24−03−15,18:10:01,Tue,60,505,300,205,0,7262,3961,3301,0,246,120,120,
0,0,0,0,0,0,44,0,0,8118,3.19,0.18,0.00,0.00,19.94,18.76,0.00,0.00

24−03−15,18:11:01,Tue,60,508,301,207,0,7260,3960,3300,0,262,120,120,
0,0,0,0,0,0,44,1,0,8136,3.21,0.18,0.00,0.00,19.94,18.76,0.00,0.00

24−03−15,18:12:01,Tue,60,508,302,206,0,7262,3960,3300,0,253,120,120,
0,0,0,0,0,0,50,0,0,8134,3.21,0.18,0.00,0.00,19.94,18.76,0.00,0.00

24−03−15,18:13:01,Tue,60,504,301,203,0,7260,3960,3300,0,261,120,120,
0,0,0,0,0,0,47,0,0,8133,3.21,0.17,0.00,0.00,19.94,18.76,0.00,0.00
```

Výpis 9: Výpis ADLog60.txt log souboru s frekvencí 60 sekund.

V zaznamenaném provozu (Výpis 9) je celkový počet přenesených UDP paketů reprezentován hodnotou *7260*. Počet odeslaných UDP paketů je reprezentován hodnotou *3960*. Počet přijatých UDP paketů je reprezentován hodnotou *3300*. Rychlost UDP uploadu reprezentuje hodnota *19.94* kBps. Rychlost UDP downloadu reprezentuje hodnota *18.76* kBps.



Obrázek 4: Schéma detekce anomálií pomocí preprocesoru Anomaly Detection.

3.6 AD - Anomaly Detection

Anomaly Detection (AD) [18] je preprocesor navržený pro SNORT, který zajišťuje detekci anomálií v rámci síťového provozu. AD dokáže analyzovat provoz TCP, UDP, ICMP, ARP a rychlost odesílaní a přijímání paketů. Pro vstupní data využívá AD log soubor SNORTu, který se nachází v adresáři */var/log/snort/ADLog60.txt*. Výstupní data reprezentuje soubor *PROFILE60.txt* umístěný v adresáři */etc/* a má stejnou datovou strukturu jako log soubor SNORT. AD pracuje s celkem 29 parametry síťového provozu, které obsahují soubory *ADLog60.txt* a *PROFILE60.txt* (Obrázek 4).

3.6.1 Parametry síťového provozu

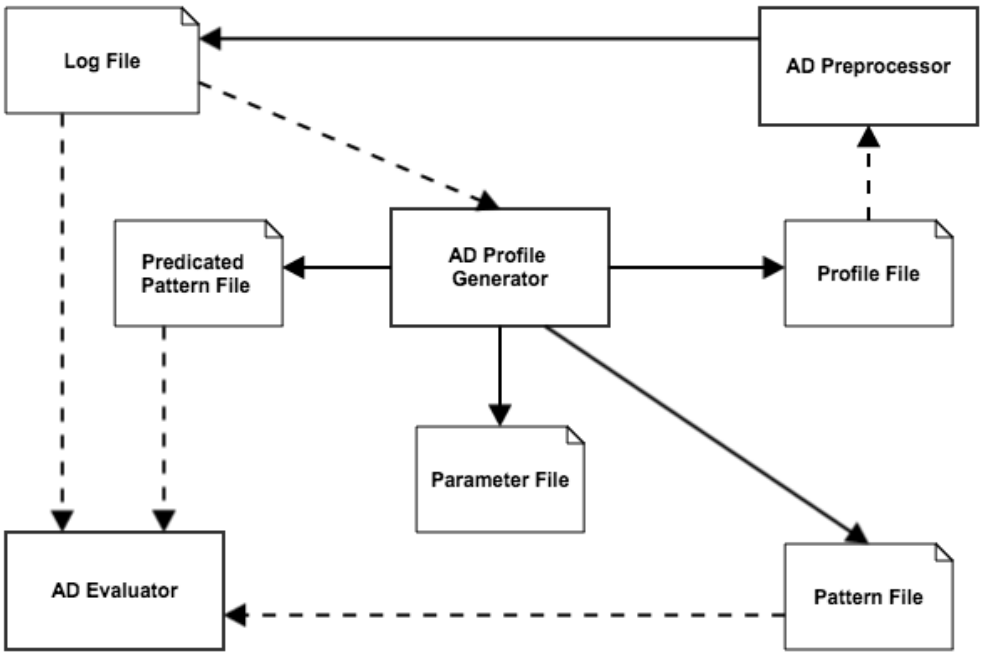
Preprocesor AD je schopen sledovat následující parametry. Výstupní soubor *PROFILE60.txt* navíc obsahuje minimální a maximální hodnotu daného parametru, čímž se počet výstupních hodnot zdvojnásobí.

- Celkový počet přenesených TCP paketů
- Počet odchozích TCP paketů
- Počet příchozích TCP paketů
- Počet TCP paketů v rámci podsítě
- Celkový počet přenesených UDP paketů
- Počet odchozích UDP paketů
- Počet příchozích UDP paketů
- Počet UDP paketů v rámci podsítě

- Celkový počet přenesených ICMP paketů
- Počet odchozích ICMP paketů
- Počet příchozích ICMP paketů
- Počet ICMP paketů v rámci podsítě
- Počet paketů TCP s SYN/ACK
- Počet odchozích TCP paketů z portu 80 (WWW)
- Počet příchozích TCP paketů z portu 80 (WWW)
- Počet odchozích UDP paketů z portu 53 (DNS)
- Počet příchozích UDP paketů z portu 53 (DNS)
- Počet ARP-request paketů
- Počet ARP-replay paketů
- Počet NOT TCP/IP paketů
- Celkový počet všech přenesených paketů
- Rychlost TCP upload (kBps)
- Rychlost TCP download (kBps)
- Rychlost WWW upload (kBps)
- Rychlost WWW download (kBps)
- Rychlost UDP upload (kBps)
- Rychlost UDP download (kBps)
- Rychlost DNS upload (kBps)
- Rychlost DNS download (kBps)

3.6.2 Struktura systému

Preprocesor AD se skláda z několika komponent, které zajišťují, vyhodnocují a porovnávají jednotlivé anomálie resp. algoritmy použité pro výpočet predikovaného modelu síťového provozu (Obrázek 5).



Obrázek 5: Struktura Anomaly Detection systému.

3.6.2.1 AD Profile Generator Profile Generator komponenta slouží ke generování souboru *PROFILE60.txt*, který reprezentuje predikovaný síťový provoz na základě dat získaných z předchozího síťového provozu ze souboru *ADLog60.txt*. Jedná se o nejdůležitější komponentu celeho AD preprocesoru. Je založen na prostředí pro statistické výpočty R [22]. Umožňuje generovat profily na základě pěti algoritmů:

- Moving average (klouzavý průměr),
- Naive method (naivní metoda),
- Autoregressive time series (autoregresivní časové řady),
- Holt-Winters,
- Holt-Winters:Brutlag.

Po zvolení algoritmu, který bude generovat predikovaný model, lze využít jeho další možnosti nastavení. Jedním z důležitých parametrů je délka období, které se má použít pro výpočet. Možnosti nastavení období:

- *LAST* - predikovaný model vychází z posledních zaznamenaných dat,
- *DAILY* - predikovaný model vychází z dat zaznamenaných za 24 hodin,

- *WEEKLY* - predikovaný model vychází z dat zaznamenaných za jeden týden a pro každý den je vypočítan odpovídající model.

Dalším z mnoha parametrů, které lze nastavit je počet predikovaných hodnot. Toto nastavení se provede pomocí parametru *-a | –ahead*, kde argumentem je právě počet požadovaných predikovaných hodnot, pro které má být model vypočítán.

Profile Generator určuje průměr hodnot a jejich směrodatnou odchylku pro každý parametr. Generovaný soubor slouží pro vyvolávání výstrah.

3.6.2.2 AD Preprocessor Hlavní funkcí preprocesoru je generovaní výstrah. Preprocesor čte všechny parametry predikovaného modelu síťového provozu ze souboru *PROFILE60.txt* a generuje výstrahy, pokud aktuální hodnota přesahuje minimální nebo maximální hodnotu definovanou v souboru *PROFILE60.txt*. Aktuální časový okamžik je reprezentován dnem v týdnu, hodinou, minutou, sekundou (záleží na nastavení SNORT v souboru */etc/snort/snort.conf* v jakém časovém intervalu se budou data zaznamenávat).

Výstrahy jsou generovány podle následujícího vzorce:

$$W \notin (w - 2\sigma; w + 2\sigma) \tag{10}$$

kde *W* je hodnota aktuálního provozu a σ je hodnota směrodatné odchylky(deviation scale), získaná ze souboru pro predikci síťového provozu *PROFILE60.txt*.

3.6.2.3 AD Evaluator Je navržen pro porovnávání statistik *Mean absolute error (průměrné absolutní chyby)* mezi dvěma soubory. Dále může být použit pro porovnání hodnot mezi predikovaným a historickým modelem nebo porovnání mezi predikovaným modelem a reálnými hodnotami.

3.6.3 Implementace

V této části se věnuji implementaci Anomaly Detection preprocesoru, který je nutný pro dosažení výsledků mé práce.

3.6.3.1 Generování predikovaného modelu Po vytvoření log souboru *ADLog60.txt*, jsem přešel k vytváření predikovaného modelu, který je zaznamenán v souboru *PROFILE60.txt*. Log soubor *ADLog60.txt* zachycuje síťový provoz po dobu 72 hodin, což je dostatečná délka pro většinu vypočetních algoritmů, vyjma AVG - klouzavého průměru. Pro vytvoření predikovaného modelu pomocí Holt-Winters jsem použil následujíci příkaz:

```
profilegenerator −l /var/log/snort/ADLog60.txt −p /etc/PROFILE−HW−1.txt −a 10080 −m HW
    −−hw DAILY −d 1 −v
```

Výpis 10: Příkaz pro spuštění AD Profile Generatoru.

Parametr	Hodnota
-l	/var/log/snort/ADLog60.txt
-p	/etc/PROFILE-HW-1.txt
-a (počet predikovaných hodnot)	10080
-m	HW
-hw	DAILY
-d (míra odchylky)	1
-v (verbose režim)	-

Tabulka 5: Nastavení AD Profile Generatoru pro výpočet predikovaného modelu pomocí Holt-Winters

Délka generovaní profile souboru byla 12.258 sekund.

Po vygenerování predikovaného můžeme vidět výsledné hodnoty v následujícím výpise (Výpis 11). Tento výpis obsahuje hodnoty minimální a maximální hodnoty mezí, pro všechny parametry, které jsou popsány v předchozích podkapitolách.

```
26−03−15,4,03:01:01,60,560,979,275,552,135,371,0,0,7052,7469,
3874,4047,3228,3372,0,0,0,2650,115,125,115,125,0,0,0,0,0,0,0,
 0,0,0,0,0,0,108,0,0,0,0,5751,10860,3.15,3.92,0,0.33,0,0,0,0,
19.41,20.31,18.29,19.13,0,0,0,0

26−03−15,4,03:02:01,60,564,983,275,552,138,374,0,0,7052,7469,
3873,4047,3228,3372,0,0,0,2667,115,125,115,125,0,0,0,0,0,0,0,
 0,0,0,0,0,0,110,0,0,0,0,5770,10879,3.14,3.92,0,0.33,0,0,0,0,
19.41,20.31,18.29,19.13,0,0,0,0

26−03−15,4,03:03:01,60,566,985,276,553,139,375,0,0,7054,7471,
3873,4047,3228,3372,0,0,0,2650,115,125,115,125,0,0,0,0,0,0,0,
 0,0,0,0,0,0,114,0,0,0,0,5763,10872,3.15,3.93,0,0.33,0,0,0,0,
19.41,20.31,18.29,19.13,0,0,0,0

26−03−15,4,03:04:01,60,567,986,273,550,142,378,0,0,7052,7469,
3873,4046,3228,3372,0,0,0,2666,115,125,115,125,0,0,0,0,0,0,0,
 0,0,0,0,0,0,111,0,0,0,0,5770,10879,3.12,3.9,0,0.33,0,0,0,0,
19.41,20.31,18.29,19.13,0,0,0,0

26−03−15,4,03:05:01,60,569,988,275,552,140,376,0,0,7052,7469,
3874,4047,3228,3373,0,0,0,2650,115,125,115,125,0,0,0,0,0,0,0,
 0,0,0,0,0,0,107,0,0,0,0,5750,10859,3.15,3.92,0,0.33,0,0,0,0,
19.41,20.31,18.29,19.13,0,0,0,0
```

Výpis 11: Výpis hodnot predikovaného modelu pomocí HW-1.txt.

Z kódu lze vyčíst hodotnoty minimální a maximální meze, které jsou definované Holt-Winters predikovaným modelem s deviation scale 1. Minimální mez má hodnotu: *18.29kBps*. Maximální mez má hodnotu: *19.13kBps*.

Jak můžeme vidět výše výsledný model je celkem striktní k odchylkám síťového provozu.

3.6.3.2 Konfigurace snort.conf souboru pro detekci anomálií Nyní jsem upravil konfigurační soubor */etc/snort/snort.conf* přidáním cesty souboru *preprocessor.rules*, který obsahuje pravidla Anomaly Detection preprocesoru. Dále jsem doplnil cestu k vygenerovanému predikovanému modelu */etc/PROFILE-HW-1.txt* a kličové slovo *alert* pro zaznamenávání anomálií do stejnojmeného souboru.

```
include preprocessor.rules
preprocessor anomalydetection:ProfilePath /etc/PROFILE-HW-1.txt LogPath /var/log/snort alert
    log time
60
```

Výpis 12: Konfigurace snort.conf pro spuštění AD preprocesoru.

3.6.3.3 Úprava pravidel AD preprocesoru Jelikož AD podporuje detekci různých protokolů, upravil jsem soubor s pravidly */etc/preprocessor.rules* pro potřeby mé práce. Odstranil jsem nepotřebné pravidla, které nemají žádnou vypovídající hodnotu o tom, zda byl daný paket injektován obsahem steganogramu či nikoliv a ponechal jsem pouze pravidlo pro detekci překročení maximálního limitu rychlosti stahování dat na UDP portu. Dalším přiznivým efektem bylo zpřehlednění, tím že výstraha byla vyvolána pouze v přpadě překročení jednoho (sledovaného) parametru. Ponechané pravidlo je následující:

```
alert ( msg: "AD_HIGH_VALUE_OF_DOWNLOAD_UDP_DATA_SPEED"; sid:1000153; gid
    :1000100; rev: 1; metadata: rule-type preproc; )
```

Výpis 13: Úprava pravidel AD preprocesoru.

3.6.3.4 Spuštění detekce anomálií Poté co jsem vše nakonfiguroval, zbývalo jen spustit síťový provoz s obsahem steganogramu (více: Realizace výměny informací pomocí vloženého pole v SIP signalizaci) a SNORT. SNORT spouštím následujícím příkazem:

```
sudo snort -e -c /etc/snort/snort.conf -h 158.196.244.197
```

Výpis 14: Spuštění SNORT AD v režimu detekce anomálií.

3.6.3.5 Detekce anomálií Pro zjistění, zda byla detekována anomálie jsem otevřel soubor */var/log/snort/alert*, kde jsem sledoval výskyt výstrah. Frekvenci logování odpovídá i frekvence výskytu výstrah v případě správné detekce anomálie. Testování probíhalo v délce 5-ti minut, pro každé testované nastavení s frekvenci logování 60 sekund. Tzn. že v případě správné detekce anomálie musí být vygenerováno 5 výstrah za 5 minut (více: následující Výpis 15). Později případě, že se nejedná o anomálii počet výstrah by se měl rovnat nule.

```
[**] [1000100:1000153:1] AD_HIGH_VALUE_OF_DOWNLOAD_UDP_DATA_SPEED [**]
[ Priority : 0]
03/27-21:29:51.013769 00:0C:29:88:96:24 -> 00:0C:29:D2:24:44 type:0x800 len:0x181
158.196.244.196:5060 -> 158.196.244.197:5060 UDP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:371
      DF
Len: 343

[**] [1000100:1000153:1] AD_HIGH_VALUE_OF_DOWNLOAD_UDP_DATA_SPEED [**]
[ Priority : 0]
03/27-21:30:51.013599 00:0C:29:88:96:24 -> 00:0C:29:D2:24:44 type:0x800 len:0x181
158.196.244.196:5060 -> 158.196.244.197:5060 UDP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:371
      DF
Len: 343

[**] [1000100:1000153:1] AD_HIGH_VALUE_OF_DOWNLOAD_UDP_DATA_SPEED [**]
[ Priority : 0]
03/27-21:31:51.012938 00:0C:29:88:96:24 -> 00:0C:29:D2:24:44 type:0x800 len:0x181
158.196.244.196:5060 -> 158.196.244.197:5060 UDP TTL:64 TOS:0x0 ID:0 IpLen:20 DgmLen:371
      DF
Len: 343
```

Výpis 15: Ukázka alertu při detekování anomálie.

Jak již je zmíněno výše, systémy pro detekci anomálií mají tendenci vyvolávat větší množství falešných výstrah. Jelikož moje řešení výchází z monitorování překročení maximální rychlosti downloadu na UDP portu, jednalo se o řádově jednotky falešných výstrah, které byly generovány. Po bližším zkoumání souboru s aktuálním provozem jsem zjistil, že se falešné výstrahy vyskytují na začátku a na konci každého testovaného scénaře (resp. při spuštění a ukončení SNORT), kdy je v rámci jedné sekundy nárazově zvýšena přenosová rychlost a většinou překročena maximální rychlost, při stejném množství přenesených UDP paketů.

4 Realizace výměny informací pomocí vloženého pole v SIP signalizaci

V této části práce se zabývám metodou, která umožňuje vložení tajné (skryté) zprávy do hlavičky v SIP protokolu. V úvodu popisuji základní vlastnosti protokolu SIP a SDP. Dále popisuji hlavičku a tělo protokolu SIP, možnosti pole User-Agent, jaké typy požadavků či metod a odpovědí existují. V následující části se věnuji softwaru pro generování SIP zpráv SIPp a přibližuji, jak pracuje se scénáři, sestavených pro konkrétní simulaci hovoru. V závěru se věnuji implementaci zmíněné problematiky, pro účely této práce.

4.1 SIP

Jedná se o protokol, který vychází z protokolu HTTP. Z toho také vychází jeho jednoduchá implementace a ladění. SIP protokol reprezentuje formu komunikace typu klient-server, kdy dochází k výměně zpráv typu: *požadavek (request)* a *odpověd (response)*. Klient i server je implementován v rámci jednoho prvku. Pokud komunikační strana vystupuje v roli klienta, bývá označovaná termínem *UAC (User Agent Client)*. Pokud komunikační strana vystupuje v roli serveru, bývá označovaná termínem *UAS (User Agent Server)*.

```
INVITE sip:bob@biloxi.com SIP/2.0
    Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bKnashds8
    To: Bob <bob@biloxi.com>
    From: Alice <alice@atlanta.com>;tag=1928301774
    Call-ID: a84b4c76e66710
    CSeq: 314159 INVITE
    Max-Forwards: 70
    Date: Thu, 21 Feb 2002 13:02:03 GMT
    Contact: <sip:alice@pc33.atlanta.com>
    Content-Type: application/sdp
    Content-Length: 351

    v=0
    o=alice 53655765 2353687637 IN IP4 pc33.atlanta.com
    s=Session SDP
    t=0 0
    c=IN IP4 pc33.atlanta.com
    m=audio 3456 RTP/AVP 0 1 3 99
    a=rtpmap:0 PCMU/8000
```

Výpis 16: Příklad metody INVITE SIP hlavičky a těla SDP.

4.1.1 Hlavička

Hlavička protokolu SIP je tvořena mnoha textovými poli, které mají formát *<název>:<hodnota>*. V článku [5] je popsáno, že v RFC 3261 [12] je definováno celkem 115 polí v SIP hlavičce z toho je 6 povinných a 109 volitelných, které lze teoreticky využít pro skrytí tajné informace neboli steganogramu. V RFC 3261 dokumentu je uvedná tabulka, za jakých podmí-

nek lze dané pole využít pro komunikaci. Konkrétně je specifikováno v jakých metodách, požadavcích a odpovědích lze aplikovat dané pole. V kódu (Výpis 16) je zobrazena hlavička metody reprezentována částí začínající řadkem *INVITE* a končící *Content-Length: 351*.

Ve své práci využívám pole *User-Agent*, které lze využít v rámci všech metod, požadavcích a odpovědích specifikovaných v RFC 3261.

4.1.1.1 Pole User-Agent Toto pole je primárně určeno pro identifikaci softwarové verze klienta, který je použit pro komunikaci. Vzhledem k tomu, že pole nemá stanovené žadné pravidla z hlediska obsahu, je jeho využití pro steganografické účely ideální. V rámci RFC 3261 je specifikovaná jeho volitelnost v rámci metod ACKNOWLEDGE, BYE, CANCEL, INVITE, OPTIONS, REGISTER. RFC 2976 specifikuje využití tohoto pole v rámci metody INFO také jako volitelné.

```
User−Agent: Softphone Beta1.5.2
```

Výpis 17: Příklad pole User-Agent v SIP hlavičce.

4.1.2 Tělo

SIP může obsahovat i tělo zprávy, které obsahuje popis medií. Toto tělo je reprezentováno protokolem SDP. Samotné tělo je od hlavičky doděleno volným řádkem (CRLF) [8]. Stejně jako v protoklou SIP i SDP má definována pole: *v*, *o*, *s*, *m*, *t* jako povinná a 15 polí jako velitelná [5]. V kódu (Výpis 16) je zobrazeno tělo metody reprezentované částí začínající řadkem *v=0* a končící *a=rtpmap:0 PCMU/8000*.

4.1.3 Typy požadavků

SIP požadavky představují metody, které lze využít v rámci komunikace. Následující výčet zobrazuje veškeré metody typu požadavek [8]:

- **INVITE** - Metoda pro inicializaci spojení nebo změnu parametrů již probíhajícího spojení.
- **ACK** - Metoda potvrzující konečné přijetí odpovědi na žádost INVITE. K sestavení relace je využito „3-way hand-shaking“, kdy volaný periodicky opakuje odpověď, dokud nepříjme ACK, což indikuje, že odpověď byla doručena.
- **BYE** - Metoda ukončuje sestavené spojení.
- **CANCEL** - Metoda se využívá ke zrušení právě sestavovaného spojení, pokud již není sestaven dialog a volaný nepotvrdil konečnou odpověď (200 OK) na žádost INVITE a volající požaduje zrušení sestavování spojení.
- **OPTIONS** - Jedná se o speciální metodu ke zjištění vlastnosti SIP zařízení (jaká je konfigurace). Tato metoda generuje jen jednu SIP odpověd (200 OK).

- **REGISTER** - Metoda reprezentuje žádost o registraci nebo deregistraci uživatele, kdy se váže logická jmenná adresa uživatele s jeho fyzickým umístěním v síťi (IP adresa a port), konkrétně jde o pole FROM a CONTACT ze SIP hlavičky. Registrace jsou časově limitované a je nutné je periodikcy opakovat.
- **PRACK** - Metoda pro potvrzení dočasné odpovědi (1xx). Posílá předběžnou odpověď na 1xx.
- **SUBSCRIBE** - Metoda slouží k přihlášení k upozornění na události.
- **NOTIFY** - Metoda informuje účastníky o nové události.
- **PUBLISH** - Publikuje událost serveru.
- **INFO** - Počet ICMP paketů v rámci podsítě.
- **REFER** - Metoda indikuje, že příjemce, který je identifikovaný pomocí request-URI, by měl kontakovat třetí stranu pomocí informací uvedených v žádosti.
- **MESSAGE** - Metoda reprezentující transportování zpráv pro instant messaging přes SIP.
- **UPDATE** - Metoda umožňuje aktualizovat stav relace (aktualizovat parametry spojení) beze změny dialogu.

V mé práci jsem využil metody *OPTIONS* a *INFO* z důvodů, že negenerují velké množství dalších SIP zpráv, pouze zprávu 200 OK, což je výhodné pro ukrytí steganogramu, protože zvýšený přenos těchto zpráv by mohl vyvolat podezření, že je v rámci komunikace umístěn skrytý komunikační kanál.

4.1.3.1 OPTIONS

Metoda OPTIONS umožňuje UAC dotázat se jiného UAC nebo proxy serveru, na jeho schopnosti v rámci přenosu. To umožňuje UAC získat informace o podporovaných metodách, typu obsahu, rozšíření, kodecích, bez vyzvánění na straně dotazovaného klienta. Všechny UAS musí podporovat metodu OPTIONS.

4.1.3.2 INFO

Metoda INFO je specifikována v rámci RFC 2976. Využívá se v rámci již probíhající komunikaci (mid-session communication) ve fázi signalizace. Poskytuje informace o cestě signalizace volání. Metoda se nepoužívá ke změně stavu SIP hovoru, ani ke změně stavu relace již zahájeného SIP hovoru. Je určen pro poskytování dalších volitelných informacích, které mohou využít aplikace využívající SIP protokol.

4.1.4 Typy odpovědí

Jsou rozděleny do šesti tříd, podle stavu ve kterém se transakce nachází. Kód odpovědi je celé číslo v intervalu 100 až 699, které značí typ odpovědi. Následující výčet zobrazuje hodnoty, které může daný stav odpovědi nabývat:

- **1xx** - Dočasné informační odpovědi, které jsou odeslány na základě žádosti volajícího, ale výsledek zpracování není aktuálně znám. 100 (Trying) a 180 (Ringing).
- **2xx** - Konečná odpověď s pozitivním výsledkem, je poslední odpověď, která je obdržena na základě žádosti volaného. Reprezentuje výsledek zpracování konkrétní žádosti. 200 (OK).
- **3xx** - Označuje odpovědi, které slouží k přesměrování (odpověď od redirect serveru). Tyto odpovědi dávají informaci o nové poloze uživatele nebo alternativní službě, která má být použita.
- **4xx** - Konečná odpověd s negativním výsledkem, které výpovídají, že problém je na straně klienta.
- **5xx** - Konečná odpověd s negativním výsledkem, které výpovídají, že problém je na straně serveru. Server selhal při zpracování, zároveň žádost je v pořádku.
- **6xx** - Představuje situaci, kdy nastala globalní chyba. Kód je vysílán pokud žádost nemůže být zpracována na žádném serveru.

4.2 SIPp

SIPp [23] je open source nástroj pro testování a generování síťového provozu pro protokol SIP. Dokáže generovat několik set horovů za sekundu v rámci jedné instance. Množství generovaných hovorů se dá jednoduše nastavovat. SIPp pro sestavení a nastavení hovoru využívá scénaře ve formátu .XML. Existují dva typy scénařů: UAC scenáře, které se využívají na straně klienta a UAS scénaře, které se využívají na straně serveru. Scenáře mohou být velmi jednoduché, ale i velmi komplexní a složité. SIPp během probíhající komunikace dynamicky zobrazují statistiky (call rate, delay, message statistics, TCP a UDP přes mutliple socket).

Mezi pokročilé funkce patří podpora protokolu IPv6, TLS, SCTP, SIP authentication, podmíněné scénáře, UDP retransmission, chyby robustnosti (call timeout, protocol defense), posilání médií (RTP) pomocí RTP echo a RTP pacp replay.

Následujcí výpis (Výpis 18) z konzole, zobrazuje probíhající SIPp komunikaci ze strany klienta (UAC). Konkrétně se jedná o první instanci, kdy je generováno 10 hovorů za sekundu, jak lze vidět v hodnotě *Call-rate*. Dále je zobrazeno *číslo portu*, na kterém komunikace probíha, celková délka hovoru reprezentovaná položkou *Total-time*, celkový počet hovorů reprezentovaný položkou *Total-calls* a IP adresu, port a protokol serveru reprezentovaný položkou *Remote-host*. Můžeme zde také vidět průběh hovoru (call flow), který odpovídá spuštěnému scénáři. Ve sloupci *Messages* vidíme celkový počet zpráv odpovídacíjích dané metodě či zprávě.

Následujcí výpis (Výpis 19) z konzole zobrazuje probíhající SIPp komunikaci ze strany serveru (UAS). Rovněž se jedná o první instanci, kdy je generováno 10 hovorů za sekundu. Jednotlivé položky jsou obdobné, jako v případě UAC, kde jsou podrobněji popsány.

```
Call−rate(length)   Port   Total−time  Total−calls  Remote−host
10.0(0 ms)/1.000s   5060   75154.08 s       751540  158.196.244.197:5060(UDP)

10 new calls during 1.002 s period      1 ms scheduler resolution
0 calls  ( limit  30)                   Peak was 3 calls,  after  83 s
0 Running, 332 Paused, 23 Woken up
0 dead call  msg (discarded)            0 out−of−call msg (discarded)
3 open sockets
                                 Messages Retrans  Timeout  Unexpected−Msg
     INVITE −−−−−−−−−−−> 751540  4         0
        100 <−−−−−−−−−−− 751540  0         0        0
        180 <−−−−−−−−−−− 751540  0         0        0
        200 <−−−−−E−RTD1 751540  0         0        0

        ACK −−−−−−−−−−−> 751540  0
    OPTIONS −−−−−−−−−−> 751540  0
        200 <−−−−−−−−−−− 751540  0         0        0

       INFO −−−−−−−−−−−> 751540  0
        200 <−−−−−−−−−−− 751540  0         0        0

        BYE −−−−−−−−−−−> 751540  0
        200 <−−−−−−−−−−− 751540  0         0        0
−−−−−− [+|−|∗|/]: Adjust rate −−−− [q]: Soft exit −−−− [p]: Pause traffic −−−−−
```

Výpis 18: SIPp UAC průběh komunikace

```
−−−−−−−−−−−−−− Scenario Screen −−−−−−−−− [1−9]: Change Screen −−
Port   Total−time  Total−calls  Transport
5060   75664.54 s       756443  UDP

10 new calls during 1.002 s period      1 ms scheduler resolution
0 calls                                 Peak was 3 calls,  after  124 s
0 Running, 331 Paused, 13 Woken up
0 dead call  msg (discarded)
3 open sockets
                                 Messages Retrans  Timeout  Unexpected−Msg
−−−−−−−−−−−> INVITE     756443  0         0        0

<−−−−−−−−−−− 100         756443  0
<−−−−−−−−−−− 180         756443  0
<−−−−−−−−−−− 200         756443  0
−−−−−−−−−−−> ACK E−RTD1 756443  0         0        0

−−−−−−−−−−−> OPTIONS    756443  0         0        0
<−−−−−−−−−−− 200         756443  0
−−−−−−−−−−−> INFO       756443  0         0        0
<−−−−−−−−−−− 200         756443  0
−−−−−−−−−−−> BYE        756443  0         0        0
<−−−−−−−−−−− 200         756443  0
−−−−−−−−−−−−−− Sipp Server Mode −−−−−−−−−−−−−−−−
```

Výpis 19: SIPp UAS průběh komunikace

4.2.1 Scénáře

SIPp využívá tzv. scénaře, ve kterých je popsána SIP komunikace. Scénáře obsahují jednotlivé metody a pole s hodnotami. Ve scénáři pro UAC jsou popsány i odpovědi (response) UAS a naopak. K popisu komunikace se používá jazyk XML. Jak lze vidět v následujícím (zkráceném) kódu scénáře (Výpis 20), jednotlivým polím můžeme přiřadit konkretní hodnotu, nebo je lze nahradit proměnnými, kde jsou hodnoty generovány pomocí SIPp. Scénář CLIENT-NOSTEG.xml umožňuje vykonat velké množství simultáních hovorů. Ve scénářích lze přímo definovat parametry spojení např. délka pauzy, nicméně většina parametru lze nastavit přímo při spuštění instance.

```
<?xml version="1.0" encoding="ISO−8859−1" ?>
<!DOCTYPE scenario SYSTEM "sipp.dtd">
<scenario name="CLIENT−NOSTEG">
  <send retrans="500">
    <![CDATA[
      INVITE sip:[service]@[remote_ip]:[remote_port] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      From: sipp <sip:sipp@[local_ip]:[local_port]>;tag=[pid]SIPpTag09[call_number]
      To: sut <sip:[service]@[remote_ip]:[remote_port]>
      Call−ID: [call_id]
      CSeq: 1 INVITE
      Contact: sip:sipp@[local_ip]:[local_port]
      Max−Forwards: 70
      Subject: Performance Test
    ]]>
  </send>
  <recv response="100" optional="true">
  </recv>
  <recv response="180" optional="true">
  </recv>
  <recv response="200" rtd="true" crlf="true">
  </recv>
    <ResponseTimeRepartition value="10,␣20,␣30,␣40,␣50,␣100,␣150,␣200"/>
  <CallLengthRepartition value="10,␣50,␣100,␣500,␣1000,␣5000,␣10000"/>
</scenario>
```

Výpis 20: Příklad SIPp scénáře (metoda INVITE).

4.3 Implementace

Pro prozkoumání, do jaké míry lze injektovat SIP hlavičky a vložit tajnou zprávu, jsem vytvořil komunikaci mezi dvěmi servery. První server s IP adresou *.196* představoval klienta UAC a server s IP adresou *.197* představoval server UAC.

Na straně serveru UAS jsem spustil dvě instance SIPp klienta z důvodů simulace, která by se přibližovala reálným podmínkám provozu. První instance komunikovala s UAC na straně klienta na portu 5060. Druhá instance měla stejný scénář i parametry jako první instance UAS s rozdílem, že komunikace s druhou instancí UAC probíhala na portu 5061.

```
sudo sipp 158.196.244.197 -sf SERVER.xml -s unlimited -trace_err -max_recv_loops 10000 -
    watchdog_minor_maxtriggers 12000
```

Výpis 21: Příkazy pro spuštění 1. a 2. instance SIPp UAS

Pro korektní spuštění všech instancí je potřeba dodržet následující postup při spoutění:

1. První instance UAS (port: 5060)
2. První instance UAC (port: 5060)
3. Druhá instance UAS (port: 5061)
4. Druhá instance UAC (port: 5061)

```
sudo sipp 158.196.244.197 -sf SERVER.xml -s unlimited -trace_err -max_recv_loops 10000 -
    watchdog_minor_maxtriggers 12000
```

Výpis 22: Příkaz pro spuštění 1. a 2. instance SIPp UAS

Na klientské části jsem spustil také dvě instance SIPp klienta. V první instanci jsem nastavil následující parametry: 10 simultaních hovorů, kde maximální limit hovorů ve špičkách je 30, neinjektovaný scénář, nastavení maximalího počtu přijatých zpráv v jednom cyklu (výchozí hodnota: 1000), kde vyšší hodnota se nastavuje při vysokém provozu a hodnota kolikrát může být watchdog spuštěn před ukončením testu (výchozí hodnota: 120).

```
sudo sipp 158.196.244.197 -sf CLIENT-NOSTEG.xml -trace_err -l 30 -r 10 -max_recv_loops
    10000 -watchdog_minor_maxtriggers 12000
```

Výpis 23: Příkaz pro spuštění 1. instance SIPp UAC

Druhá instance obsahovala stejné nastavení s tím rozdílem, že počet simultaních hovorů jsem nastavil na hodnotu 1, maximalní limit hovorů ve špičce na hodnotu 3 a specifikace portu pro komunikaci s UAS serverem na hodnotou 5061.

```
sudo sipp 158.196.244.197:5061 -sf CLIENT-NOSTEG.xml -trace_err -l 3 -r 1 -
    max_recv_loops 10000 -watchdog_minor_maxtriggers 12000
```

Výpis 24: Příkaz pro spuštění 2. instance SIPp UAC

Po spustění všech instancí jsem zároveň spustil na straně serveru SNORT s preprocesorem Anomaly Detection. Následně jsem po zaznamenání provozu (vytvoření modelu chování) sítě zvolil predikovaný model, podle algoritmu, který jsem chtěl aktuálně zkoumat.

Vytvořil jsem sadu testovacích scénářů (detailnější popis níže), pomocí kterých jsem ověřoval schopnost detekce anomálie pomocí daného predikovaného algoritmu.

Testovací scénáře se steganogramem, jsem spouštěl v rámci druhé instance na klientské straně s frekvencí 1 nový hovor za sekundu. Všechny ostatní instance zůstaly nezměněny, a měly parametry příkazu zobrazené výše.

4.3.1 Neinjektované scénáře

Neinjektované scénáře reprezentují soubory CLIENT-NOSTEG.xml, určený pro klinetskou komunikační část a soubor SERVER.xml, ktery je určen pro serverovou komunikační část. Tyto soubory neobsahují steganogram samotný, ale pouze prázdné pole *User-Agent* narozdíl od injektovaných scénářů. Vytvořil jsem scénáře, které obsahují metody: INVITE, ACK, OPTIONS, INFO, BYE.

Tyto scénáře jsem využíval k záznamu síťového provozu, který mi poté sloužil jako referenční model.

```
<?xml version="1.0" encoding="ISO−8859−1" ?>
<!DOCTYPE scenario SYSTEM "sipp.dtd">
<scenario name="CLIENT−NOSTEG">
  <send>
    <![CDATA[
      OPTIONS sip:[service]@[remote_ip] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      Max−Forwards: 70
      To: <sip:[service]@[remote_ip]>
      From: sipp <sip:sipp@[local_ip]:[local_port]>;tag=[call_number]
      Call−ID: [call_id]
      CSeq: 1 OPTIONS
      Contact: <sip:sipp@[local_ip]:[local_port]>
      User−Agent:
    ]]>
  </send>
  <recv response="200" crlf="true">
  </recv>
    <ResponseTimeRepartition value="10,␣20,␣30,␣40,␣50,␣100,␣150,␣200"/>
  <CallLengthRepartition value="10,␣50,␣100,␣500,␣1000,␣5000,␣10000"/>
</scenario>
```

Výpis 25: Neinjektovaný SIPp scénář (metoda OPTIONS).

4.3.2 Injektované scénáře

Injektované scénáře jsou prakticky stejné s tím rozdílem, že v metodě OPTIONS jsem umístil do pole User-Agent, steganogram reprezentovaný textem o délce 100 až 60000 znaků. Jednotlivé injektované scénáře se liší v počtu injektovaných znaků. Použitý text je generován náhodně. Zdrojový kód (Výpis 26) reprezentuje scénář *STEG-2.xml*, který obsahuje 500 injektovaných znaků.

```
<?xml version="1.0" encoding="ISO−8859−1" ?>
<!DOCTYPE scenario SYSTEM "sipp.dtd">
<scenario name="CLIENT−STEG−2">
  <send>
    <![CDATA[
      OPTIONS sip:[service]@[remote_ip] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      Max−Forwards: 70
      To: <sip:[service]@[remote_ip]>
```

```
      From: sipp <sip:sipp@[local_ip]:[local_port]>;tag=[call_number]
      Call-ID: [call_id]
      CSeq: 1 OPTIONS
      Contact: <sip:sipp@[local_ip]:[local_port]>
      User-Agent: Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo
          ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient
          montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium
           quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec
          , vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo.
          Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibu
    ]]>
  </send>
  <recv response="200" crlf="true">
  </recv>
    <ResponseTimeRepartition value="10,␣20,␣30,␣40,␣50,␣100,␣150,␣200"/>
  <CallLengthRepartition value="10,␣50,␣100,␣500,␣1000,␣5000,␣10000"/>
</scenario>
```

Výpis 26: Injektovaný SIPp scénář (metoda OPTIONS).

4.3.3 Sada testovacích scénářů

Testovací sada scénářů se skládá s celkem deseti injektovaných scénářů a jednoho scénáře neinjektovaného, který sloužil k tvorbě referenčního modelu. Počet injektovaných znaků v jednotlivých scénařích jsem zvyšoval postupně. V následující tabulce jsou zobrazeny jednotlivé scénáře s počtem injektovaných znaků. Dále lze porovnat celkovou velikost scénáře (Byte) a velikost SIP OPTIONS zprávy (Byte), kterou jsem zjistil pomocí software WireShark.

	Počet injektovaných znaků	Velikost souboru (Byte)	Velikost SIP OPTIONS zprávy (Byte)
NOSTEG.xml	0	4769	325
STEG-1.xml	100	4869	425
STEG-2.xml	500	5269	825
STEG-3.xml	1000	5769	1325
STEG-4.xml	5000	9769	5325
STEG-5.xml	10000	14769	10325
STEG-6.xml	20000	24769	20325
STEG-7.xml	30000	34769	30325
STEG-8.xml	40000	44769	40325
STEG-9.xml	50000	54769	50325
STEG-10.xml	60000	64769	60325

Tabulka 6: Testované scénáře s velikostí souboru a SIP zprávy

5 Využití RTP pro přenos textových informací

V této části popisuji novou metodu pro přenos tajné textové informace pomocí RTP protokolu. Konkrétně se jedná o využití informací získaných z payloadu jednotlivých RTP paketů. Jelikož jsou veškerá data v digitální technice reprezentována a přenášena v binárním kódu, je zde velký potenciál pro jejich využití. V rámci RTP je přenášeno velké množství paketů, které představují velké množství dat při typické délce hovoru. V úvodu se věnuji RTP a RTCP protokolu, kodeku G.711, ASCII tabulkce reprezentující znakové symboly. Poté popisuji princip této metody, přenosovou kapacitu skrytého kanálu, předpoklady pro její realizaci, převod znaků do binární podoby, detekci v RTP přenosu, pointer v SIP hlavičce, kódování pointeru. V závěru popisuji princip přenosu informací během hovoru.

5.1 RTP (Real-time Transport Protocol)

Jak je již popsáno v části (Úvod do VoIP a steganografie), je binární protokol RTP postaven na protokolu UDP. Navíc obsahuje nové vlastnosti pro zajištění vyšší kvality multimediálního přenosu. Pro přenos nepoužívá fixní číslo portu. Port je alokován dynamicky při sestavování pro přenos média. Číslo alokovaného portu je zasláno druhé komunikační straně pomocí protokolu SDP.

Komunikace RTP protokolu je jednosměrná, z toho vyplývá nutnost použití dvou RTP toků pro hlasovou komunikaci a čtyř RTP toků v rámci videohovoru.

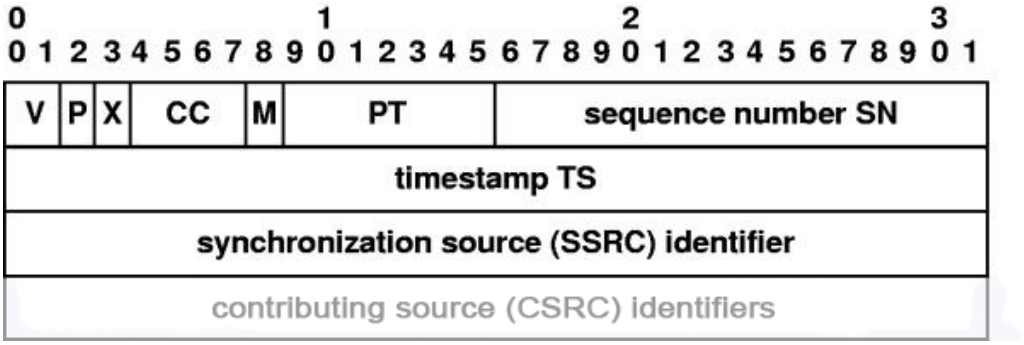
RTP protokol zasílá velké množství paketů v pravidelných intervalech (typicky 20 ms). Velikost a frekvence odesílání paketu se odvíjí od typu použitého kodeku a požadavků na kvalitu přenosu. V rámci této práce budu využívat kodek *G.711*.

Vzhledem k tomu, že je RTP navržený k co největší úspoře přenosového pásma, obsahuje hlavička protokolu pouze nezbytné údaje. Ostatní režijní data jsou zasílana pomocí jiných přenosových kanálů. Tyto přenosové kanály reprezentují protokoly SIP, SDP a RTCP.

5.1.1 Hlavička

Základní verze hlavičky RTP protokolu má velikost 12 B. Hlavička obsahuje následující pole:

- **V (Version)** - Značí jaká verze RTP je pro přenos použita. Aktualně se jedná o hodnotu 2.
- **P (Padding)** - Pokud je hodnota nastavena na 1, to značí, že je přenášena i patička (oktety). Poslední oktet obsahuje informaci o tom, kolik oktetů bylo celkem přidáno. Konkrétně se jedná se o doplnění nulami.
- **X (xEtension)** - Pokud je hodnota nastavena, značí to použití rozšířené hlavičky tzn., že za pevnou hlavičkou následuje právě jedno rozšíření s definovaným formátem.



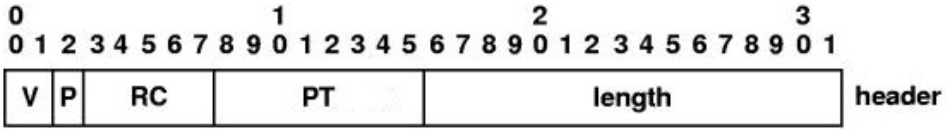
Obrázek 6: Formát hlavičky RTP paketu

- **M (Marker)** - V rámci hlasových paketů označuje začátek talkspurt (spřádání hovoru), kde se jedná o korekci zpoždění.
- **CC (CSRC count)** - Obsahuje počet (délku seznamu zdrojů) CSRC za základní hlavičkou.
- **PT (Payload Type)** - Popisuje o jaký typ dat se jedná v rámci přenášeného paketu. Má délku 7-bitů, a může tak nabývat hodnoty v rozsahu 0-127. Definovány jsou i některé statické hodnoty. Např. G.711 μ-Law reprezetuje hodnota 0, G.711 A-Law reprezentuje hodnota 8 a G.729 reprezentuje hodnota 18. Interval hodnot 96-127 je rezervován pro dynamický typ.
- **SN (Sequence Number)** - sekvenční číslo, začínající náhodnou hodnotou, které je inkrementováno o 1, při každém odeslaném RTP paketu. Této vlastnosti se využívá k detekci paketů, které jsou mimo pořadí.
- **TS (TimeStamp)** - Slouží k synchronizaci přehrávání, hodnota vzrůstá v závislosti na vzorkovací frekvenci přenášeného média.
- **SSRC (Synchronization Source)** - Obsahuje 32-bitový identifikátor zdroje dat (typ zdroje synchronizace, identifikace uživatele), který je náhodně zvolen stranou, která zahájila konverzaci.
- **CSRC (Contributing Source)** - Obsahuje 32-bitový identifikátor zdroje dat, které přispívají svým obsahem do paketu.

5.2 RTCP (Real-Time Control Protocol)

Jak je již popsáno v úvodní části (Úvod do VoIP a steganografie), jedná se o protokol určený k řízení toku dat, který doplňuje protokol RTP o poskytování zpětné vazby určující kvalitu služeb (QoS) poskytovanou RTP.

RTCP protokol podobně jako protokol RTP, nemá fixně definované číslo portu. V rámci konvence platí jednoduché pravidlo, pro definování portu na základě RTP portu.



Obrázek 7: Formát hlavičky RTCP paketu

Pokud RTP protokol komunikuje na portu *p*, pak RTCP využívá pro komunikaci port *p+1*. RTP protokol pro komunikaci využívá sudá čísla portů, narozdíl od RTCP protokolu, který využívá porty reprezentující lichá čísla. RTCP poskytuje nasledcí funkce:

- poskytování informací o kvalitě hovoru (jitter, feedback, latence, počet odeslaných Byte, počet odeslaných paketů a počet ztracených paketů),
- detekce celkového množství účastníků komunikace,
- kontrola proti zalhcení sítě,
- poskytování dalších informací o spojení.

5.2.1 Hlavička

RTCP protokol má základní (počáteční) hlavičku o délce 8 B. Obsahuje pole V, P, X, RC, PT, length. Pole *V* definuje verzi protokolu (typicky s hodnotou 2), pole *P* značí informaci o použití patičky, pole *RC* reprezentuje počet reciever portů, pole *PT* reprezentuje typ zprávy a pole *length* reprezentuje délku zprávy. Tuto základní hlavičku využívají zprávy RTCP protokolu.

5.2.2 Typy zpráv

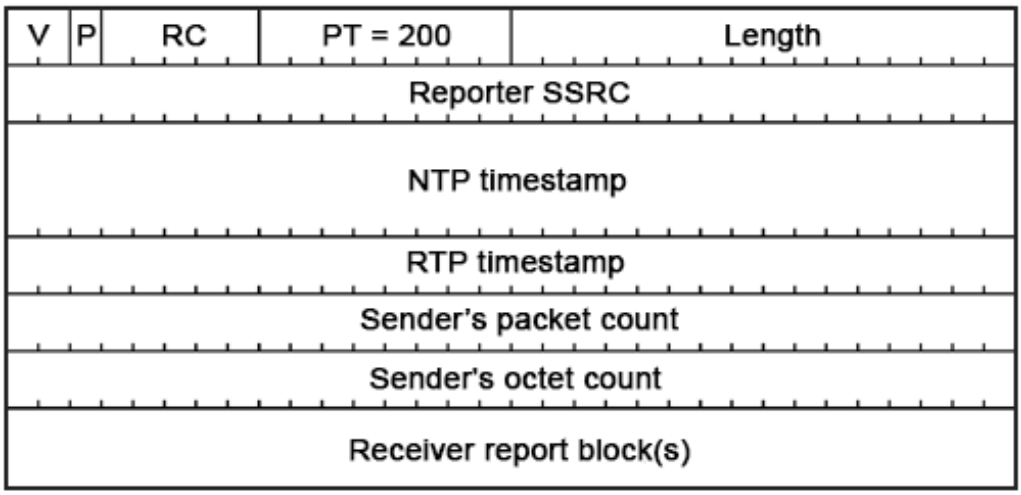
RTCP protokol umožňuje straně, která je v roli přijemce, odeslat různé zprávy straně, která je v roli odesílatele. V rámci RTCP je definováno pět typů zpráv.

RTCP umožňuje také spojení více typů zpráv do jednoho paketu, z důvodu úspory nároků na řežii komunikace. Takovému paketu se říká tzv. *složený paket*.

Složený paket nevyužívá žádné oddělovače. Zprávy jsou naskládány za sebe, a poté jsou detekovány na základě informace o délce zprávy umístěné v hlavičce.

5.2.2.1 SR (Sender Report) Sender Report zprávy obsahují informace o právě probíhající komunikaci a také přijímací statistiky v rámci všech zaslaných paketů RTP. Tyto informace jsou velmi důležité pro analyzování kvality hovoru (resp. celého spojení).

Zpráva obsahuje pole *SSRC of sender*, které identifikuje původ dat, pole *NTP timestamp - most significant word, NTP timestamp - least significant word, RTP timestamp* reprezentující časové razítka pro synchronizaci přijemců, pole *sender's packet count* reprezentuje počet odeslaných paketů a pole *sender's octet count* reprezentuje počet odeslaných Byte.



Obrázek 8: RTCP - SR (Sender Report) zpráva.

5.2.2.2 RR (Receiver Report) Tyto zprávy jsou určeny pro pasivní účastníky komunikace tzn., že neodesílají žadné RTP pakety. Všichni účastníci s tímto statusem tyto zprávy odesílají. Zpráva informuje odesílatele a další příjemce o kvalitě spojení.

Zpráva Receiver Report obsahuje totožná pole jako zpráva Sender Report s tím rozdílem, že pole PT obsahuje číslo 201 a pět polí, které poskytují informace o odesílateli jsou vynechána (NTP a RTP timestamps, sender's packet count, sender's octet count).

5.2.2.3 SDES (Source Description) Je zpráva, která nese informace o zdroji RTP dat a ty dále poskytuje ostatním účastníkům komunikace. Odesílaní zprávy se periodicky opakuje. Zpráva se skládá z hlavičky a bloků (nula a více). Každý blok je složen z položek popisujících zdroje identifikované v tomto bloku. Popis zdroje se skládá z pole *SSRC/CSRC*, které reprezentuje identifikátor zdroje a jakéhokoliv počtu polí *SDES items*. Jednotlivé položky začínají polem *type*, které je popsáno 8-bitovým kódem a polem *length*, které reprezentuje délku.

Pole *type* může obsahovat následující typy informací [25]:

- **CNAME** - kanoické jméno,
- **NAME** - jméno uživatele,
- **EMAIL** - emailová adresa,
- **PHONE** - telefonní číslo,
- **LOC** - geografická poloha,
- **TOOL** - název aplikace, která generuje RTP provoz,

- **NOTE** - zpráva popisující současný stav strany odesílající data,
- **PRIV** - aplikační a experimentální rozšíření,
- **END** - konec SDES seznamu.

5.2.2.4 BYE (Goodbye Message) Je zpráva k ukončení streamu dat, kterou odesílá strana odesílající data. Tato zpráva oznamuje ostatním účastníkům komunikace, že strana, která zprávu BYE odeslala, opouští konferenci.

5.2.2.5 APP (Application-Specific Message) Typ této zprávy nemá definován fixní význam ve standardu, a tím umožňuje definici nových zpráv. APP pakety se také používají k experimentálním účelům.

5.3 G.711

Protokol G.711 je jeden z nejpoužívanějších kodeků u v oblasti internetové telefonie. Vzorkovací frekvence 8 kHz a kvantifikování do 8-bitů, vytváří tok 64kbps. Pro kódování signálu je využívána logaritmická komprese, kde je dvanácti nebo třinácti bitový signál převeden na osmibitový. Logaritmická komprese v Evropě a Austrálii využívá jiný vzorec A-law, narozdíl od severní Ameriky a Japonska, kde se používá vzorec μ-law, který přináší vyšší kompresi z důvodu, že osmý bit je používán pro přenos signalizace a sedm bitů pro přenos hlasu. V Evropě se přenáší signalizace samostatným kanálem a hovor je přenášen v rámci všech osmi bitů. Důsledkem toho je, že kvalita hovoru je v Evropě a Austrálii vyšší. Frekvence paketizace je u kodeku G.711 20 ms.

5.4 ASCII

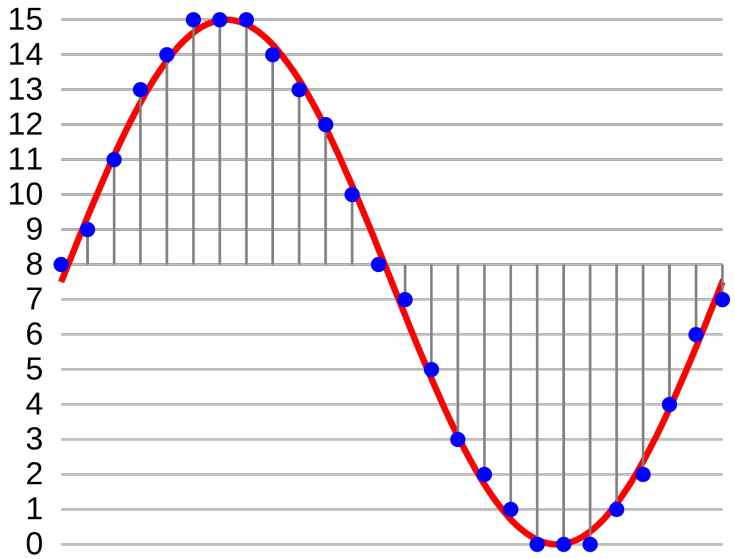
ASCII (American Standard Code for Information Interchange) je kódovací tabulka pro kódování alfanumerických a speciálních znaků. Je historicky nejvyužívanější znakovou sadou, která je základem většiny současných standardů pro kódování znaků.

Tabulka obsahuje znaky:

- **tisknutelné** - písmena (anglické abecedy), číslice, matematické znaky, interpunkční znaménka a speciální znaky,
- **netisknutelné** - řídící kódy původně určené pro řízení periferních zařízení (tiskárny, dálnopisy apod.).

Původní tabulka ASCII je 7-bitová, čímž dokáže zakódovat *128 znaků*. Z důvodu potřeby kódování ostatních jazyků vznikla i její 8-bitová verze, která dokáže zakódovat až 256 znaků. Avšak ani toto množství není dostačující pro kódování národních abeced, proto používají jiné kódovací tabulky navržené organizací ISO.

V rámci této práce využívám 7-bitovou ASCII tabulku, která je pro kódování základních anglických alfanumerických znaků dostačující.



Obrázek 9: Kvantizace signálu do 4-bitů

Decimal - Binary - Octal - Hex – ASCII
Conversion Chart

Decimal	Binary	Octal	Hex	ASCII	Decimal	Binary	Octal	Hex	ASCII	Decimal	Binary	Octal	Hex	ASCII	Decimal	Binary	Octal	Hex	ASCII
0	00000000	000	00	NUL	32	00100000	040	20	SP	64	01000000	100	40	@	96	01100000	140	60	`
1	00000001	001	01	SOH	33	00100001	041	21	!	65	01000001	101	41	A	97	01100001	141	61	a
2	00000010	002	02	STX	34	00100010	042	22	"	66	01000010	102	42	B	98	01100010	142	62	b
3	00000011	003	03	ETX	35	00100011	043	23	#	67	01000011	103	43	C	99	01100011	143	63	c
4	00000100	004	04	EOT	36	00100100	044	24	$	68	01000100	104	44	D	100	01100100	144	64	d
5	00000101	005	05	ENQ	37	00100101	045	25	%	69	01000101	105	45	E	101	01100101	145	65	e
6	00000110	006	06	ACK	38	00100110	046	26	&	70	01000110	106	46	F	102	01100110	146	66	f
7	00000111	007	07	BEL	39	00100111	047	27	'	71	01000111	107	47	G	103	01100111	147	67	g
8	00001000	010	08	BS	40	00101000	050	28	(	72	01001000	110	48	H	104	01101000	150	68	h
9	00001001	011	09	HT	41	00101001	051	29	)	73	01001001	111	49	I	105	01101001	151	69	i
10	00001010	012	0A	LF	42	00101010	052	2A	*	74	01001010	112	4A	J	106	01101010	152	6A	j
11	00001011	013	0B	VT	43	00101011	053	2B	+	75	01001011	113	4B	K	107	01101011	153	6B	k
12	00001100	014	0C	FF	44	00101100	054	2C	,	76	01001100	114	4C	L	108	01101100	154	6C	l
13	00001101	015	0D	CR	45	00101101	055	2D	-	77	01001101	115	4D	M	109	01101101	155	6D	m
14	00001110	016	0E	SO	46	00101110	056	2E	.	78	01001110	116	4E	N	110	01101110	156	6E	n
15	00001111	017	0F	SI	47	00101111	057	2F	/	79	01001111	117	4F	O	111	01101111	157	6F	o
16	00010000	020	10	DLE	48	00110000	060	30	0	80	01010000	120	50	P	112	01110000	160	70	p
17	00010001	021	11	DC1	49	00110001	061	31	1	81	01010001	121	51	Q	113	01110001	161	71	q
18	00010010	022	12	DC2	50	00110010	062	32	2	82	01010010	122	52	R	114	01110010	162	72	r
19	00010011	023	13	DC3	51	00110011	063	33	3	83	01010011	123	53	S	115	01110011	163	73	s
20	00010100	024	14	DC4	52	00110100	064	34	4	84	01010100	124	54	T	116	01110100	164	74	t
21	00010101	025	15	NAK	53	00110101	065	35	5	85	01010101	125	55	U	117	01110101	165	75	u
22	00010110	026	16	SYN	54	00110110	066	36	6	86	01010110	126	56	V	118	01110110	166	76	v
23	00010111	027	17	ETB	55	00110111	067	37	7	87	01010111	127	57	W	119	01110111	167	77	w
24	00011000	030	18	CAN	56	00111000	070	38	8	88	01011000	130	58	X	120	01111000	170	78	x
25	00011001	031	19	EM	57	00111001	071	39	9	89	01011001	131	59	Y	121	01111001	171	79	y
26	00011010	032	1A	SUB	58	00111010	072	3A	:	90	01011010	132	5A	Z	122	01111010	172	7A	z
27	00011011	033	1B	ESC	59	00111011	073	3B	;	91	01011011	133	5B	[	123	01111011	173	7B	{
28	00011100	034	1C	FS	60	00111100	074	3C	<	92	01011100	134	5C	\	124	01111100	174	7C	|
29	00011101	035	1D	GS	61	00111101	075	3D	=	93	01011101	135	5D	]	125	01111101	175	7D	}
30	00011110	036	1E	RS	62	00111110	076	3E	>	94	01011110	136	5E	^	126	01111110	176	7E	~
31	00011111	037	1F	US	63	00111111	077	3F	?	95	01011111	137	5F	_	127	01111111	177	7F	DEL

Obrázek 10: ASCII převodní tabulka.

5.5 Princip metody

Veškerá datová komunikace v digitalních sítích je realizovaná přenosem informací, které jsou popsány v binární soustavě. To představuje obrovské množství nul a jedniček, které jsou za sebou přenášeny. Vzhledem k tomuto velkému množství dat, je vysoká pravděpodobnost, že nalezneme sekvenci, pomocí které jsme schopni následnou informaci interpretovat v námi požadované podobě. Takové sekvence jsme schopni nalézt také v telefonní komunikaci využívající IP protokol. Největší množství dat, které můžeme využít, představuje samotný datový stream, který slouží pro přenos hlasu.

Pro tento účel je navržený RTP protokol, který zajišťuje přenos mediálních dat. RTP přenáší v rámci každého paketu *160 B* dat, které slouží k popisu hlasu, tzv. payload. Při použití kodeku G.711 jsou pakety generovány každých *20 ms*.

Při hovoru o délce 5 minut máme potencál *2400000* sekvencí, které můžou reprezentovat námi požadovaný znak.

Pro kódování dat můžeme použít již vytvořené tabulky, které jsou mezinárodně standardizované, nebo si můžeme vytvořit vlastní tabulku, kterou ovšem budeme muset poskytnout druhé straně, které chceme steganogram přenést. Pokud budeme uvažovat, že chceme popsat celou 7-bitovou ASCII tabulku, tak to znamená, že potřebujeme najít 128 unikátních sekvencí, které jsou reprezentovány v binárním kódu. Každý znak z ASCII tabulky lze popsat 7-bity. Pro větší přehlednost používám v rámci mé práce 8 bitů s tím, že 1. bit je reprezentován hodnotu binární 0. Pokud bych chtěl dále redukovat počet znaků, které využiji, např. alfanumerikcé znaky vč. malé abecedy, tak mi bude stačit 62 unikátních sekvencí. Pro oddělení a přesnou identifikaci polohy sekvenci je potřeba si dále alokovat 4 speciální znaky.

Pro identifikaci paketu, ve kterém se potřebné sekvence nalézají, využiji *Sequence number*, které poskytuje RTP protokol a je zároveň unikátní, což jednoznačně určuje polohu dat. Jednotlivé sekvence, které odpovídají námi požadovanému znaku, vyhledáváme v payloadu daného paketu a zaznamenáváme informaci o jejich poloze. V případě že se daná sekvence v paketu nenalézá, využijeme paket jiný.

Jelikož musíme předat informaci, o který paket se jedná a poloze sekvence v payloadu druhé straně, využijeme hlavičku protokolu SIP, kde umístíme informace o paketu a o poloze v rámci payloadu daného paketu.

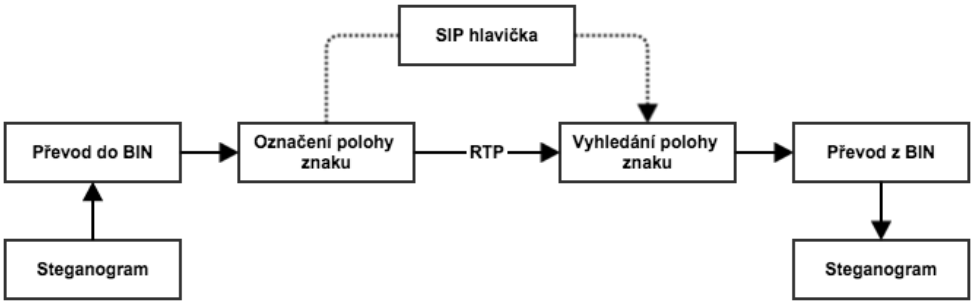
Po přenesení informací ze SIP hlavičky, již může druhá strana začít vyhledávat konkrétní polohy sekvencí reprezentující jednotlivé znaky za předpokladu, že druhá strana disponuje kódovací tabulkou.

Po dekódování binárních sekvencí pomocí kódovací tabulky se dá již sestavit steganogram, odeslaný prvním účastníkem resp. odesílatelem.

5.6 Přenosová kapacita skrytého kanálu

Přenosová kapacita skrytého kanálu je dána kodekem použitým pro kódování hlasu, frekvencí generování paketů, velikostí payloadu v rámci jednoho RTP paketu a délkou hovoru. V rámci této práce využívám kodek G.711 s logaritmickou kompresí A-law.

Přenosovou kapacitu skrytého kanálu vypočítáme podle následujícího vzorce (11.):



Obrázek 11: Schéma steganografické metody

$$CS = \frac{1000}{PF} \times PC \times t[Byte] \tag{11}$$

kde CS je kapacita skrytého kanálu, PF je frekvence generování paketů [ms], PC je velikost payloadu v rámci jednoho paketu [Byte] a t je délka hovoru [sec].

Po dosazení do vzorce nám vyjde hodnota $8000B$ za jednu sekundu. To znamená, že máme potenciál *8000* sekvencí za sekundu, které mohou reprezentovat námi požadovaný znak. Při hovoru o délce 5 minut máme k dispozici již *2400000* sekvencí.

Je třeba mít na paměti, že ne všechny sekvence budou použitlené pro řešení popsané v této práci. Nicméně při optimalizaci převodní tabulky se dá využitelná kapacita zvýšit.

Dalším faktorem, který ovlivňuje využití potenciálu kvantizace do 8-bitů, je využití 8-bitové kódovací tabulky, která dokáže asociovat všech 256 znaků. Ve své práci využívám 7-bitovou tabulku ASCII, z důvodu vytvoření lepší představy o tom, jak tato metoda funguje.

5.7 Předpoklady pro realizaci

Při využití kodeku G.711, který je také znám jako PCM, což je označení pro pulsně kódovou modulaci, probíha kvantizace do 8 bitů, tzn. že jeden vzorek (sample) může mít hodnoty od 0 do 255, nebo v binární soustavě od 00000000 do 11111111. Tyto kvantizační kroky popisují amplitudu v daném vzorku. Proto, abychom v payloadu dostali co nejrozmanitější množinu sekvencí je vhodné, aby zdrojový signál byl foneticky rozmanitý. Poté se pravděpodobnost výskytu různých sekvencí zvýší.

5.8 Převod znaků do binární soustavy

Převod znaků do binární soustavy je triviální úkol s využitím kódovací tabulky ASCII. Pokud budu chtít převést řetězec např. *STEGANOGRAM2015*, tak jednotlivým znakům bude odpovídat binární reprezentace uvedená v následující tabulce:

Poté co jsou znaky převedeny na binarní sekvence, je nutné tyto sekvence nalézt v jednotlivých RTP paketech v oblasti payload.

Znak	Binární reprezentace
S	01010011
T	01010100
E	01000101
G	01000111
A	01000001
N	01001110
O	01001111
G	01000111
R	01010010
A	01000001
M	01001101
2	00110010
0	00110000
1	00110001
5	00110101

Tabulka 7: Znaky s odpovídající binární reprezentací (ASCII)

5.9 Detekce sekvence v RTP paketu

Pro detekování požadované sekvence v paketu potřebujeme packet sniffer, který nám umožní sledovat payload v binární podobě. Ještě lépe, protocol analyzer, který dokáže filtrovat jednotlivé protokoly a rozdělit payload jednotlivých paketů od sebe. Typickým představitelem je software Wireshark [28], který nám pro realizaci této steganografické metody postačí.

Samotné označení sekvence probíhá tak, že jakmile máme převedený znak pomocí kódovací tabulky do binarního kódu, např. písmeno *S* má v binárním kódu reprezentaci *01010011*, tak vyhledáme paket RTP paket, ve kterém se nachází požadovaná sekvence. Např. sekvence *01010011* se nachází v paketu se sequence number: *39796*. Obsah payload tohoto paketu je zobrazen na obrázku (Obrázek 12). Ve zmíněném paketu si zobrazíme pouze payload, kde vyhledáme sekvenci *01010011*, která začína na pozici *73*. Tato pozice bude výchozí, od které budeme dále dopočítavat vzdálenost ostatních sekvencí. Dalším znakem je znak *T*, jeho binární reprezentace má hodnotu *01010100*. Tato sekvence začíná na pozici *673*, ale jelikož zaznamenáváme vzdálenost od prvního znaku, musíme ji dopočítat. K tomuto výpočtu použijeme následující vzorec (12.):

$$P = B - F \tag{12}$$

kde P je výsledná pozice znaku, B je číslo pozice, kde začíná sekvence reprezentující daný znak a F je číslo pozice sekvence prvního znaku.

Po aplikovaní výše zmíněného vzorce (673-73) se dostaváme k hodnotě *600*, což je vzálenost sekvence od sekvence prvního znaku. Další sekvence vyhledáváme analogicky,

do té doby, dokud nezaznamenáme pozice sekvencí všech znaků, které potřebujeme pro přenos steganogramu.

Při detekci pozic jednotlivých znaků může nastat situace, kdy se bude pozice prvního znaku nacházet na konci payloadu. Poté se pozice dalších znaků bude nacházet před znakem prvním, z čehož vyplývá jejich menší hodnota pozice. V takovém případě budou hodnoty vzdáleností následujících znaků od znaku prvního nabývat záporných hodnot. Tento jev můžeme vidět při detekci pozic znaků *2015*, kde se pozice prvního znaku *2* nachází na pozici *1225*. Při zaznamenání pozice následujícího znaku *0* dostáváme hodnotu vzdálenosti od prvního znaku *-168*.

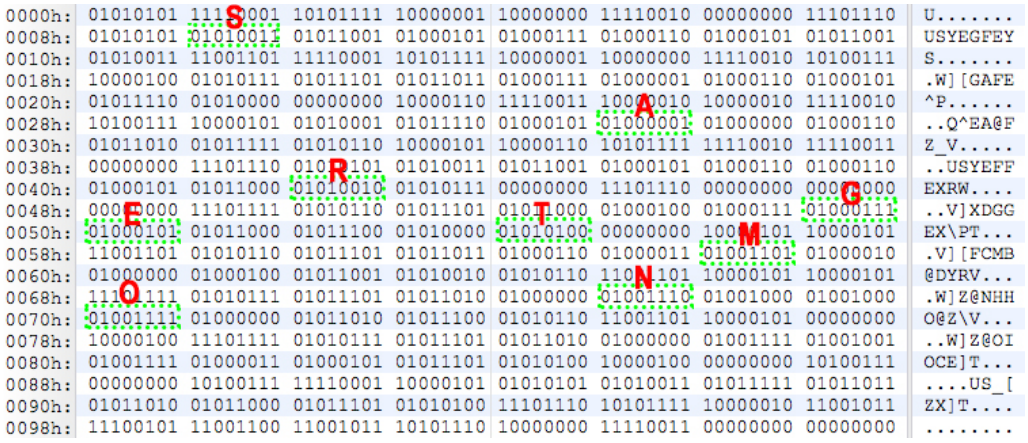
Znak	Binární reprezentace	Pozice sekvence	Sequence number
S	01010011	73	39796
T	01010100	600	39796
E	01000101	568	39796
G	01000111	560	39796
A	01000001	288	39796
N	01001110	800	39796
O	01001111	824	39796
G	01000111	560	39796
R	01010010	456	39796
A	01000001	288	39796
M	01001101	680	39796
2	00110010	1225	50556
0	00110000	-168	50556
1	00110001	-8	50556
5	00110101	-656	50556

Tabulka 8: Určení pozice sekvencí jednotlivých znaků (ASCII)

Jak můžeme vidět v tabulce (Tabulka 8), podařilo se nám pro steganogram s řetězcem *STEGANOGRAM2015* vyhledat pozice sekvencí pouze pro znaky *STEGANOGRAM* v rámci jednoho paketu se sequence number *39796*. Z toho důvodu přistoupíme k jinému paketu, kde se budou nacházet zbývající sekvence, které představují binarní reprezentaci požadovaných znaků *2015*. Požadované sekvence pro znaky *2015*, se nacházejí v paketu *50556*.

5.10 Pointer v SIP hlavičce

Po získání polohy jednotlivých sekvencí odpovídající požadovaným znakům v jednotlivých paketech, přejdeme k přenosu informací o poloze námi označených dat. Jelikož RTP protokol se využívá společně s protokolem SIP, můžeme využít jeho možností pro přenos informací o vybraných paketech a poloze jednotlivých sekvencí, které reprezentují znak. V části Realizace výměny informací pomocí vloženého pole v SIP signalizaci



Obrázek 12: RTP paket 39796 - payload

je popsána realizace přenosu informací v rámci SIP hlavičky, pomocí pole User-Agent. Nyní využijeme možnosti pole *Via* a modifikaci parametru *branch*.

5.10.1 Parametr branch

Tento parametr slouží pro jednoznačnou identifikaci SIP transakce. Hodnota parametru musí být jedinečná v rámci prostoru a času pro všechny žádosti zaslané UA. Jedinou vyjímkou tohoto pravidla jsou metody CANCEL a ACK pro odpovědi, které nejsou typu 2xx. Metoda CANCEL musí mít stejnou hodnotu branch jako žádost o CANCEL. Struktura *branch* řetězce může obsahovat alfanumerické znaky.

Typická délka řetězce *branch* genrovaného pomocí aplikací pro IP telefonii tzv. softphones je různá (Tabulka 9).

Aplikace (softphone)	Délka řetězce (bez magic cookie)
Media5-fone/4.1.3.3034 iOS/8.3	17
SessionTalk Version 5.11 iOS/8.3	28
X-Lite 4.8.0 75950-02930038-M10.10.2 OSX	28
YATE/5.0.0 Linux	9

Tabulka 9: Délka řetězce branch generovaného aplikacemi

Zdroj [32] uvádí, že maximální délka pole *Via* může nabývat hodnoty *0* až *65535*B. Typická délka je přednastavena na 1024B v rámci SNORT SIP preprocesoru. Implementace OpenSIPS[33] implementace odkazuje na velikost *branch* parametru *32B*.

Branch parametr obsahuje tzv. *magic cookie*, což je řetětezec, který je prefixem pro každou transakci. Tento řetezec se skládá z následujících znaků: *z9hG4bK* a je konstantní.

Pro potřeby přenosu informací o poloze sekvencí ve vybraných paketech využijeme tento parametr, kde požadované informace umístíme za prefix *z9hG4bK*. Následující vý-

pis obsahuje informace o poloze sekvencí pro přenos zprávy skladající se ze znaků *STEGANOGRAM2015*.

```
branch=z9hG4bK39796{73|600|568|560|288|800|824|560|456|288|680}50556{1225|
~168|~8|~656}
```

Výpis 27: Pointer v SIP hlavičce (základní)

Ovšem *branch* parametr s řetězcem zobrazeným ve zdrojovém kódu (Výpis 27), není moc vhodný pro přenos, protože takový řetězec by mohl vyvolat podezření, že obsahuje skrytý komunikační kanál. Z tohoto důvodu je nutné zakódovat i samotné ukazatele na RTP data v paketech, aby nebylo na první pohled zřejmé, že se jedná o skrytý komunikační kanál.

5.11 Kódování pointeru v SIP hlavičce

Pro kódování pointeru v SIP hlavičce jsem se snažil využít informace, které mi poskytuje stávající převodní ASCII tabulka. Jako nejschůdnější variantu jsem zvolil převod decimálních hodnot, které se vyskytují v branch řetězci na hodnoty hexadecimální. 7-bitová ASCII tabulka na obrázku (Obrázek 10) obsahuje 128 hodnot, které využiji pro kódování. Vzhledem k nutnosti použití minimálně čtyř speciálních znaků, které potřebuji pro oddělení pointeru, začátku, konce paketu a reprezentaci znaku mínus, použiji pouze *122* hodnot pro kódování obsahu a *4* hodnoty pro kódování speciálních znaků. Postupně procházíme řetězec po třech znacích. Pokud bude numerická hodnota větší než hodnota 122, odeberu jeden znak zprava a zakóduji hodnotu složenou ze dvou znaků. Tato číselná hodnota reprezentuje hodnotu v decimální soustavě. Pomocí převodní ASCII tabulky zjistíme odpovídající hodnotu, která reprezentuje danou hodnotu v hexadecimální soustavě. Následně postupujeme analogicky, dokud nezakódujeme celý řetězec.

Speciální případy nastávají při kódování znaků, které reprezentují začátek a konec pointerů, které se vztahují k dalšímu paketu, dále oddělení jednotlivých pointerů od sebe a reprezenací znaménka mínus, které je nahrazeno jiným symbolem z důvodů přehlednosti a především jednoznačenému kódovacímu schématu. Každý speciální symbol se kóduje samostatně.

Znak (ASCII)	HEX	Funkce
{	7B	počátek množiny pointerů v rámci paketu
|	7C	oddělení jednotlivých pointerů v množině
}	7D	ukončení množiny pointerů v rámci paketu
~	7E	znak reprezentující znaménko mínus

Tabulka 10: Speciální znaky pro kódování SIP hlavičky

Například pro zakódování řetězce *39796{73|600* uvažujeme první *tři* znaky řetězce zleva *397* a porovnáme, zda je numerická hodnota vetší, než hodnota *122*. Pokud je numerikcá hodnota vetší než *122*, budeme uvažovat pouze první *dva* znaky řetězce zleva. Nyní se dostáváme k numerické hodnotě *39*, kterou jsme pomocí převodní ASCII tabulky

schopni zakódovat. V decimální soustavě vyhledáme odpovídající numerickou hodnotu, v tomto případě *39*, které odpovídá hexadecimální hodnota *27*. V následujícím kroku máme numerickou hodnotu *796*, která je opět větší než hodnota *122*. Proto použijeme numerickou hodnotu *79*, které odpovídá hexadecimální hodnota *4F*. Následující znaky kódujeme analogicky, směrem zleva doprava.

Jedinou vyjímkou jsou speciální znaky, které kódujeme jako samostané znaky, převodem z ASCII symbolů do jejich hexadecimální podoby, jak popisuje tabulka 10.

```
branch=z9hG4bK274F067B497C3C067C38087C38007C1C087C50007C52047
C38007C2D067C1C087C44007D3237067B7A057C7E10087C7E087C7E41067D
```

Výpis 28: Kódovaný pointer v SIP hlavičce

Dalšího zvýšení bezpečnosti proti odhalení skrytého komunikačního kanálu docílíme tím, že můžeme hexadecimální hodnoty zobrazovat libovolně s velými *A, B, C, D, E, F* či malými *a, b, c, d, e, f* znaky i jejich kombinací.

```
branch=z9hG4bK274f067B497c3C067C38087c38007c1c087C50007c52047
C38007c2d067C1C087C44007D3237067b7a057C7E10087c7e087C7E41067d
```

Výpis 29: Kódovaný pointer v SIP hlavičce (velká a malá písmena)

5.12 Přenos informací během hovoru

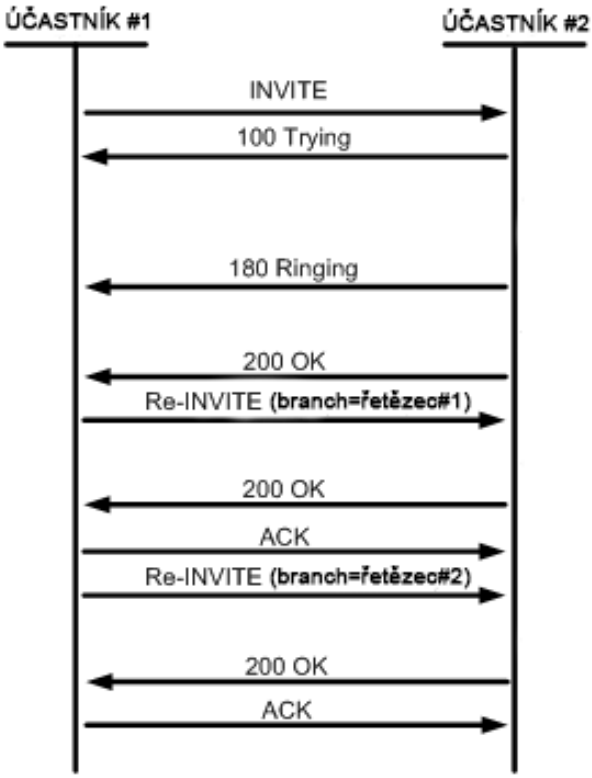
Výše popsaná metoda přenosu skrytého kanálu v rámci RTP kanálu se dá použít jak po ukončeném hovoru, tak při právě probíhajícím hovoru. Hlavní překážkou je, jak přenést informace o poloze znaků v jednotlivých paketech payloadu, aniž bych musel navazovat další spojení, pomocí kterého bych přenesl pointer v SIP hlavičce. Pro řešení lze využít metodu re-INVITE (INVITE) protokolu SIP, která umožňuje přenos informací v rámci probíhajícího hovoru. Změnou parametru již navázaného spojení pomocí metody re-INVITE, lze přenést SIP zprávu s modifikovanou hlavičkou v poli *branch* opakovaně.

V rámci pole *branch* první metody INVITE lze ověřit, zda je druhý účastník připraven přijmout steganogram nebo pole může zůstat prázdné (vyjma magic cookie). Při následující metodě re-INVITE již do pole *branch* vložíme kódovaný řetezec, který reprezentuje polohy binárních sekvencí odpovídajících znaků. Takovým způsobem můžeme přenášet tajné informace v rámci probíhajícího hovoru.

5.12.1 Metoda re-INVITE

Metoda *re-INVITE* slouží k modifikaci parametru již navázáného spojení. Tato metoda obsahuje stejně nastavené pole From, To, Call-ID jako ve zprávě *INVITE*, ale ostatní parametry lze modifikovat. *re-INVITE* lze použít až tehdy, pokud předchozí metoda *INVITE* navázala dialog, který je ukončen zprávou ACK. Pokud je *re-INVITE* z jakéhokoliv důvodu neúspšený, je i nadálě používána původní *INVITE* metoda.

Na obrázku (Obrázek 13) lze vidět, jak může přenos pomocí metody *re-INVITE* probíhat. V první fázi spojení pomocí metody *INVITE*, lze dohodnout s druhou stranou



Obrázek 13: Metoda re-INVITE

(účastníkem) , zda je připravena příjmout steganografickou zprávu. Poté již můžeme přenášet samotný tajný obsah pomocí metody *re-INVITE,* kde umístíme zakódovaný první řetezec. Takových metod *re-INVITE* můžeme v rámci hovoru vytvořit libovolné množství dle potřeby.

6 Zhodnocení dosažených výsledků

V této části prezentuji a komentuji dosažené výsledky vycházející z dat získaných v rámci vyhotovení této práce. Nejdříve se věnuji výsledkům získaných při zpracování části metody pro detekci anomálií a realizace výměny informací pomocí SIP hlavičky. V další části rozebírám výsledky získané v rámci kapitoly Využití RTP pro přenos textových informací. Pro lepší představu o výsledcích lze nahlédnout do příloh této práce, kde jsou zobrazeny tabulky, grafy a zdrojové kódy SIPp scénářů.

6.1 Realizace výměny informací pomocí vloženého pole v SIP signalizaci

Metody pro detekci anomálií a realizace výměny informací pomocí vloženého pole v SIP signalizaci jsou úzce propojeny. Na základě využití programu SIPp pro generování SIP provozu a softwarové sondy SNORT obohacené o Anomaly Detection preprocesor se mi podařilo zjistit, do jaké míry lze injektovat SIP hlavičku skrytými daty, porovnat algoritmy Holt-Winters, Holt-Winters:Brutlag, Naive Bayes a zjistit zda a za jakých podmínek jsou schopny detekovat steganogram ukrytý v SIP hlavičce.

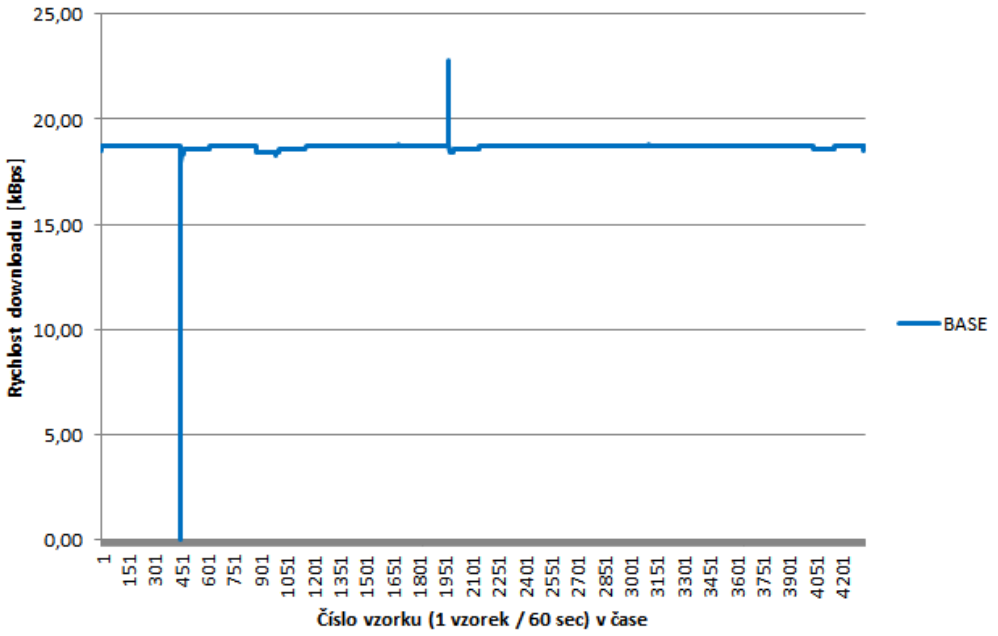
6.1.1 Optimální nastavení SNORT

Při zpracovávání této práce jsem zkoumal, jaké je optimální nastavení SNORT pro detekci skrytých dat. Zaměřil jsem se na frekvenci, s jakou SNORT zaznamenává data o provozu. V první fázi jsem nastavil frekvenci sběru dat na 1 sekundu. Tento interval se mi zpočátku jevil jako nejlepší, jelikož jsem viděl zaznamenaná data se zpožděním 1 sekundy, a mohl jsem tak přímo vidět závislosti na aktuálním nastavení a generovaném provozu. Nicméně s takto zvolenou frekvencí SNORT generuje poměrně velké množství dat, ze kterého plynou zvýšené nároky na pamětovou kapacitu. Za 24 hodin je vygenerováno 86400 záznamů.

Vytvořil jsem třídenní záznam síťového provozu, který jsem následně použil pro vygenerování predikovaného provozu. Bohužel jsem narazil na limity, preprocesoru SNORT AD či výpočetního softwaru R. Anomaly Detection. SNORT AD vyvolal chybovou hlášku *ERROR: ABNORMAL TERMINATION IN LNSRCH*. Zkoušel jsem změnu výpočetního algoritmu i modifikaci nastavení, ale bez úspěchu. Chyba byla pravděpodobně způsobena velkým množstvím dat.

Proto jsem přikročil ke snížení frekvence generování provozu nejprve na 10 sekund a později na frekvenci 60 sekund. Tuto změnu jsem si mohl dovolit, jelikož v této práci využívám data o záznamu přenosové rychlosti stahování, která jsou zaznamenávána v jednotce kBps.

Jak můžeme vidět v grafu (Obrázek 14), zaznamenaná přenosová rychlost stahování dat má velmi stabilní hodnotu pohybující se kolem 18.76 kBps. Teoreticky by hodnota měla být naprosto konstantní, nicméně je ovlivňována mnoha faktory, mezi které patří kvalita spojení (zejména nežádoucí je její kolísání), doba odezvy, nepřesné zaznamenání softwarem SNORT a kolísáním množství generovaných zpráv softwarem SIPp. Také si můžeme všimnout výrazného propadu na křivce k nulovým hodnotám, jehož příčinou



Obrázek 14: Vývoj přenosové rychlosti (download) v čase, při zaznamenávání provozu

byl výpadek spojení po dobu 2 minut. Dále je na křivce vidět výrazný nárůst přenosové rychlosti k hodnotám 22.80 a 21.04 kBps. Bez zjevné příčiny došlo ke zvýšenému přenosu dat. Jednou z možných příčin tohoto jevu je kolísání kvality spojení se servery VŠB-TU Ostrava.

6.1.2 Nastavení anomaly detection algoritmů

SNORT AD obsahuje možnosti pro nastavení algoritmů, jako je nastavení periodicity a deviation scale. Každý algoritmus má jiné požadavky na množství zaznamenaných dat, které jsou závislé na periodicitě. Např. algoritmus Holt-Winters:Brutlag potřebuje mít záznam dvou period námi požadované délky tj. minimálně dva dny. To znamená, že pokud chceme využít WEEKLY (týdenní) periodicitu, tak potřebujeme záznam síťového provozu o délce dvou týdnů. Tuto časově náročnou periodicitu jsem musel opustit a zaměřil jsem se na periodicitu DAILY (denní). Zaznamenával jsem provoz po dobu 3 dnů, a vygenerovaný log soubor jsem použil jako vstupní data pro preprocesor.

Další parametr, který jsem používal a modifikoval jej, byl *Devitation scale*. Pomocí tohoto parametru jsem mohl určit míru odchylky, která v praxi ovlivňuje minimální a maximální hodnotu, kterou daný algoritmus definuje hranice, které při překročení vyvolají výstrahu o detekci anomálie. Konkrétně jsem používal hodnoty 1, 2, 2.5 (pouze Holt-Winters:Brutlag), 3, 6. Tento rozsah jsem zvolil pro ověření míry modifikace maximální a minimální hranice a následné míry detekce anomálie. Je nutné zmínit, že reálně

se u algoritmu Holt-Winters:Brutlag používají hodnoty v intervalu 2 až 3. Hodnoty mimo tento interval jsem zvolil pro experimentální účely.

6.1.3 Porovnání detekčních algoritmů

V rámci mé práce jsem získal data, na jejichž základě jsem mohl porovnávat schopnost jednotlivých algoritmů s různým nastavením detekovat anomálie. Porovnával jsem algoritmy Holt-Winters, Holt-Winters:Brutlag a Naive Bayes a jejich schopnost detekovat injektované scénáře s různým počtem znaků. Data uvedené v tabulce (11) reprezentují 1. den predikovaného modelu.

Z tabulky (11) můžeme vyčíst, že scénář *STEG-1*, který obsahuje 100 injektovaných znaků, není schopen detekovat žádný algoritmus.

Jiný případ nastává u scénáře *STEG-2* s obsahem 500 injektovaných znaků. V tomto případě lze rozpoznat rozdíly mezi jednotlivými algoritmy a jejich nastavením *Devitation scale*. Algoritmus Holt-Winters:Brutlag byl schopen správně detekovat (D) anomálii a to při všech nastaveních devitation scale reprezentované hodnotami 1, 2, 2.5, 3, 6. Také algoritmus Naive Bayes byl schopen správně detekovat anomálii, avšak pouze s hodnotami deviation scale 1 a 2. V případě hodnot deviation scale 3 a 6, algoritmus nebyl schopen korektně detekovat anomálii. Algoritmus Holt-Winters nedokázal správně detekovat anomálii v žádném nastavení.

Scénař *STEG-3* s počtem injektovaných znaků 1000, dokázal detekovat algoritmus Holt-Winters s deviation scale 1 a 2, Holt-Winters:Brutlag ve všech nastaveních a Naive Bayes v nastavení deviation scale 1, 2 a 3. Scénař *STEG-4* obsahující 5000 injektovaných znaků jsou schopny detekovat všechny algoritmy při všech nastaveních vyjma algoritmu Holt-Winters s deviation scale 3 a 6.

Všechny následující scénáře s počtem znaků *10000* až *60000* jsou detekovatelné všemi algoritmy opět vyjma algoritmu Holt-Winters s deviation scale 3 a 6.

Z tabulky (11) vyplývá, že pokud chceme zábránit aplikaci steganografických technik v rámci modifikace SIP hlavičky je vhodné použít algoritmus Holt-Winters:Brutlag, nebo algoritmus Naive Bayes s deviation scale 1 nebo 2, nicméně žadný algoritmus není schopen detekovat 100 injektovaných znaků.

V případě, že naopak chceme přenést tajná data metodou modifikace SIP hlavičky, je vhodné hledat systémy, které používají pro detekci anomálií algoritmus Holt-Winters s dostatečně vysokým deviation scale.

Je důležité zmínit, že data v tabulce (11) se vztahují na 1. predikovaný den. Jelikož s časem algoritmy Holt-Winters a Holt-Winters:Brutlag postupně zvyšují hranice pro detekci minimalní či maximální hodnoty. To zapříčiňuje jejich čím dál menší schopnost detekovat anomálie. Tato vlastnost je dána výpočetním principem Holt-Winters algoritmu, jež je reprezentován rostoucím trendem (uptrend).

Naopak, jelikož má Naive Bayes algoritmus konstantní hodnoty pro minimální a maximální hranici, jeho účinnost se s časem nemění.

	STEG 1	STEG 2	STEG 3	STEG 4	STEG 5	STEG 6	STEG 7	STEG 8	STEG 9	STEG 10
HW-1	-	-	D	D	D	D	D	D	D	D
HW-2	-	-	D	D	D	D	D	D	D	D
HW-3	-	-	-	-	-	-	-	-	-	-
HW-6	-	-	-	-	-	-	-	-	-	-
BR-1	-	D	D	D	D	D	D	D	D	D
BR-2	-	D	D	D	D	D	D	D	D	D
BR-2.5	-	D	D	D	D	D	D	D	D	D
BR-3	-	D	D	D	D	D	D	D	D	D
BR-6	-	D	D	D	D	D	D	D	D	D
NA-1	-	D	D	D	D	D	D	D	D	D
NA-2	-	D	D	D	D	D	D	D	D	D
NA-3	-	-	D	D	D	D	D	D	D	D
NA-6	-	-	-	D	D	D	D	D	D	D

Tabulka 11: Výsledky schopnosti algoritmů detekovat anomálií - 4.den (1. predikovaný den)

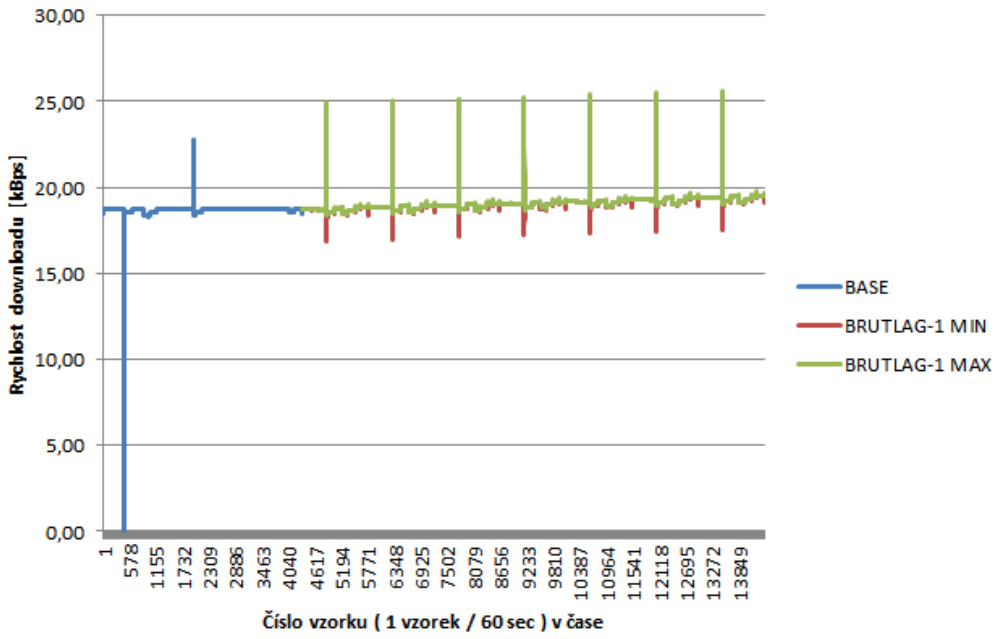
6.1.4 Závislost schopnosti detekce na konstantní přenosové rychlosti

V rámci vypracování této práce jsem pozoroval i okolnosti, které ovlivňují schopnost algoritmu detekovat anomálie. Jelikož je mnou použitá metoda založena na přenosové rychlosti, z toho vyplývá, že vyhodnocení, zda jde o anomálii či nikoliv, bude záviset na volatilitě hodnot u přenosové rychlosti při stahování. To je zapříčiněno samotnými vlastnostmi výpočetních algoritmů. V případě, že je zaznamenávaná rychlost stahování téměř konstantní, kdy odchylky jsou v řádu setin kBps, algoritmy vypočítají velmi přesné predikční modely. Naopak pokud dochází k vysokým změnám hodnoty přenosové rychlosti v řádu desetin, či jednotek kBps, tak predikční modely jsou více nepřesné a tím se omezuje jejich schopnost detekovat anomálii.

Dalším faktorem, který ovlivňuje schopnost detekce anomálií je kvalita komunikační insfrastruktury. V případě, že dochází k velkým změnám odezvy jedné z komunikujících stran, dochází k tomu, že zaznamenaná data pomocí SNORT nejsou přesná. Tyto nepřesnoti se projevují záznamem přenosové rychlosti s rázově nízkými až nulovými hodnotami, nebo naopak vysokými hodnotami. Tyto vlastnosti opět způsobují nepřesnosti při detekci anomálií.

6.1.5 Závislost schopnosti detekce na čase

Každý predikovaný model je vypočítán na základě základního (BASE) záznamu chování sítě. Tyto predikované modely vypočítavají minimální a maximální hodnoty pro následující dny. Čím je predikován vzdálenější den od posledního referenčního záznamu chování sítě, tím minimalní a maximální hodnoty mezí rostou. Tento trend se týká algorimů Holt-Winters a Holt-Winters:Brutlag. Naopak algoritmus Naive Bayes má konstatní hodnoty



Obrázek 15: Holt-Winters:Brutlag - BASE + predikovaný model. Deviation scale: 1

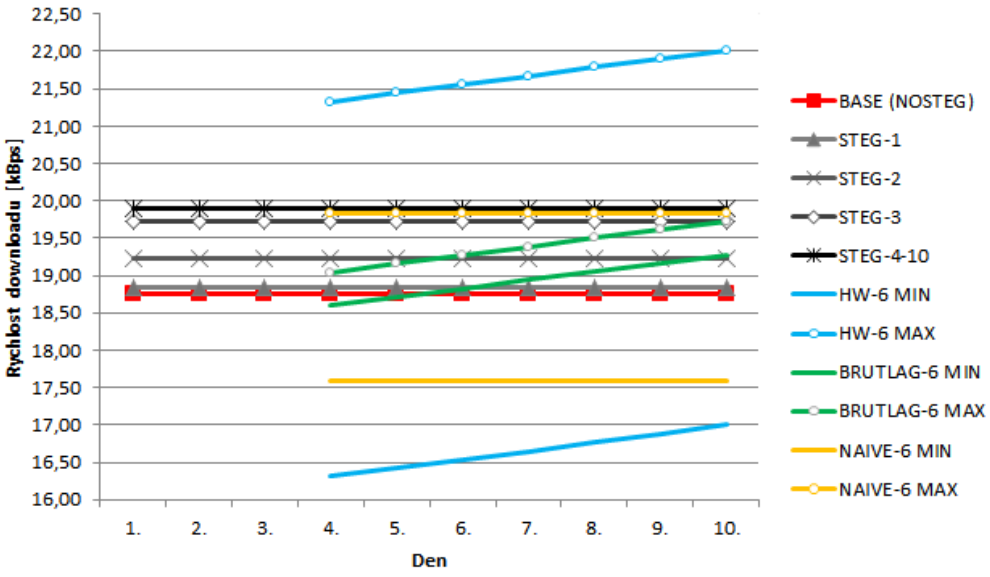
minimální a maximální meze. V grafu (Obrázek 15) můžeme vidět, jak se mění hodnoty minimální a maximální meze v závislosti na čase. Na ose X je 4. den prvním dnem, kdy je vypočten predikovaný model jednotlivými algoritmy.

6.1.6 Závislost schopnosti detekce na hodnotě deviation scale

Schopnost detekce anomálie každého algoritmu je závislá na nastavení hodnoty deviation scale. Jak je již zmíněno dříve, deviation scale určuje minimální a maximální meze sledovaných hodnot. Pokud se aktuální hodnota vybočí z těchto mezí, je vyvolána výstraha oznamující detekování anomálie. Z podstaty věci vyplývá, že pokud nastavíme malou hodnotu devitation scale, tak je schopnost detekce anomálie vyšší. V případě takového striktního nastavení se ovšem začne projevovat nežádoucí efekt, kterým je zvýšený počet falešných poplachů. To v praxi vede k tomu, že pokud je počet falešných poplachů opravdu velký a častý, tak většinou administrátor tomuto jevu přestane časem přikládat patřičnou důležitost nebo zvýší hodnotu devitation scale, čímž se sice sníží počet falešných poplachů, ale také se omezí schopnost míry detekce anomálií.

6.2 Využití RTP pro přenos textových informací

V části věnované využití RTP pro přenos textových informací jsem popsal metodu, díky které je možné přenést skrytou informaci pomocí RTP datového toku. V následující části popíšu možnosti optimalizace a také omezení této metody.



Obrázek 16: Závislost schopnosti detekce v čase 00:10:01 na deviation scale: 6.

6.2.1 Testování

Pro ověření teoretických úvah vztahujících se k této problematice jsem vytvořil standardní hovor v délce 5 minut, za využití kodeku G.711 A-law. Využíval pobočkovou ústřednu Asterisk, ke které byly připojeny dvě stanice. První stanice využívala operační systém Windows 7 se softphone aplikací Yate 5.4.2. Druhá stanice využívala operační systém OS X 10.10 se softphone aplikací X-Lite 4.8.0. Na obou stanicích byl také nainstalován software WireShark, na kterém jsem sledoval průběh komunikace. Poté jsem převedl pomocí ASCII tabulky požadovaný text do binární podoby. Následně jsem vyhledal paket, který obsahoval požadované sekvence, zaznamenal jsem si pozici první binární sekvence, od které jsem vypočítal rozdíl vzdáleností dalších binárních sekvencí. Získané polohy s číslem paketu jsem zaznamenal do parametru branch pole Via v SIP hlavičce. Následně jsem řetězec parametru branch zakódoval do hexadecimálních hodnot.

6.2.2 Optimalizace kódovací tabulky

V této práci je využita 7-bitová ASCII převodní tabulka, která nedokáže plně využít potencál 8-bitového kódování. Z tohoto důvodu je omezena přenosová kapacita skrytého kanálu. Při optimalizaci převodní tabulky s důrazem na typickou reprezentaci binárních sekvencí v payloadu běžného hovoru, bude možné efektivněji vyjádřit nejpoužívanější znaky, pomocí nejčastěji se vyskytujících binárních sekvencí.

6.2.3 Typická hodnota branch

Typická exaktní hodnota pro délku parametru branch není definovaná. V každém softphone je implementace branch parametru odlišná. V mnou testované skupině softphone se délka parametru branch pohybovala v rozmezí 9 až 28 znaků (bez magic cookie). Pokud budeme vycházet z informací uváděných v definici RFC 3261, tak musí každá implementace SIP protokolu pomocí UDP dokázat zpracovat zprávy do maximální velikosti tj. 65535 B. V rámci SIP preprocessoru pro SNORT je výchozí délka branch parametru nastavená na 1024 B s možností změny hodnoty v rozmezí 0 až 65535 B. OpenSIPS implementace odkazuje na délku 32 B. Pokud bychom chtěli zamezit využití metody popsané v této práci, pomocí omezení délky parametru na konkrétní hodnotu např. hodnotu typické délky tohoto parametru, může se v určitých situacích stát, že dojde k omezení VoIP komunikace, protože nebyla dodržena specifikace implementace definována v RFC 3261.

V situaci, kdy je nastavena hodnota délky parametru branch na hodnotu typickou pro tento parametr z důvodů bezpečnosti, lze takovému omezení přizpůsobit pointer v SIP hlavičče tak, že bude mít kratší délku, tedy délku, která je považovaná za typickou. Následně vyšším počtem zpráv re-INVITE v rámci hovoru lze kompenzovat množství přenesených dat, zapříčiněné redukcí délky parametru branch.

Pokud má pointer v SIP hlavičce typickou délku parametru branch, je tato steganografická metoda prakticky nezjistitelná.

6.2.4 Okrajové podmínky

Popsaná steganografická metoda umožňuje vytvořit skrytou komunikaci mezi dvěma koncovými účastníky. Jisté omezení plyne při umístění B2BUA (Back-to-back User Agent) do komunikační cesty. B2BUA vložený mezi dvě komunikující strany ukončí spojení výchozí stanice a vytvoří zcela nové spojení, které je směrováno na cílovou stanici. Vytvořením nového spojení dojde k vytvoření nových informací v SIP hlavičce, čímž se ztratí informace o ukazatelích umístěných v parametru branch v poli Via. Podobné omezení platí i v případě, že se v komunikační cestě nachází SBC (Session Border Controller) prvek.

Nicméně toto omezení není limitující, jelikož může být vytvořena komunikační cesta obsahující dvě SIP proxy, které jsou vzájemně propojeny pomocí SIP trunk, nebo může být vytvořené spojení přímo mezi komunikujícími UA.

Dalé lze tuto metodu využít pro detekci B2BUA prvků v komunikační cestě, na základě výše zmíněných vlastností B2BUA prvků.

7 Závěr

Práce se zabývá využítím steganografie v oblasti IP telefonie. Cílem této práce bylo rozvést myšleny prof. Szczypiorského v oblasti využití specifických polí SIP hlavičky a využití RTP pro přenos textových informací.

V úvodu jsou popsány technologie, na kterých je postavena IP telefonie a steganografie a jsou rozebrány metody pro její implementaci.

První část se zabývá oblastí výměny informací pomocí vloženého pole v SIP signalizaci, kde jsem realizoval výměnu informací prostřednictvím SIP zpráv generovaných aplikace SIPp, kde jsem vytvořil scénáře, které reprezentují požadovanou komunikaci. Využil jsem metody OPTIONS, a jejího pole User-Agent pro injektování skrytých dat. Vytvořené injektované scénárě obsahovaly 100 až 60000 znaků.

Přínosem v oblasti výměny informací pomocí vloženého pole v SIP signalizaci, je zjištění limitů pro množství přenesených znaků v metodě OPTIONS s využítím pole User-Agent. Tento přístup byl prakticky demonstorván.

Také jsem se věnoval metodám pro detekci anomálií, kde jsem používal behaviorální analýzu síťového provozu pomocí IDS SNORT s preprocesorem AD využívající statistických algoritmů Naive Bayes, Holt-Winters a Holt-Winters:Brutlag. Vytvořil jsem referenční model síťového provozu, který byl generován pomocí aplikace SIPp, spuštené ve dvou instancích. Poté jsem vytvořil predikované modely síťového provozu pomocí výše zmíněných algoritmů. Následně jsem spustil dvě instance aplikace SIPp, z nichž v jedné instanci byl spuštěn injektovaný scénář a ve druhé instanci neinjektovaný scénář. Použité scénáře byly vytvořeny v rámci kapitoly Realizace výměny informací pomocí vloženého pole v SIP. signalizaci. Metoda detekce byla založena na principu rozdílné rychlosti downloadu normální a skryté komunikace.

Přínosem práce v této oblasti je zjištění míry schopnosti výše zmíněných algoritmů detekovat anomálii v rámci IP telefonie s využitím signalizačního protokolu SIP. Výsledkem je, že skrytou komunikaci se 100 injektovanými znaky nedokáže detekovat žádný z algoritmů, ani při nastavení deviation scale na hodnotu 1.

Ve druhé části práce přicházím s vlastním řešením utajení komunikace v datovém toku protokolu RTP. Tato metoda je založena na principu označení binárních sekvencí v RTP toku, kde tyto sekvence reprezentují znaky zakódované pomocí tabulky ASCII. Poloha jednotlivých binárních sekvencí a paketu, ve kterém se nachází je zaznamenána v parametru branch pole Via v SIP hlavičce. Pro zvýšení utajení jsem dále využil překódování do hexadecimální podoby. Samotný přenos série ukazatelů v SIP hlavičce je realizován pomocí metody re-INVITE. Jisté omezení použití této metody nastává v případě, že se v komunikační cestě nachází prvek, který znehodnotí původní hodnotu parametru branch v poli Via, nicméně toto omezení lze eliminovat využitím dvou SIP proxy propojených pomocí SIP trunk v komunikační cestě, nebo využít přímé komunikační cesty mezi UA.

Přínosem této steganografické metody je, že nemodifikuje přenášená data a tudíž nedochází k degradaci či změně přenášených informací. Dále je tímto přístupem zamezeno využití steganoanalýzy. Pokud délka ukazatele v parametru branch odpovídá maximu typických hodnot, je tato metoda prakticky nezjistitelná.

8 Reference

[1] W. Mazurczyk, K. Szczypiorski, Steganography of VoIP streams [online]. 2008 [cit. 2014-10-12]. *Steganography of VoIP streams*, Dostupné z: http://www.academia.edu/2608063/

[2] W. Mazurczyk, K. Szczypiorski, Covert Channels in SIP for VoIP signalling [online]. 2008 [cit. 2014-10-12]. *Covert Channels in SIP for VoIP signalling*, Dostupné z: http://arxiv.org/pdf/0805.3538.pdf

[3] M. Szmit, A. Szmit, Usage of Modified Holt-Winters Method in the Anomaly Detection of Network Traffic: Case Studies [online]. 2012 [cit. 2014-10-14]. *Usage of Modified Holt-Winters Method in the Anomaly Detection of Network Traffic: Case Studies*, Dostupné z: http://downloads.hindawi.com/journals/jcnc/2012/192913.pdf

[4] V. Berk, A. Giani, G. Cybenko, Detection of Covert Channel Encoding in Network Packet Delays [online]. 2005 [cit. 2014-10-14]. *Detection of Covert Channel Encoding in Network Packet Delays*, Dostupné z: http://www.ists.dartmouth.edu/library/149.pdf

[5] M. Mehić, M. Mikulec, M. Voznak, L. Kapicak, Creating Covert Channel Using SIP [online]. 2014 [cit. 2014-10-14]. *Creating Covert Channel Using SIP*, Dostupné z: http://link.springer.com/chapter/10.1007/978-3-319-07569-3_15

[6] T. Heitel, Utilizing psychoacoustics model and wavelet packet transform for purposes of audio signal watermarking [online]. 2010 [cit. 2015-01-14]. *Utilizing psychoacoustics model and wavelet packet transform for purposes of audio signal watermarking*, Dostupné z: http://www.vutbr.cz/www_base/zav_prace_soubor_verejne.php?file_id=27568

[7] R. Žilka, Steganografie a stegoanalýza [online]. 2008 [cit. 2015-01-14]. *Steganografie a stegoanalýza*, Dostupné z: http://is.muni.cz/th/73058/fi_m/Steganografie_a_stegoanalyza.pdf

[8] M. Vozňák, Voice over IP, Vysokoškolská skripta [online]. 2012 [cit. 2014-10-21]. *Voice over IP, Vysokoškolská skripta*, Dostupné z: http://homel.vsb.cz/~voz29/publikace.html

[9] P. Bartel, Detekce anomálií a vizualizace síťového provozu [online]. 2011 [cit. 2015-02-08]. *Detekce anomálií a vizualizace síťového provozu*, Dostupné z: http://is.muni.cz/th/207644/fi_m/dp.pdf

[10] M. Barabas, M. Drozd, Pokročilé formy útoků a jejich detekce, SystemOnLine.cz [online]. 2013 [cit. 2015-02-08]. *Pokročilé formy útoků a jejich detekce*, Dostupné z: http://www.systemonline.cz/it-security/pokrocile-formy-utoku-a-jejich-detekce.htm

[11] M. Mehic, J. Slachta, M. Voznak, Hiding Data in SIP Session, In Proc. 37th International Conference on Telecommunication and Signal Processing, Berlin, July 1-3, 2014, ISBN 978-80-214-4983-1, ISSN 1805-5435, pp. 18-22. . 2014 [cit. 2015-02-08]. *Hiding Data in SIP Session*

[12] RFC 3261 - datatracker.ietf.org [online]. 2002 [cit. 2015-02-08]. *RFC 3261 - datatracker.ietf.org*, Dostupné z: http://www.rfc-editor.org/rfc/pdfrfc/rfc3261.txt.pdf

[13] RFC 2327 - datatracker.ietf.org [online]. 1998 [cit. 2015-02-08]. *RFC 2327 - datatracker.ietf.org*, Dostupné z: http://www.rfc-editor.org/rfc/pdfrfc/rfc2327.txt.pdf

[14] J. Arlt, M. Arltová, E. Rublíková. Analýza ekonomických časových řad s příklady [online]. 2002 [cit. 2015-03-10]. *J. Arlt, M. Arltová, E. Rublíková. Analýza ekonomických časových řad s příklady*, Dostupné z: http://nb.vse.cz/ arltova/vyuka/crsbir02.pdf

[15] L. Koritarová. Holtova-Wintersova metoda pro sezónní vyrovnávání [online]. 2014 [cit. 2015-03-10]. *L. Koritarová. Holtova-Wintersova metoda pro sezónní vyrovnávání*, Dostupné z: https://is.cuni.cz/webapps/zzp/download/130128212/?lang=cs

[16] R. Orkáč. IDS Snort [online]. 2006 [cit. 2015-03-10]. *R. Orkáč. IDS Snort*, Dostupné z: http://www.cs.vsb.cz/grygarek/SPS/projekty0506/Snort.pdf

[17] SNORT IDS [online]. [cit. 2015-03-10]. *SNORT IDS*, Dostupné z: https://www.snort.org

[18] SNORT AD [online]. [cit. 2015-03-10]. *SNORT AD*, Dostupné z: http://anomalydetection.info

[19] T. Kaur. A Hybrid approach using Signature and Anomaly Detection to detect network Intrusions [online]. 2013 [cit. 2015-03-10]. *T. Kaur. A Hybrid approach using Signature and Anomaly Detection to detect network Intrusions*, Dostupné z: http://dspace.thapar.edu:8080/dspace/bitstream/10266/2332/1/tejvir.pdf

[20] AsteriskWin32 [online]. [cit. 2015-03-10]. *Asterisk pro platformu Windows*, Dostupné z http://www.asteriskwin32.com

[21] LTE-Advanced - LTE Release10 [online]. [cit. 2015-03-10]. *LTE-Advanced - LTE Release10*, Dostupné z http://www.3gpp.org/technologies/keywords-acronyms/97-lte-advanced

[22] R Project for Statistical Computing [online]. [cit. 2015-03-10]. *R Project for Statistical Computing*, Dostupné z: http://www.r-project.org

[23] SIPp [online]. [cit. 2015-03-10]. *SIPp*, Dostupné z: http://sipp.sourceforge.net

[24] RTP header [online]. [cit. 2015-03-25]. *RTP header*, Dostupné z: https://prof.hti.bfh.ch/myf1/www/projects/polyphem/www/documents/images/rtp_header.jpg

[25] Wikipedia.org RCTP [online]. [cit. 2015-03-25]. *Wikipedia.org RCTP*, Dostupné z: http://cs.wikipedia.org/wiki/RTCP

[26] RTCP - SR (Sender Report) zpráva [online]. [cit. 2015-03-25]. *Wikipedia.org RCTP*, Dostupné z: http://flylib.com/books/4/245/1/html/2/files/05fig05.gif

[27] ASCII převodní tabulka [online]. [cit. 2015-04-06]. *ASCII převodní tabulka*, Dostupné z: http://web.alfredstate.edu/weimandn/miscellaneous/ascii/ASCII%20Conversion%20Chart.gif

[28] Wireshark [online]. [cit. 2015-04-06]. *Wireshark*, Dostupné z: https://www.wireshark.org

[29] M. Bartík, Vyuková demonstrace digitální steganografie [online]. 2013 [cit. 2014-10-21]. *M. Bartík, Vyuková demonstrace digitální steganografie*, Dostupné z: https://dip.felk.cvut.cz/browse/pdfcache/bartimar_2013bach.pdf

[30] J. Bartl, Steganografie a možnosti jejího využití [online]. 2010 [cit. 2014-10-21]. *J. Bartl, Steganografie a možnosti jejího využití*, Dostupné z: http://digilib.k.utb.cz/bitstream/handle/10563/11693/bartl_2010_dp.pdf?sequence=1

[31] M. Szmit, A. Szmit, S. Adamus, S. Bugala, Usage of Holt-Winters Model and Multilayer Perceptron in Network Traffic Modelling and Anomaly Detection [online]. 2012 [cit. 2014-10-21]. *M. Szmit, A. Szmit, S. Adamus, S. Bugala, Usage of Holt-Winters Model and Multilayer Perceptron in Network Traffic Modelling and Anomaly*, Dostupné z: https://www.academia.edu/5553132/Usage_of_Holt-Winters_Model_and_Multilayer_Perceptron_in_Network_Traffic_Modelling_and_Anomaly_Detection

[32] SNORT Manual - SIP preprocessor [online].[cit. 2015-04-06]. *SNORT Manual - SIP preprocessor*, http://manual.snort.org/node172.html

[33] Opensips [online].[cit. 2015-04-06]. *Opensips*, http://fossies.org/dox/opensips-1.11.4_src/md5utils_8h.htmla8f0d65e6fa1060f639981aec2c9fa5c5

A Tabulky

Zde jsou umístěny tabulkové podklady vypracované v rámci této práce.

	NO STEG	STEG 1	STEG 2	STEG 3	STEG 4	STEG 5	STEG 6	STEG 7	STEG 8	STEG 9	STEG 10
HW-1	-	-	-	D	D	D	D	D	D	D	D
HW-2	-	-	-	D	D	D	D	D	D	D	D
HW-3	-	-	-	-	-	-	-	-	-	-	-
HW-6	-	-	-	-	-	-	-	-	-	-	-
BRUTLAG-1	-	-	D	D	D	D	D	D	D	D	D
BRUTLAG-2	-	-	D	D	D	D	D	D	D	D	D
BRUTLAG-2.5	-	-	D	D	D	D	D	D	D	D	D
BRUTLAG-3	-	-	D	D	D	D	D	D	D	D	D
BRUTLAG-6	-	-	D	D	D	D	D	D	D	D	D
NAIVE-1	-	-	D	D	D	D	D	D	D	D	D
NAIVE-2	-	-	D	D	D	D	D	D	D	D	D
NAIVE-3	-	-	-	D	D	D	D	D	D	D	D
NAIVE-6	-	-	-	-	D	D	D	D	D	D	D

Tabulka 12: Porovnání výsledků algoritmů při detekci anomálií - 4.den (1. predikovaný den)

	NO STEG	STEG 1	STEG 2	STEG 3	STEG 4	STEG 5	STEG 6	STEG 7	STEG 8	STEG 9	STEG 10
HW-1	-	-	-	D	D	D	D	D	D	D	D
HW-2	-	-	-	-	D	D	D	D	D	D	D
HW-3	-	-	-	-	-	-	-	-	-	-	-
HW-6	-	-	-	-	-	-	-	-	-	-	-
BRUTLAG-1	-	-	D	D	D	D	D	D	D	D	D
BRUTLAG-2	-	-	D	D	D	D	D	D	D	D	D
BRUTLAG-2.5	-	-	D	D	D	D	D	D	D	D	D
BRUTLAG-3	-	-	D	D	D	D	D	D	D	D	D
BRUTLAG-6	-	-	D	D	D	D	D	D	D	D	D
NAIVE-1	-	-	D	D	D	D	D	D	D	D	D
NAIVE-2	-	-	D	D	D	D	D	D	D	D	D
NAIVE-3	-	-	-	D	D	D	D	D	D	D	D
NAIVE-6	-	-	-	-	D	D	D	D	D	D	D

Tabulka 13: Porovnání výsledků algoritmů při detekci anomálií - 5.den (2. predikovaný den)

	NO STEG	STEG 1	STEG 2	STEG 3	STEG 4	STEG 5	STEG 6	STEG 7	STEG 8	STEG 9	STEG 10
HW-1	-	-	-	D	D	D	D	D	D	D	D
HW-2	-	-	-	-	D	D	D	D	D	D	D
HW-3	-	-	-	-	-	-	-	-	-	-	-
HW-6	-	-	-	-	-	-	-	-	-	-	-
BRUTLAG-1	-	-	D	D	D	D	D	D	D	D	D
BRUTLAG-2	-	-	D	D	D	D	D	D	D	D	D
BRUTLAG-2.5	-	-	D	D	D	D	D	D	D	D	D
BRUTLAG-3	-	-	D	D	D	D	D	D	D	D	D
BRUTLAG-6	-	-	-	D	D	D	D	D	D	D	D
NAIVE-1	-	-	D	D	D	D	D	D	D	D	D
NAIVE-2	-	-	D	D	D	D	D	D	D	D	D
NAIVE-3	-	-	-	D	D	D	D	D	D	D	D
NAIVE-6	-	-	-	-	D	D	D	D	D	D	D

Tabulka 14: Porovnání výsledků algoritmů při detekci anomálií - 6.den (3. predikovaný den)

	NO STEG	STEG 1	STEG 2	STEG 3	STEG 4	STEG 5	STEG 6	STEG 7	STEG 8	STEG 9	STEG 10
HW-1	-	-	-	D	D	D	D	D	D	D	D
HW-2	-	-	-	-	-	-	-	-	-	-	-
HW-3	-	-	-	-	-	-	-	-	-	-	-
HW-6	-	-	-	-	-	-	-	-	-	-	-
BRUTLAG-1	-	-	D	D	D	D	D	D	D	D	D
BRUTLAG-2	-	-	-	D	D	D	D	D	D	D	D
BRUTLAG-2.5	-	-	-	D	D	D	D	D	D	D	D
BRUTLAG-3	-	-	-	D	D	D	D	D	D	D	D
BRUTLAG-6	-	-	-	D	D	D	D	D	D	D	D
NAIVE-1	-	-	D	D	D	D	D	D	D	D	D
NAIVE-2	-	-	D	D	D	D	D	D	D	D	D
NAIVE-3	-	-	-	D	D	D	D	D	D	D	D
NAIVE-6	-	-	-	-	D	D	D	D	D	D	D

Tabulka 15: Porovnání výsledků algoritmů při detekci anomálií - 7.den (4. predikovaný den)

	NO STEG	STEG 1	STEG 2	STEG 3	STEG 4	STEG 5	STEG 6	STEG 7	STEG 8	STEG 9	STEG 10
HW-1	-	-	-	D	D	D	D	D	D	D	D
HW-2	-	-	-	-	-	-	-	-	-	-	-
HW-3	-	-	-	-	-	-	-	-	-	-	-
HW-6	-	-	-	-	-	-	-	-	-	-	-
BRUTLAG-1	-	-	-	D	D	D	D	D	D	D	D
BRUTLAG-2	-	-	-	D	D	D	D	D	D	D	D
BRUTLAG-2.5	-	-	-	D	D	D	D	D	D	D	D
BRUTLAG-3	-	-	-	D	D	D	D	D	D	D	D
BRUTLAG-6	-	-	-	D	D	D	D	D	D	D	D
NAIVE-1	-	-	D	D	D	D	D	D	D	D	D
NAIVE-2	-	-	D	D	D	D	D	D	D	D	D
NAIVE-3	-	-	-	D	D	D	D	D	D	D	D
NAIVE-6	-	-	-	-	D	D	D	D	D	D	D

Tabulka 16: Porovnání výsledků algoritmů při detekci anomálií - 8.den (5. predikovaný den)

	NO STEG	STEG 1	STEG 2	STEG 3	STEG 4	STEG 5	STEG 6	STEG 7	STEG 8	STEG 9	STEG 10
HW-1	-	-	-	-	D	D	D	D	D	D	D
HW-2	-	-	-	-	-	-	-	-	-	-	-
HW-3	-	-	-	-	-	-	-	-	-	-	-
HW-6	-	-	-	-	-	-	-	-	-	-	-
BRUTLAG-1	-	-	-	D	D	D	D	D	D	D	D
BRUTLAG-2	-	-	-	D	D	D	D	D	D	D	D
BRUTLAG-2.5	-	-	-	D	D	D	D	D	D	D	D
BRUTLAG-3	-	-	-	D	D	D	D	D	D	D	D
BRUTLAG-6	-	-	-	D	D	D	D	D	D	D	D
NAIVE-1	-	-	D	D	D	D	D	D	D	D	D
NAIVE-2	-	-	D	D	D	D	D	D	D	D	D
NAIVE-3	-	-	-	D	D	D	D	D	D	D	D
NAIVE-6	-	-	-	-	D	D	D	D	D	D	D

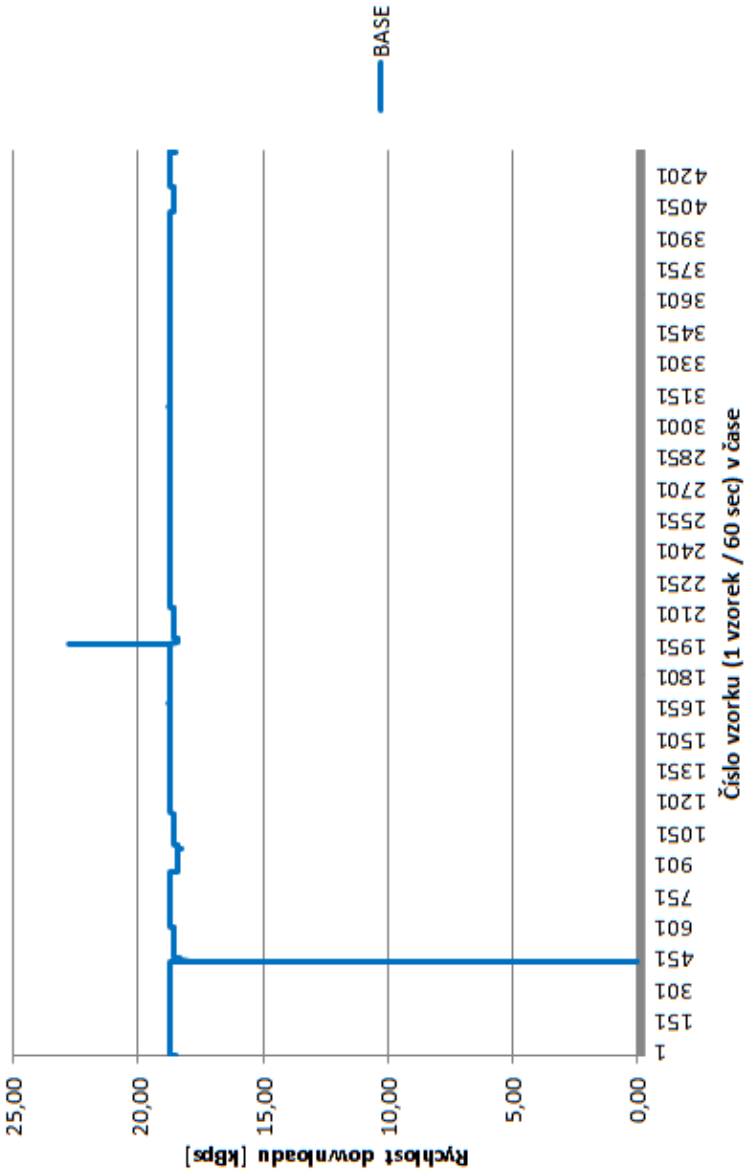
Tabulka 17: Porovnání výsledků algoritmů při detekci anomálií - 9.den (6. predikovaný den)

	NO STEG	STEG 1	STEG 2	STEG 3	STEG 4	STEG 5	STEG 6	STEG 7	STEG 8	STEG 9	STEG 10
HW-1	-	-	-	-	-	-	-	-	-	-	-
HW-2	-	-	-	-	-	-	-	-	-	-	-
HW-3	-	-	-	-	-	-	-	-	-	-	-
HW-6	-	-	-	-	-	-	-	-	-	-	-
BRUTLAG-1	-	-	-	D	D	D	D	D	D	D	D
BRUTLAG-2	-	-	-	D	D	D	D	D	D	D	D
BRUTLAG-2.5	-	-	-	D	D	D	D	D	D	D	D
BRUTLAG-3	-	-	-	D	D	D	D	D	D	D	D
BRUTLAG-6	-	-	-	-	D	D	D	D	D	D	D
NAIVE-1	-	-	D	D	D	D	D	D	D	D	D
NAIVE-2	-	-	D	D	D	D	D	D	D	D	D
NAIVE-3	-	-	-	D	D	D	D	D	D	D	D
NAIVE-6	-	-	-	-	D	D	D	D	D	D	D

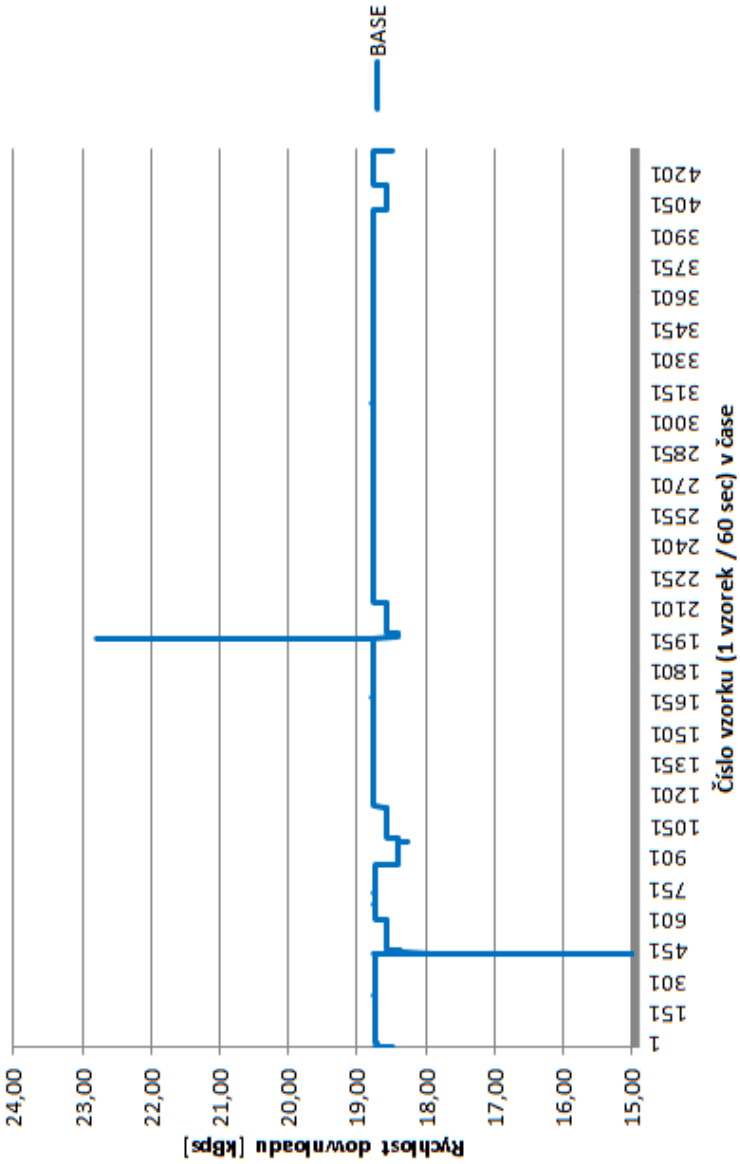
Tabulka 18: Porovnání výsledků algoritmů při detekci anomálií - 10.den (7. predikovaný den)

B Grafy

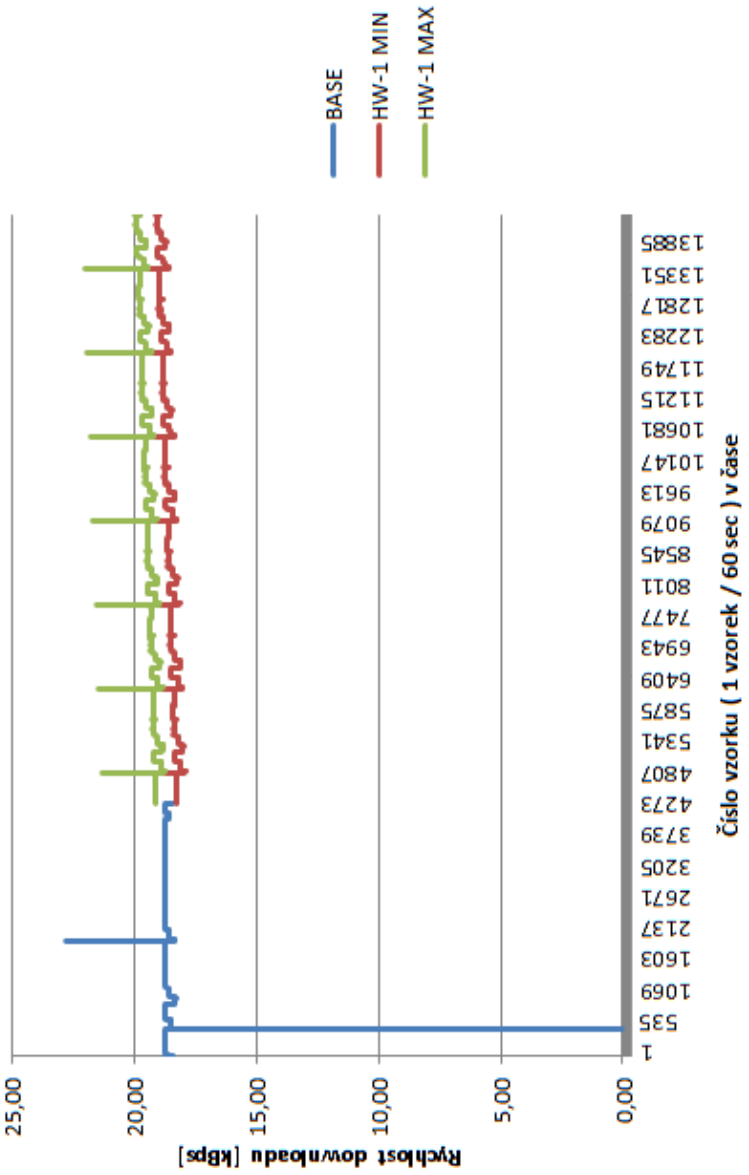
Zde jsou umístěny grafy vypracované v rámci této práce.



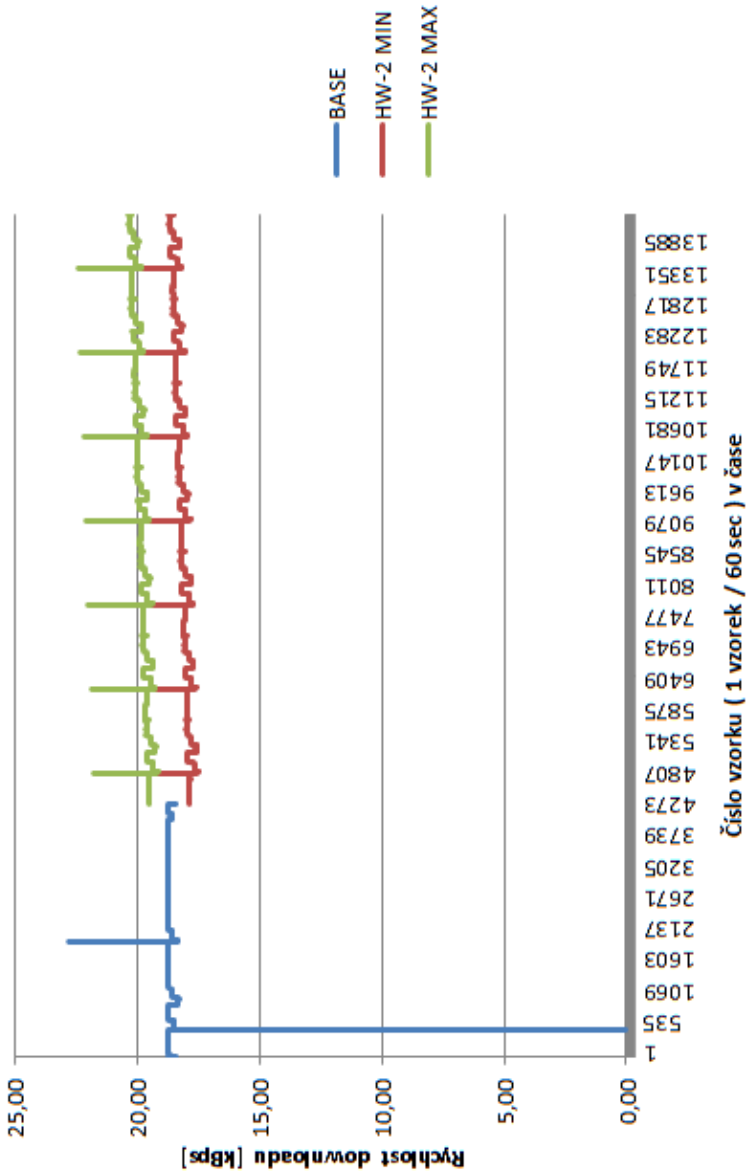
Obrázek 17: Záznam základního (BASE) modelu (ADLog60.txt)



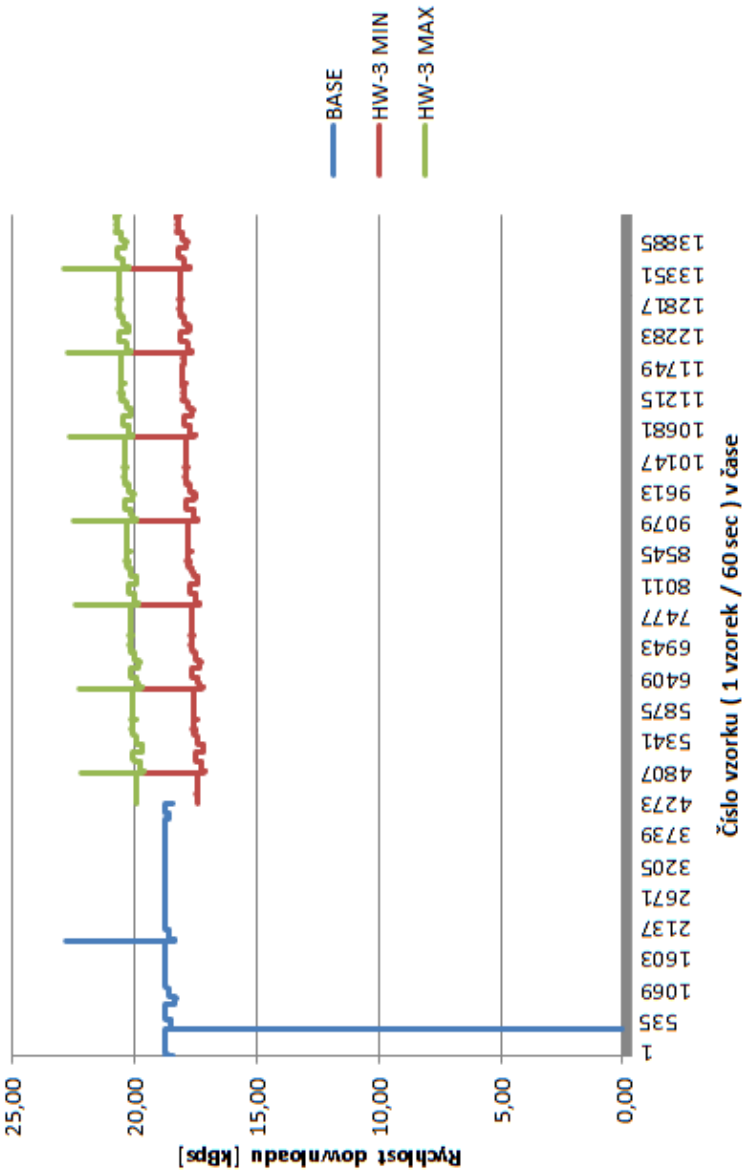
Obrázek 18: Záznam základního (BASE) modelu - detail (ADLog60.txt)



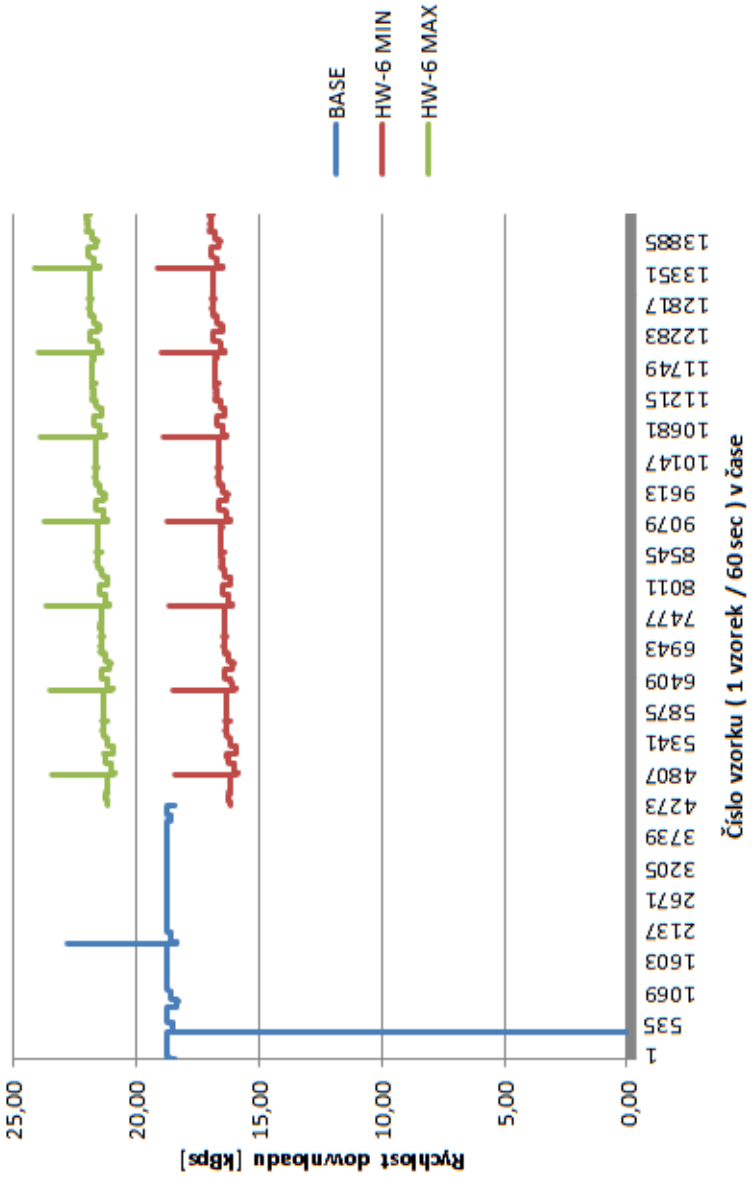
Obrázek 19: Holt-Winters - BASE + predikovaný model. Deviation scale: 1



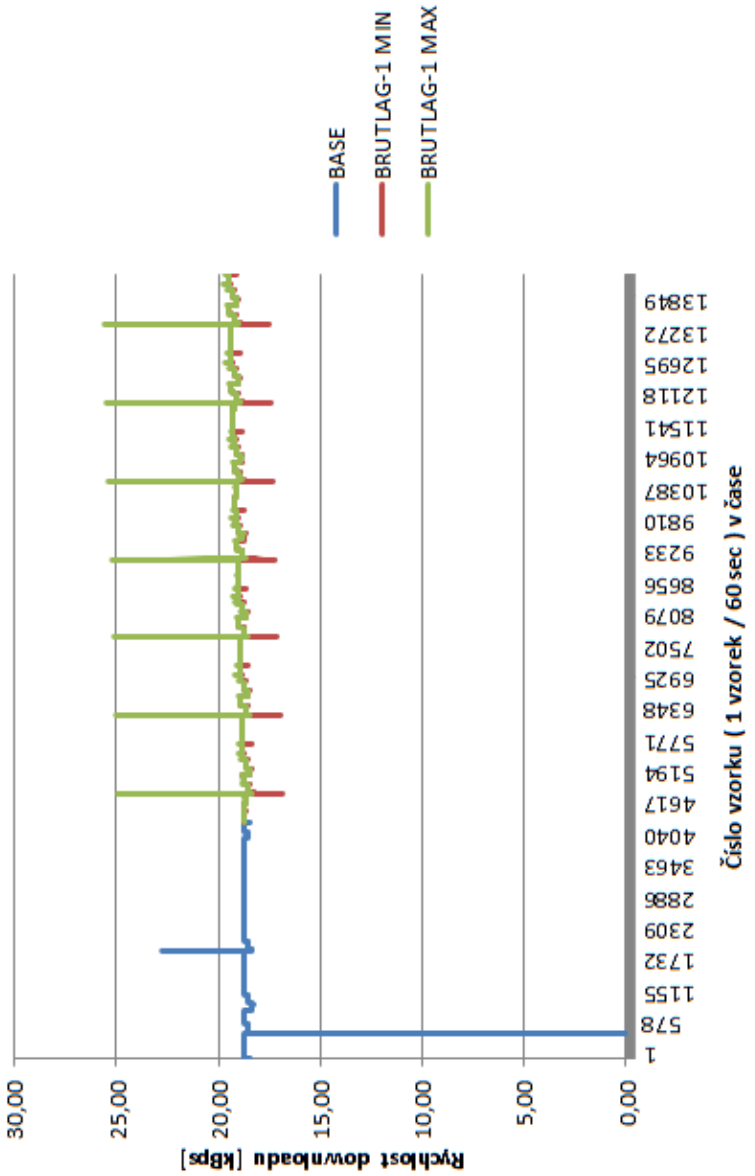
Obrázek 20: Holt-Winters - BASE + predikovaný model. Deviation scale: 2



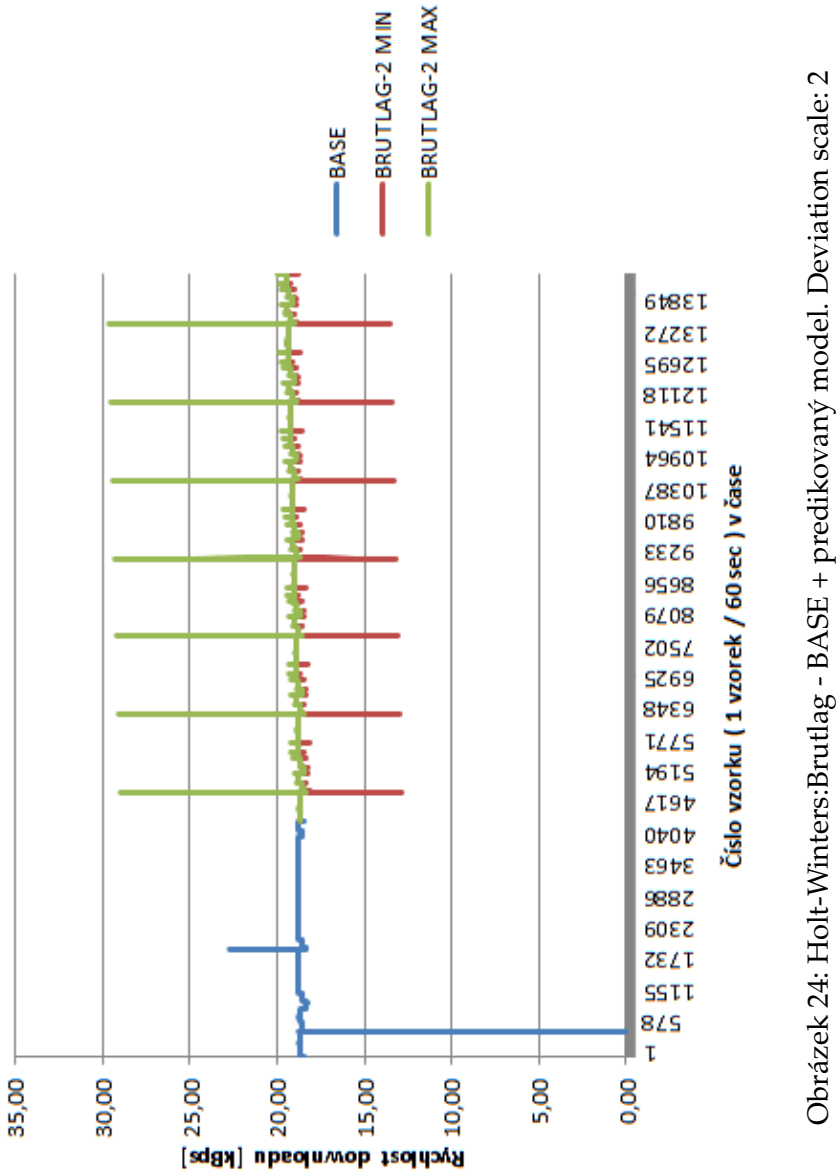
Obrázek 21: Holt-Winters - BASE + predikovaný model. Deviation scale: 3



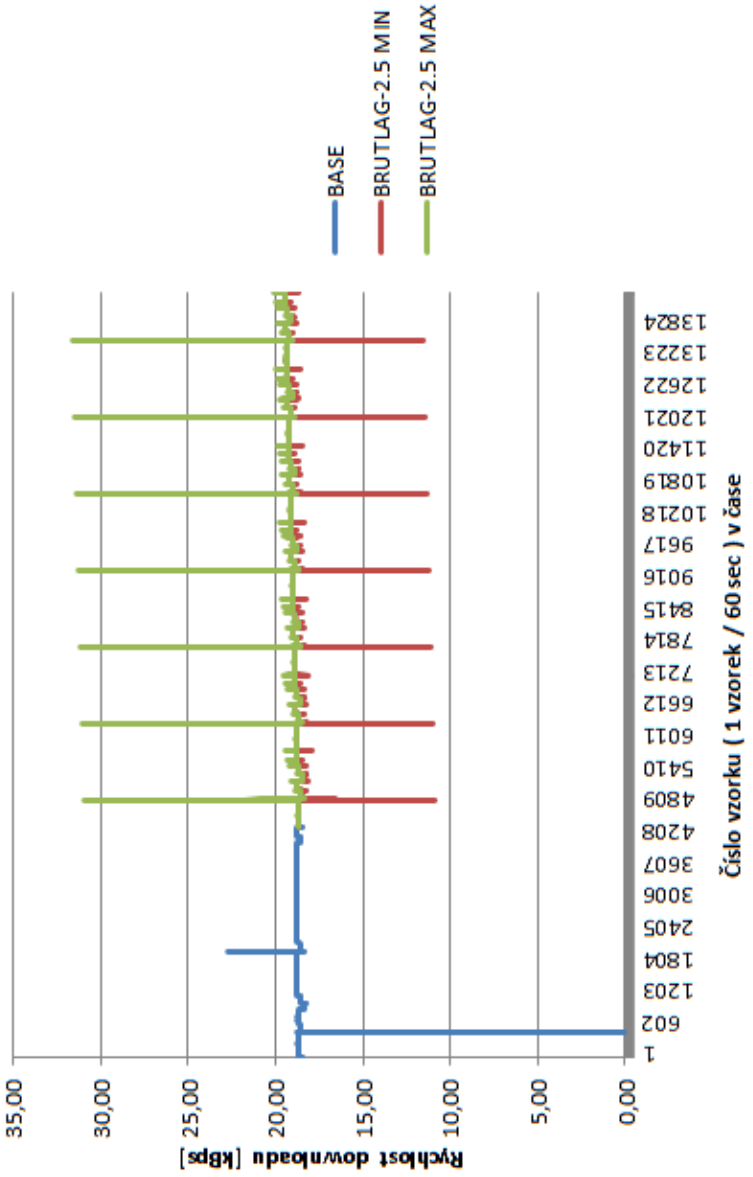
Obrázek 22: Holt-Winters - BASE + predikovaný model. Deviation scale: 6



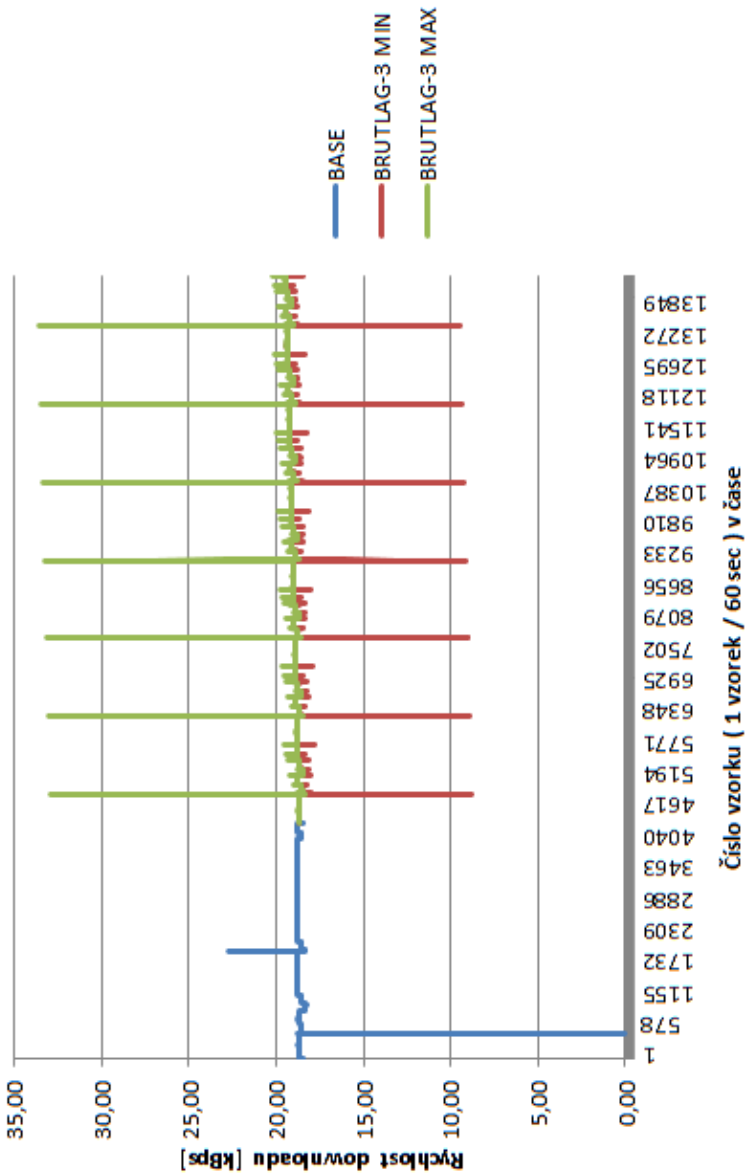
Obrázek 23: Holt-Winters:Brutlag - BASE + predikovaný model. Deviation scale: 1



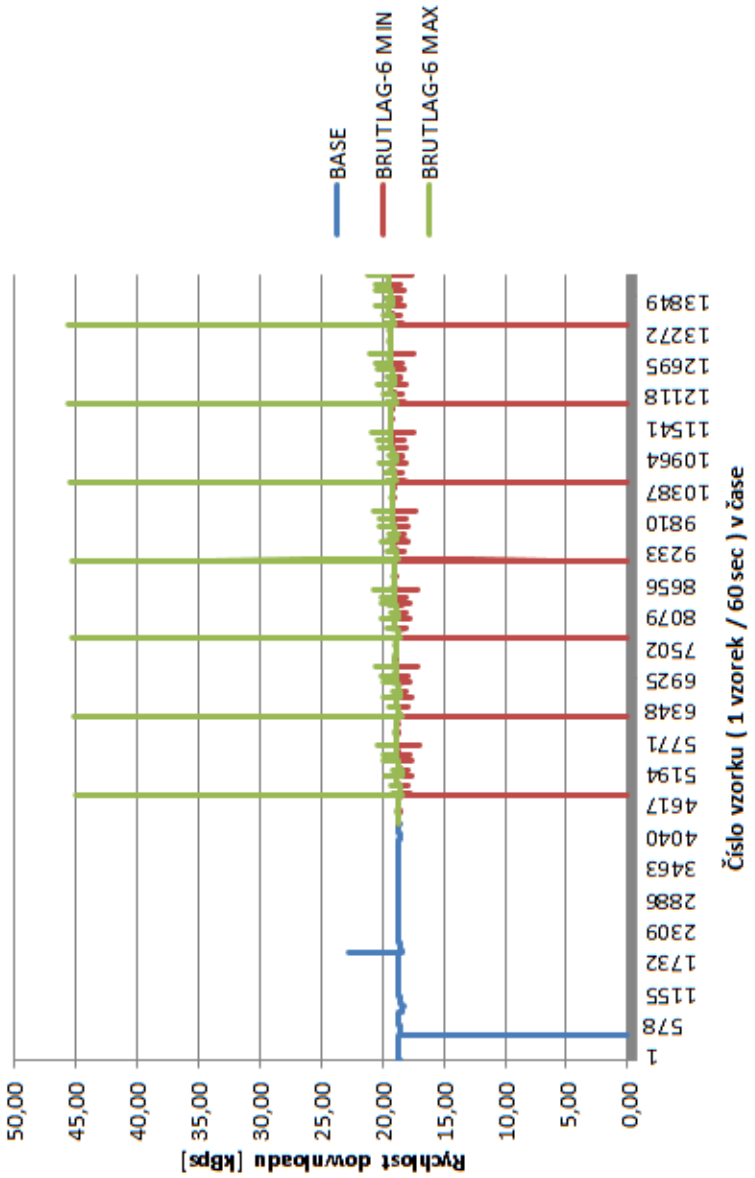
Obrázek 24: Holt-Winters:Brutlag - BASE + predikovaný model. Deviation scale: 2



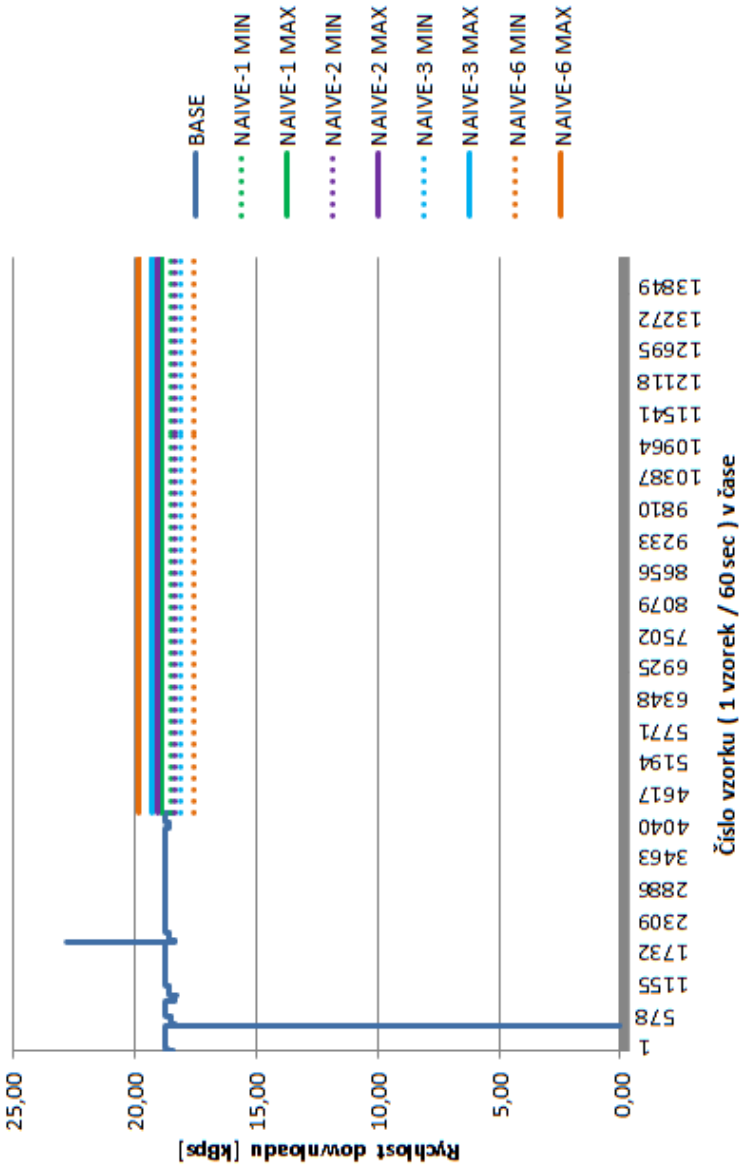
Obrázek 25: Holt-Winters:Brutlag - BASE + predikovaný model. Deviation scale: 2.5



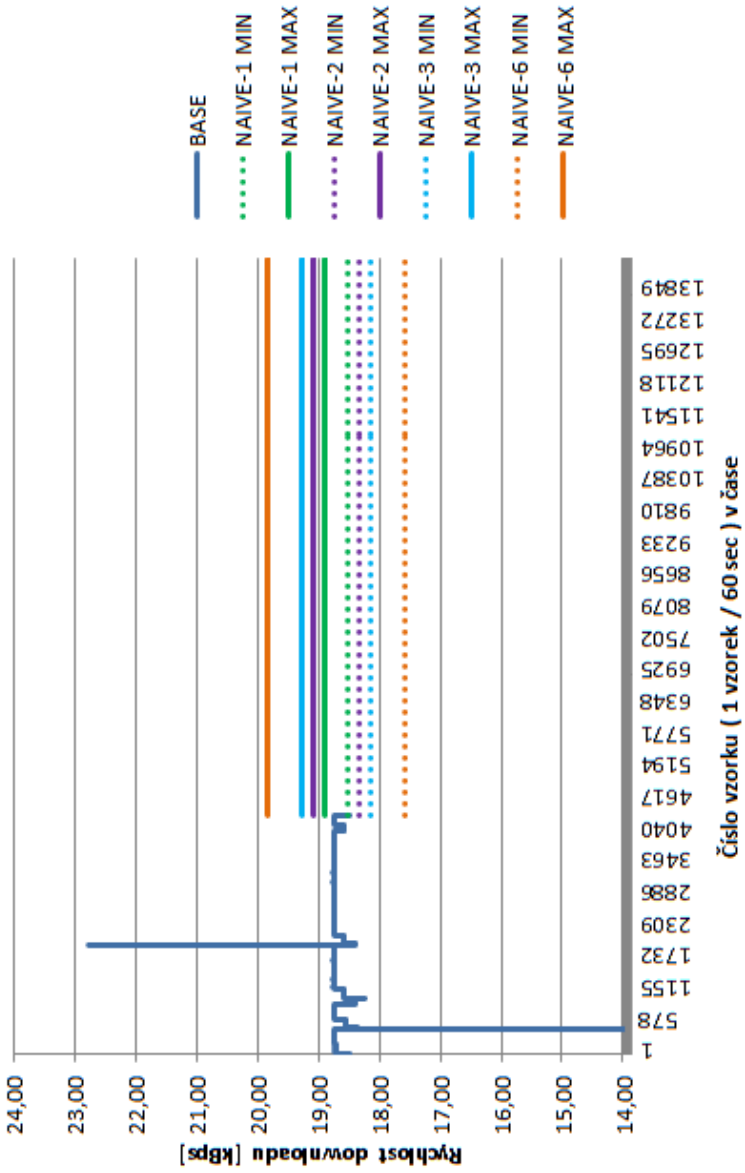
Obrázek 26: Holt-Winters:Brutlag - BASE + predikovaný model. Deviation scale: 3



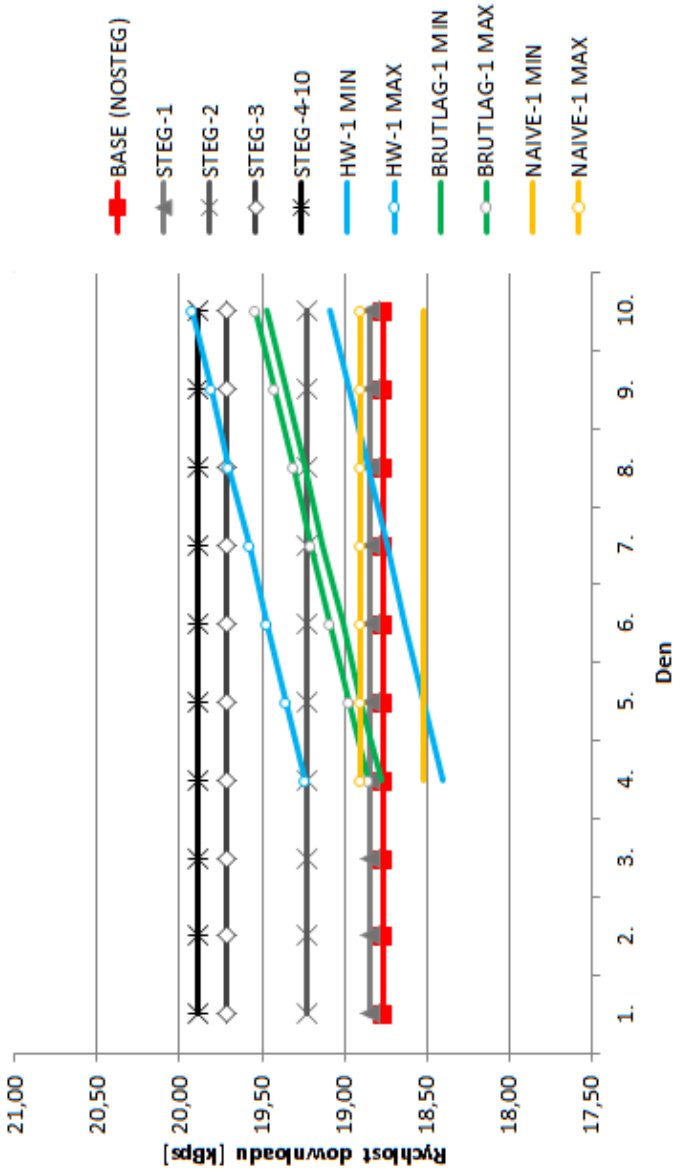
Obrázek 27: Holt-Winters:Brutlag - BASE + predikovaný model. Deviation scale: 6



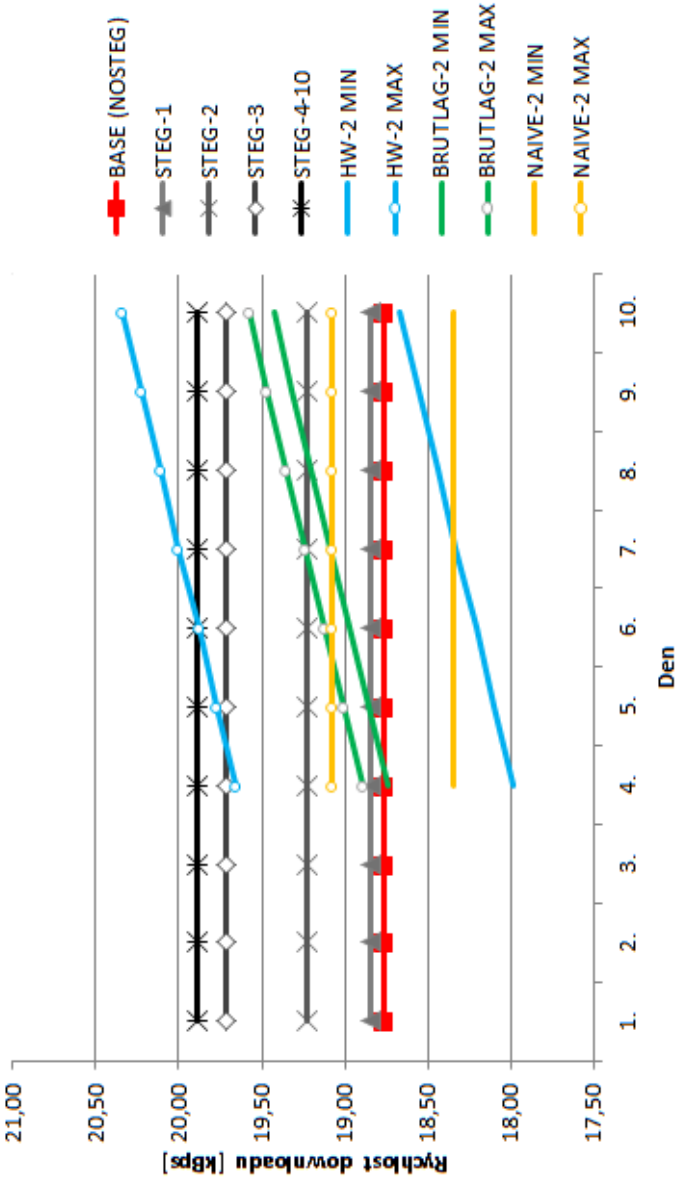
Obrázek 28: Naive - BASE + predikovaný model. Deviation scale: 1, 2, 3, 6



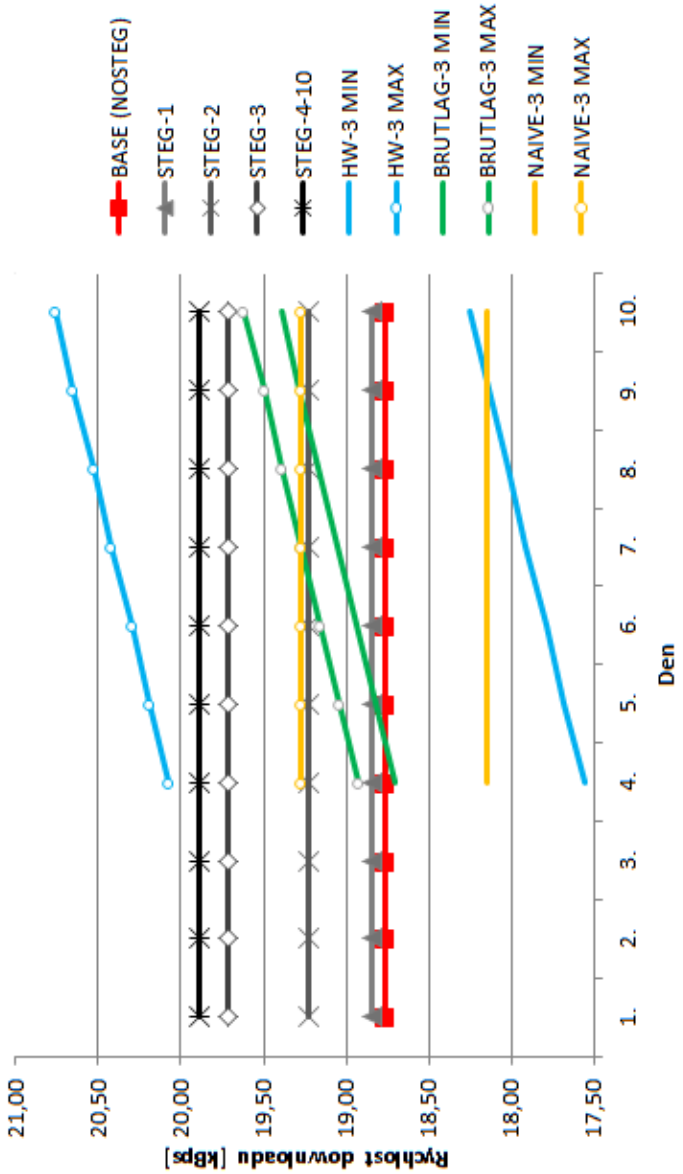
Obrázek 29: Naive - BASE + predikovaný model. Deviation scale: 1, 2, 3, 6 - detail



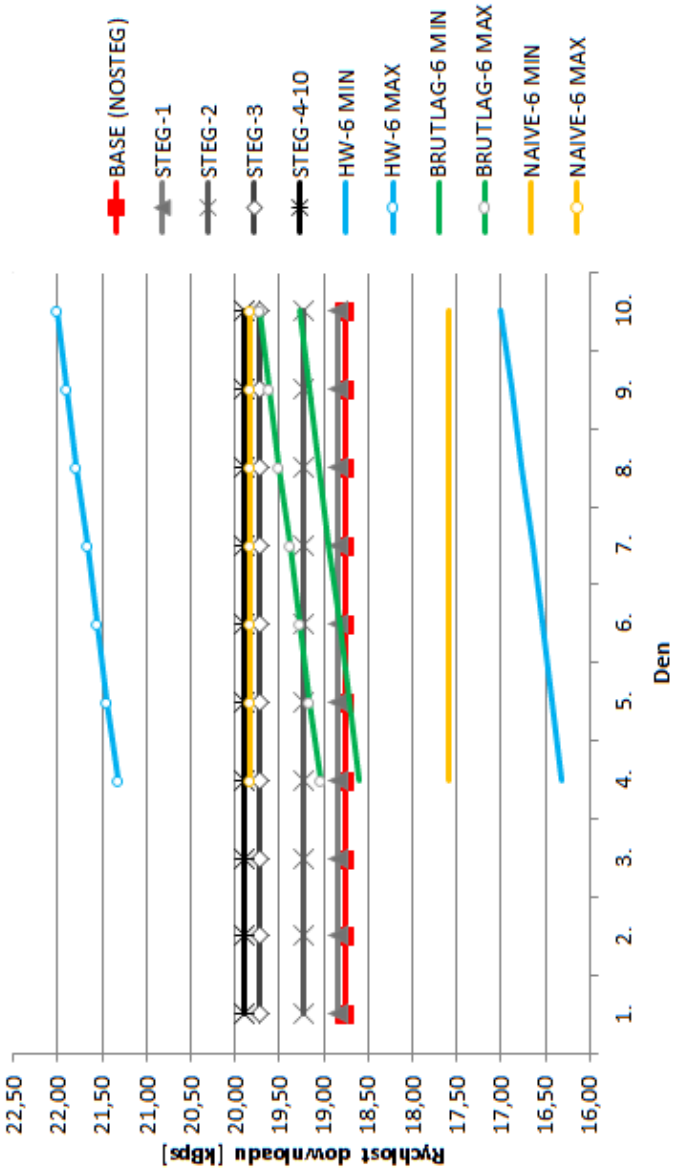
Obrázek 30: Schopnost detekce jednotlivých algoritmů v čase 00:10:01. Deviation scale: 1



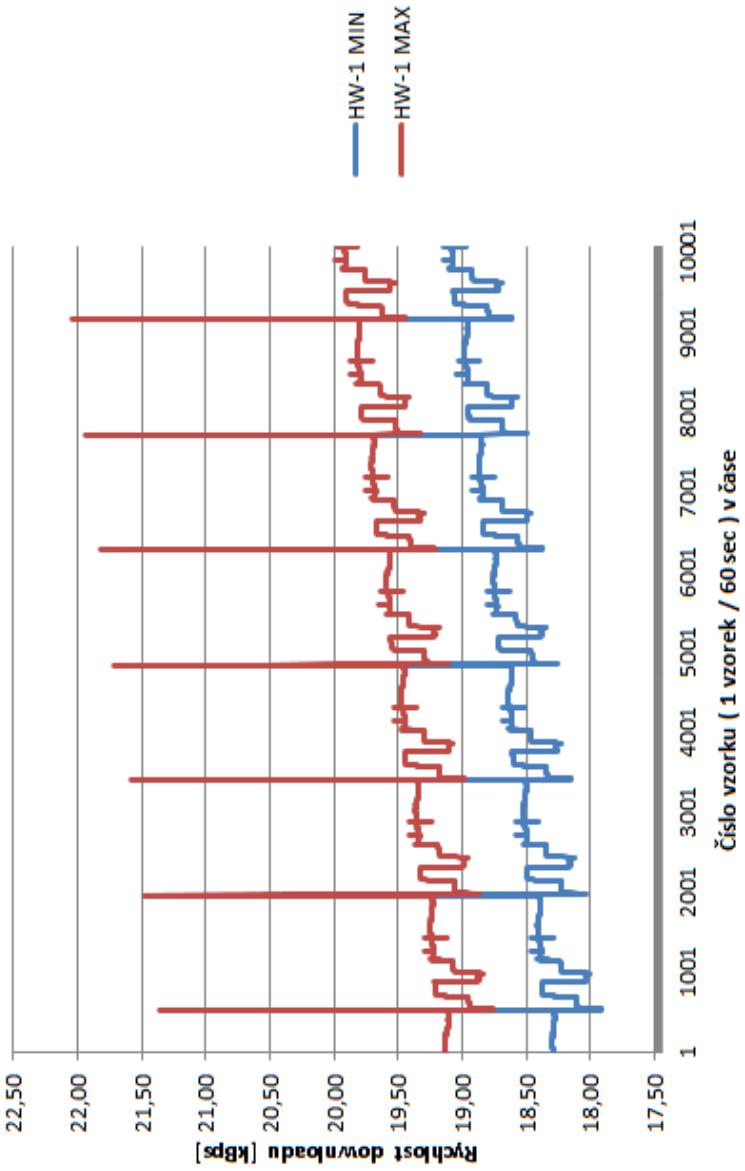
Obrázek 31: Schopnost detekce jednotlivých algoritmů v čase 00:10:01. Deviation scale: 2



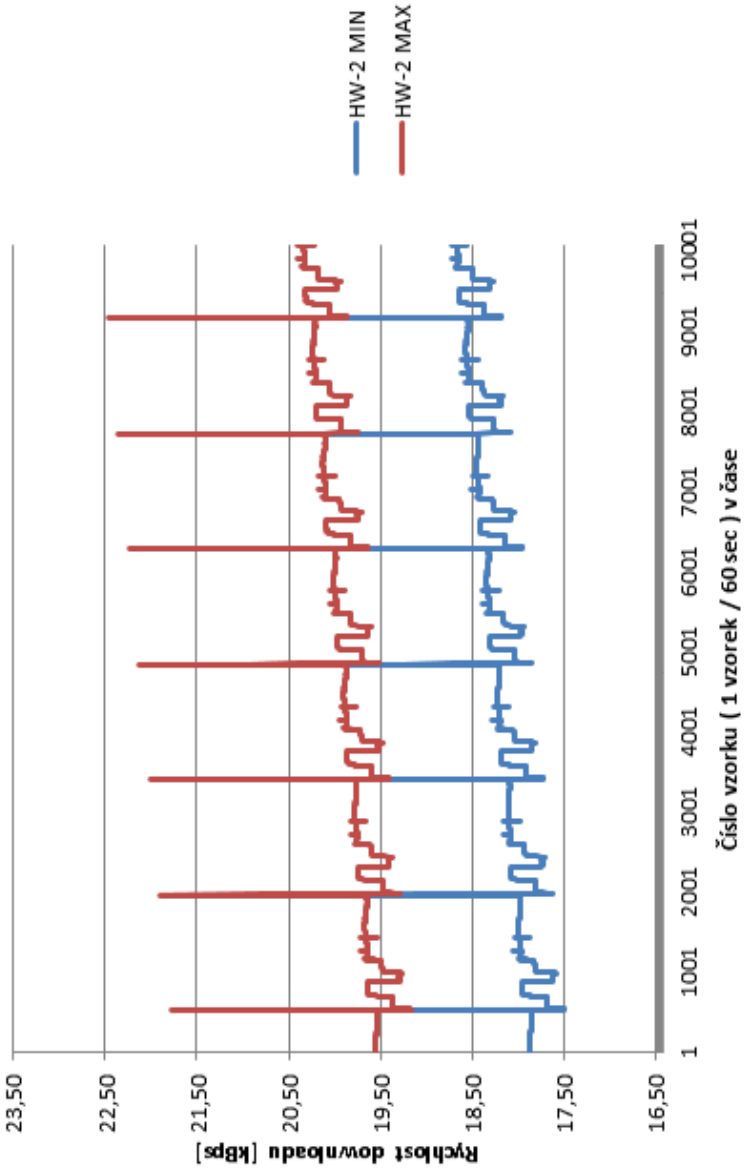
Obrázek 32: Schopnost detekce jednotlivých algoritmů v čase 00:10:01. Deviation scale: 3



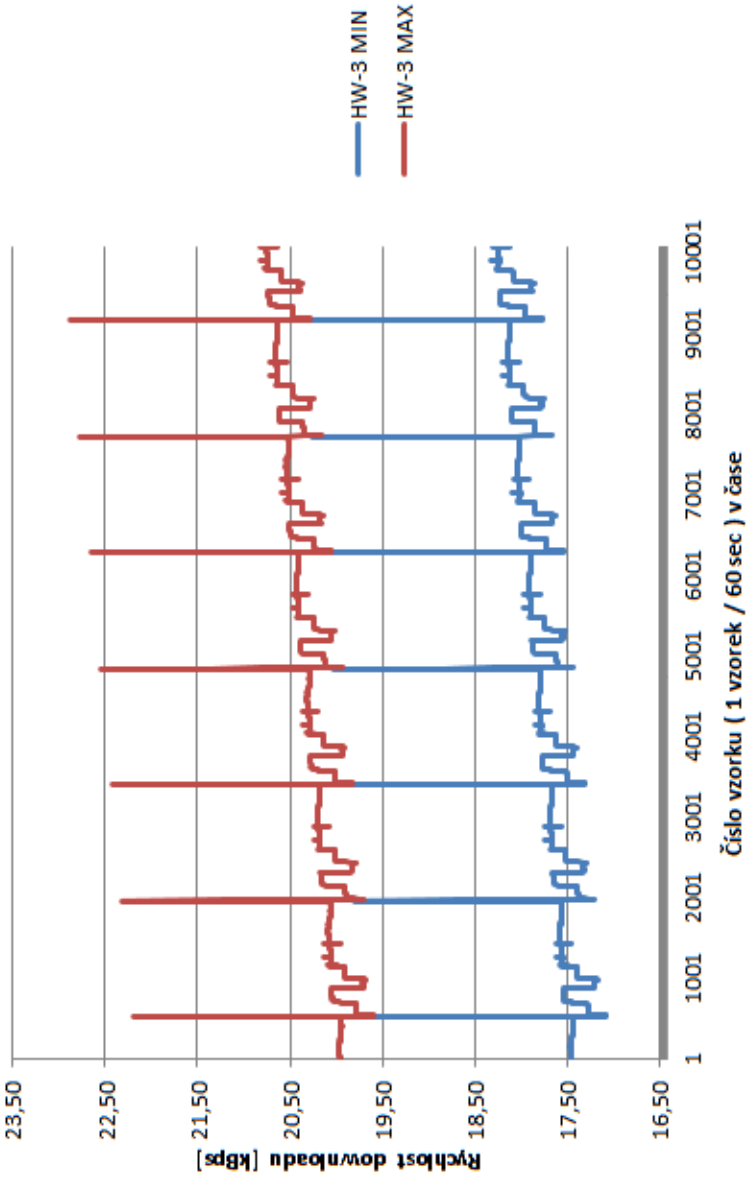
Obrázek 33: Schopnost detekce jednotlivých algoritmů v čase 00:10:01. Deviation scale: 4



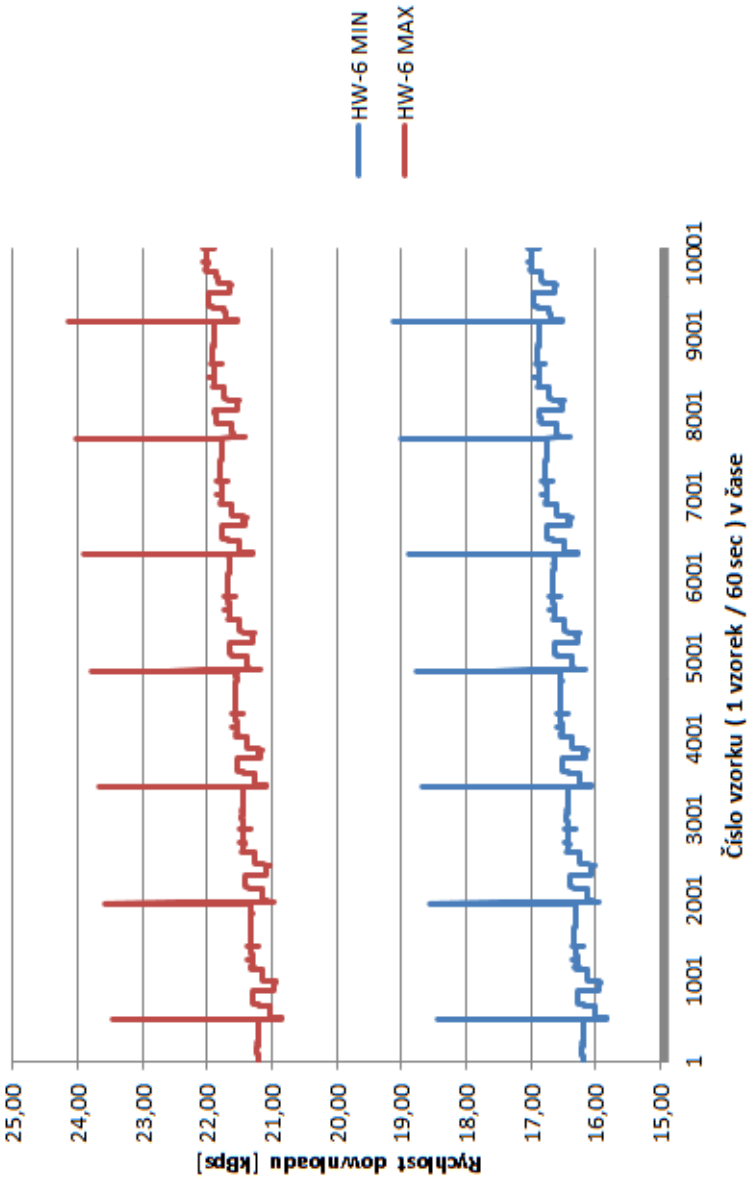
Obrázek 34: Holt-Winters - predikovaný model. Deviation scale: 1



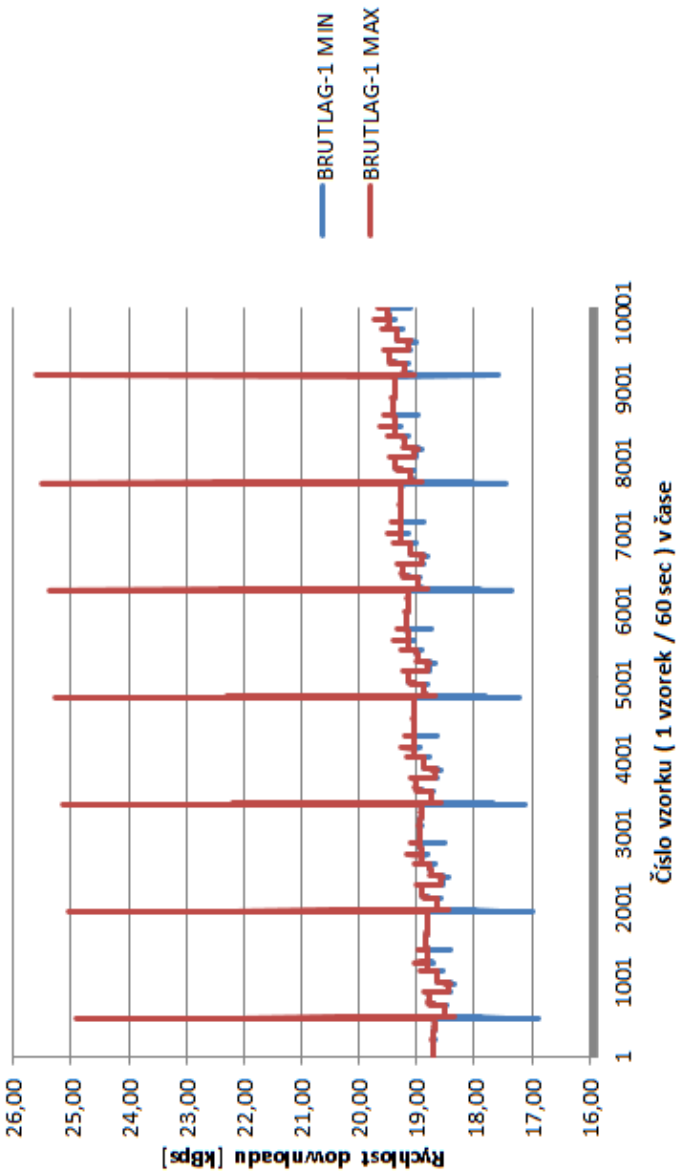
Obrázek 35: Holt-Winters - predikovaný model. Deviation scale: 2



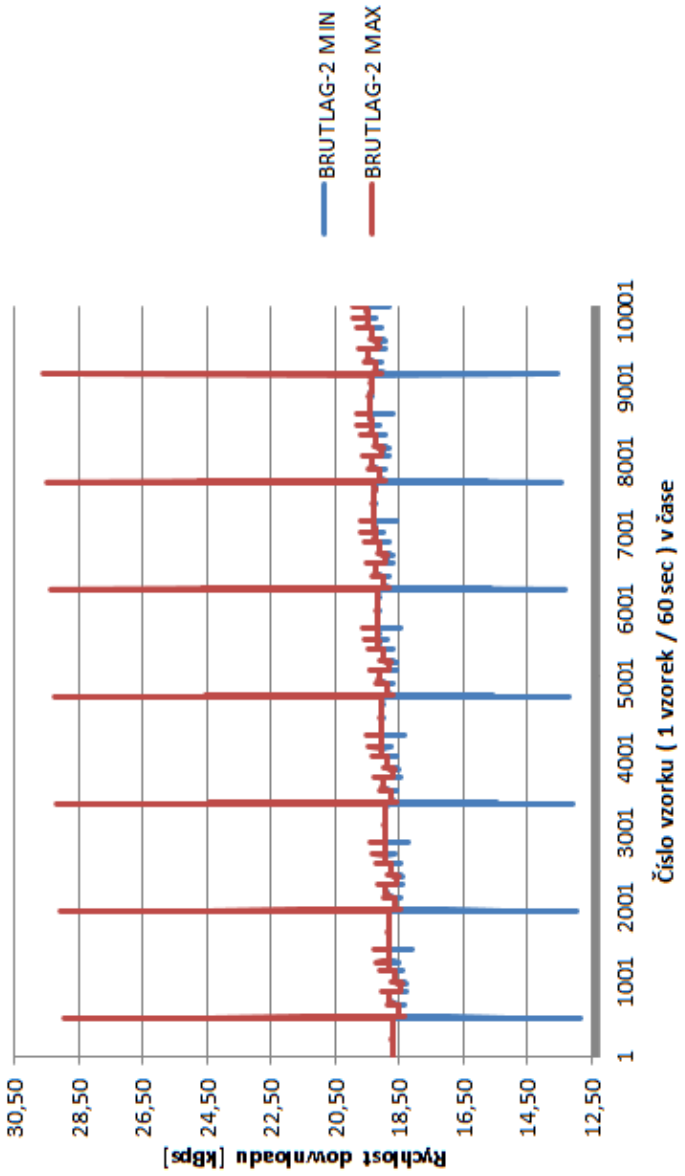
Obrázek 36: Holt-Winters - predikovaný model. Deviation scale: 3



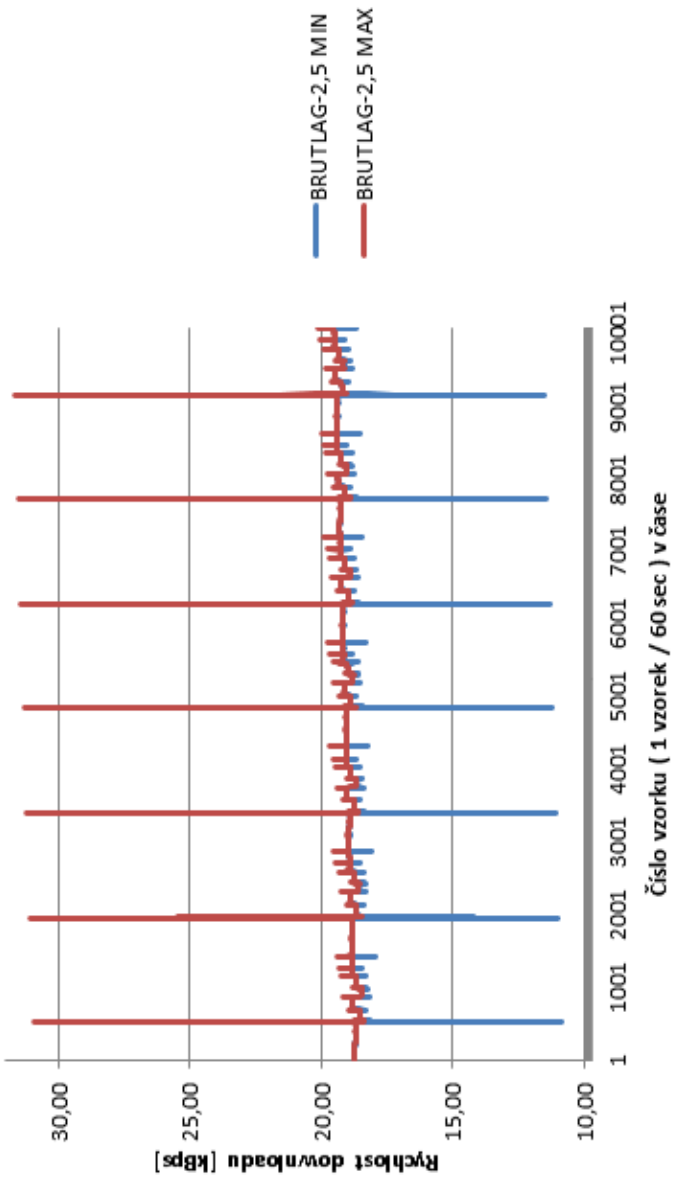
Obrázek 37: Holt-Winters - predikovaný model. Deviation scale: 6



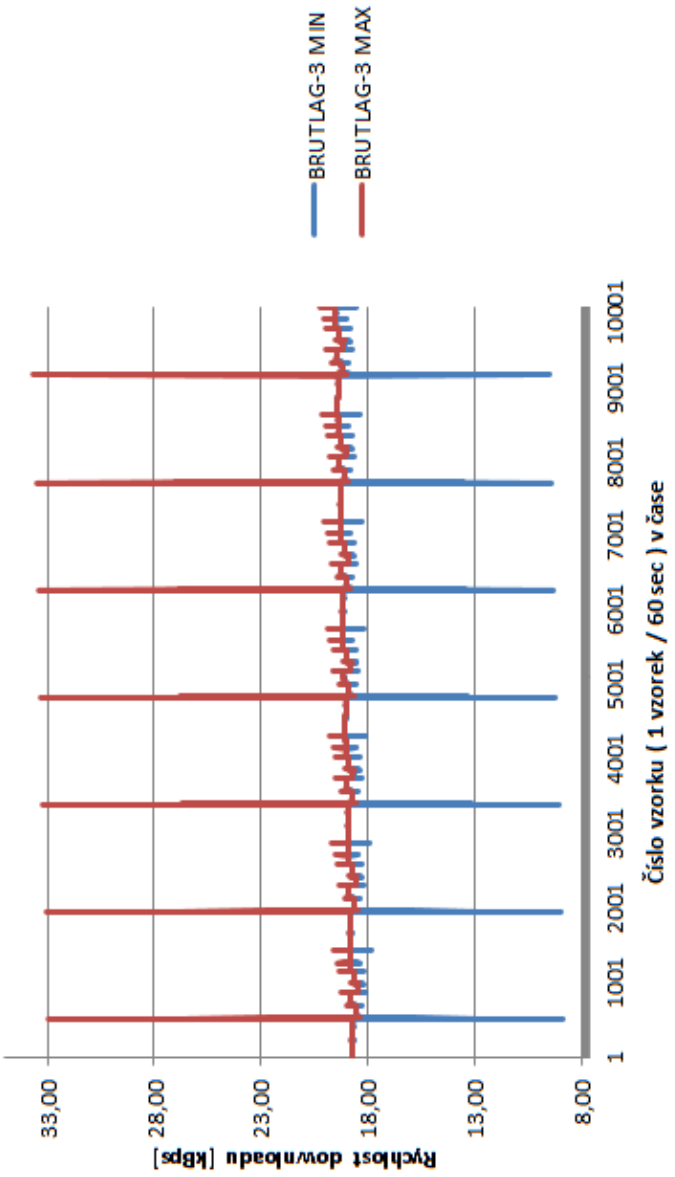
Obrázek 38: Holt-Winters:Brutlag - predikovaný model. Deviation scale: 1



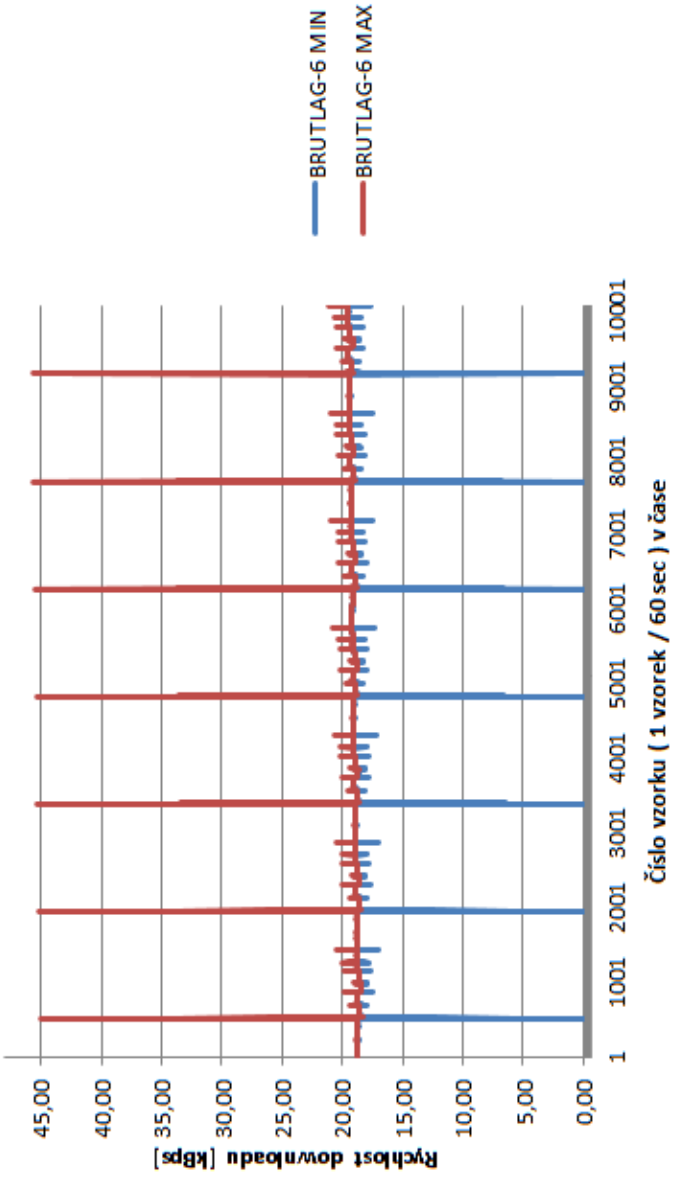
Obrázek 39: Holt-Winters:Brutlag - predikovaný model. Deviation scale: 2



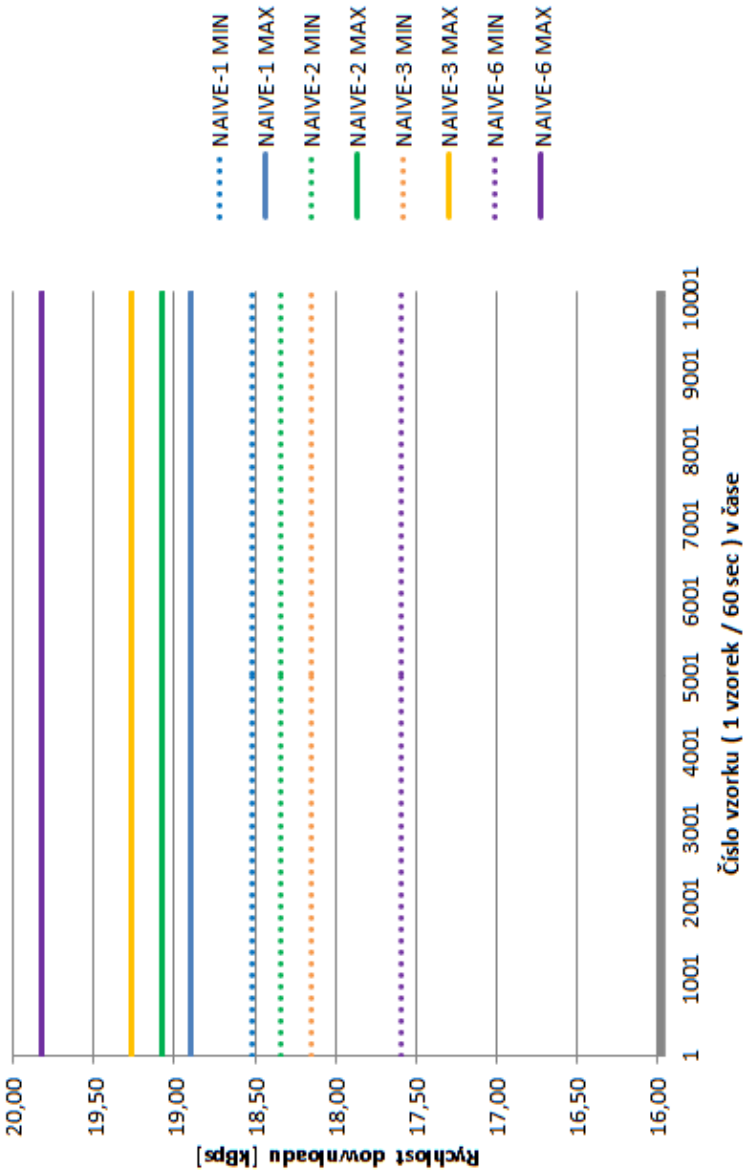
Obrázek 40: Holt-Winters:Brutlag - predikovaný model. Deviation scale: 2.5



Obrázek 41: Holt-Winters:Brutlag - predikovaný model. Deviation scale: 3



Obrázek 42: Holt-Winters:Brutlag - predikovaný model. Deviation scale: 6



Obrázek 43: Naive - predikovaný model. Deviation scale: 1, 2, 3, 6

C SIPp scénáře

Zde jsou umístěny vybrané SIPp scénaře, které jsem vytvořil v rámci této práce.

```
<?xml version="1.0" encoding="ISO−8859−1" ?>
<!DOCTYPE scenario SYSTEM "sipp.dtd">
<scenario name="SERVER">
  <recv request="INVITE" crlf="true"  rrs="true">
  </recv>

  <send>
    <![CDATA[
      SIP/2.0 100 Trying
      [last_Via :]
      [last_From:]
      [last_To :]; tag=[pid]SIPpTag01[call_number]
      [last_Call−ID:]
      [last_CSeq:]
      Contact: <sip:[ local_ip ]:[ local_port ]; transport=[transport]>
    ]]>
  </send>

  <send>
    <![CDATA[
      SIP/2.0 180 Ringing
      [last_Via :]
      [last_From:]
      [last_To :]; tag=[pid]SIPpTag01[call_number]
      [last_Call−ID:]
      [last_CSeq:]
      Contact: <sip:[ local_ip ]:[ local_port ]; transport=[transport]>
    ]]>
  </send>

  <send>
    <![CDATA[
      SIP/2.0 200 OK
      [last_Via :]
      [last_From:]
      [last_To :]; tag=[pid]SIPpTag01[call_number]
      [last_Call−ID:]
      [last_CSeq:]
      Contact: <sip:[ local_ip ]:[ local_port ]; transport=[transport]>
    ]]>
  </send>

  <recv request="ACK"
        rtd="true"
        crlf ="true">
  </recv>

  <recv request="OPTIONS">
  </recv>

  <send>
    <![CDATA[
      SIP/2.0 200 OK
```

```
      [last_Via :]
      [last_From:]
      [last_To :]
      [last_Call-ID:]
      [last_CSeq:]
      Contact: <sip:[local_ip]:[local_port];transport=[transport]>
    ]]>
  </send>

  <recv request="INFO">
  </recv>

  <send>
    <![CDATA[
      SIP/2.0 200 OK
      [last_Via :]
      [last_From:]
      [last_To :]
      [last_Call-ID:]
      [last_CSeq:]
    ]]>
  </send>

  <recv request="BYE">
  </recv>

  <send>
    <![CDATA[
      SIP/2.0 200 OK
      [last_Via :]
      [last_From:]
      [last_To :]
      [last_Call-ID:]
      [last_CSeq:]
      Contact: <sip:[local_ip]:[local_port];transport=[transport]>
    ]]>
  </send>

  <!-- definition of the response time repartition table (unit is ms) -->
  <ResponseTimeRepartition value="10,␣20,␣30,␣40,␣50,␣100,␣150,␣200"/>

  <!-- definition of the call length repartition table (unit is ms) -->
  <CallLengthRepartition value="10,␣50,␣100,␣500,␣1000,␣5000,␣10000"/>
</scenario>
```

Výpis 30: SIPp scénář SERVER

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE scenario SYSTEM "sipp.dtd">
<scenario name="CLIENT-NOSTEG">
  <send retrans="500">
    <![CDATA[
      INVITE sip:[service]@[remote_ip]:[remote_port] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      From: sipp <sip:sipp@[local_ip]:[local_port]>;tag=[pid]SIPpTag09[call_number]
      To: sut <sip:[service]@[remote_ip]:[remote_port]>
      Call-ID: [call_id]
      CSeq: 1 INVITE
      Contact: sip:sipp@[local_ip]:[local_port]
      Max-Forwards: 70
      Subject: Performance Test
    ]]>
  </send>

  <recv response="100" optional="true">
  </recv>

  <recv response="180" optional="true">
  </recv>

  <recv response="200" rtd="true" crlf="true">
  </recv>

  <send>
    <![CDATA[
      ACK sip:[service]@[remote_ip]:[remote_port] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      From: sipp <sip:sipp@[local_ip]:[local_port]>;tag=[pid]SIPpTag09[call_number]
      To: sut <sip:[service]@[remote_ip]:[remote_port]>[peer_tag_param]
      Call-ID: [call_id]
      CSeq: 1 ACK
      Contact: sip:sipp@[local_ip]:[local_port]
      Max-Forwards: 70
      Subject: Performance Test
    ]]>
  </send>

  <send>
    <![CDATA[
      OPTIONS sip:[service]@[remote_ip] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      Max-Forwards: 70
      To: <sip:[service]@[remote_ip]>
      From: sipp <sip:sipp@[local_ip]:[local_port]>;tag=[call_number]
      Call-ID: [call_id]
      CSeq: 1 OPTIONS
      Contact: <sip:sipp@[local_ip]:[local_port]>
      User-Agent:
    ]]>
  </send>
```

```
  <recv response="200" crlf="true">
  </recv>

  <send>
    <![CDATA[
      INFO sip:[service]@[remote_ip] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      To: <sip:[service]@[remote_ip]>
      From: sipp <sip:sipp@[local_ip]:[local_port]>;tag=[call_number]
      Call-ID: [call_id]
      CSeq: 1 INFO
    ]]>
  </send>
  <recv response="200" crlf="true">
  </recv>

  <send>
    <![CDATA[
      BYE sip:[service]@[remote_ip]:[remote_port] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      From: sipp <sip:sipp@[local_ip]:[local_port]>;tag=[pid]SIPpTag09[call_number]
      To: sut <sip:[service]@[remote_ip]:[remote_port]>[peer_tag_param]
      Call-ID: [call_id]
      CSeq: 2 BYE
      Contact: sip:sipp@[local_ip]:[local_port]
      Max-Forwards: 70
      Subject: Performance Test
    ]]>
  </send>

  <recv response="200" crlf="true">
  </recv>

  <!-- definition of the response time repartition table (unit is ms) -->
  <ResponseTimeRepartition value="10,␣20,␣30,␣40,␣50,␣100,␣150,␣200"/>

  <!-- definition of the call length repartition table (unit is ms) -->
  <CallLengthRepartition value="10,␣50,␣100,␣500,␣1000,␣5000,␣10000"/>
</scenario>
```

Výpis 31: SIPp scénář CLIENT NOSTEG

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE scenario SYSTEM "sipp.dtd">
<scenario name="CLIENT-STEG-1">
  <send retrans="500">
    <![CDATA[
      INVITE sip:[service]@[remote_ip]:[remote_port] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip ]:[ local_port ];branch=[branch]
      From: sipp <sip:sipp@[local_ip ]:[ local_port ]>;tag=[pid]SIPpTag09[call_number]
      To: sut <sip :[service]@[remote_ip]:[remote_port]>
      Call-ID: [ call_id ]
      CSeq: 1 INVITE
      Contact: sip :sipp@[local_ip ]:[ local_port ]
      Max-Forwards: 70
      Subject: Performance Test
    ]]>
  </send>

  <recv response="100" optional="true">
  </recv>

  <recv response="180" optional="true">
  </recv>

  <recv response="200" rtd="true" crlf="true">
  </recv>

  <send>
    <![CDATA[
      ACK sip:[service]@[remote_ip]:[remote_port] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip ]:[ local_port ];branch=[branch]
      From: sipp <sip:sipp@[local_ip ]:[ local_port ]>;tag=[pid]SIPpTag09[call_number]
      To: sut <sip :[service]@[remote_ip]:[remote_port]>[peer_tag_param]
      Call-ID: [ call_id ]
      CSeq: 1 ACK
      Contact: sip :sipp@[local_ip ]:[ local_port ]
      Max-Forwards: 70
      Subject: Performance Test
    ]]>
  </send>

  <send>
    <![CDATA[
      OPTIONS sip:[service]@[remote_ip] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip ]:[ local_port ];branch=[branch]
      Max-Forwards: 70
      To: <sip :[service]@[remote_ip]>
      From: sipp <sip:sipp@[local_ip ]:[ local_port ]>;tag=[call_number]
      Call-ID: [ call_id ]
      CSeq: 1 OPTIONS
      Contact: <sip:sipp@[local_ip ]:[ local_port ]>
      User-Agent: Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo
          ligula eget dolor. Aenean m
    ]]>
```

```
  </send>
  <recv response="200" crlf="true">
  </recv>

  <send>
    <![CDATA[
      INFO sip:[service]@[remote_ip] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      To: <sip:[service]@[remote_ip]>
      From: sipp <sip:sipp@[local_ip]:[local_port]>;tag=[call_number]
      Call-ID: [call_id]
      CSeq: 1 INFO

    ]]>
  </send>
  <recv response="200" crlf="true">
  </recv>

  <send>
    <![CDATA[
      BYE sip:[service]@[remote_ip]:[remote_port] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      From: sipp <sip:sipp@[local_ip]:[local_port]>;tag=[pid]SIPpTag09[call_number]
      To: sut <sip:[service]@[remote_ip]:[remote_port]>[peer_tag_param]
      Call-ID: [call_id]
      CSeq: 2 BYE
      Contact: sip:sipp@[local_ip]:[local_port]
      Max-Forwards: 70
      Subject: Performance Test
    ]]>
  </send>

  <recv response="200" crlf="true">
  </recv>

  <!-- definition of the response time repartition table (unit is ms) -->
  <ResponseTimeRepartition value="10,␣20,␣30,␣40,␣50,␣100,␣150,␣200"/>

  <!-- definition of the call length repartition table (unit is ms) -->
  <CallLengthRepartition value="10,␣50,␣100,␣500,␣1000,␣5000,␣10000"/>
</scenario>
```

Výpis 32: SIPp scénář CLIENT STEG 1

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE scenario SYSTEM "sipp.dtd">
<scenario name="CLIENT-STEG-2">
  <send retrans="500">
    <![CDATA[
      INVITE sip:[service]@[remote_ip]:[remote_port] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      From: sipp <sip:sipp@[local_ip]:[local_port]>;tag=[pid]SIPpTag09[call_number]
      To: sut <sip:[service]@[remote_ip]:[remote_port]>
      Call-ID: [call_id]
      CSeq: 1 INVITE
      Contact: sip:sipp@[local_ip]:[local_port]
      Max-Forwards: 70
      Subject: Performance Test
    ]]>
  </send>

  <recv response="100" optional="true">
  </recv>

  <recv response="180" optional="true">
  </recv>

  <recv response="200" rtd="true" crlf="true">
  </recv>

  <send>
    <![CDATA[
      ACK sip:[service]@[remote_ip]:[remote_port] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      From: sipp <sip:sipp@[local_ip]:[local_port]>;tag=[pid]SIPpTag09[call_number]
      To: sut <sip:[service]@[remote_ip]:[remote_port]>[peer_tag_param]
      Call-ID: [call_id]
      CSeq: 1 ACK
      Contact: sip:sipp@[local_ip]:[local_port]
      Max-Forwards: 70
      Subject: Performance Test
    ]]>
  </send>

  <send>
    <![CDATA[
      OPTIONS sip:[service]@[remote_ip] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      Max-Forwards: 70
      To: <sip:[service]@[remote_ip]>
      From: sipp <sip:sipp@[local_ip]:[local_port]>;tag=[call_number]
      Call-ID: [call_id]
      CSeq: 1 OPTIONS
      Contact: <sip:sipp@[local_ip]:[local_port]>
      User-Agent: Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo
          ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient
          montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium
```

```
            quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec
            , vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo.
            Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibu
    ]]>
  </send>
  <recv response="200" crlf="true">
  </recv>

  <send>
    <![CDATA[
      INFO sip:[service]@[remote_ip] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      To: <sip:[service]@[remote_ip]>
      From: sipp <sip:sipp@[local_ip]:[local_port]>;tag=[call_number]
      Call-ID: [call_id]
      CSeq: 1 INFO

    ]]>
  </send>
  <recv response="200" crlf="true">
  </recv>

  <send>
    <![CDATA[
      BYE sip:[service]@[remote_ip]:[remote_port] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      From: sipp <sip:sipp@[local_ip]:[local_port]>;tag=[pid]SIPpTag09[call_number]
      To: sut <sip:[service]@[remote_ip]:[remote_port]>[peer_tag_param]
      Call-ID: [call_id]
      CSeq: 2 BYE
      Contact: sip:sipp@[local_ip]:[local_port]
      Max-Forwards: 70
      Subject: Performance Test
    ]]>
  </send>

  <recv response="200" crlf="true">
  </recv>

  <!-- definition of the response time repartition table (unit is ms) -->
  <ResponseTimeRepartition value="10,␣20,␣30,␣40,␣50,␣100,␣150,␣200"/>

  <!-- definition of the call length repartition table (unit is ms) -->
  <CallLengthRepartition value="10,␣50,␣100,␣500,␣1000,␣5000,␣10000"/>
</scenario>
```

Výpis 33: SIPp scénář CLIENT STEG 2

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE scenario SYSTEM "sipp.dtd">
<scenario name="CLIENT-STEG-3">
  <send retrans="500">
    <![CDATA[
      INVITE sip:[service]@[remote_ip]:[remote_port] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      From: sipp <sip:sipp@[local_ip]:[local_port]>;tag=[pid]SIPpTag09[call_number]
      To: sut <sip:[service]@[remote_ip]:[remote_port]>
      Call-ID: [call_id]
      CSeq: 1 INVITE
      Contact: sip:sipp@[local_ip]:[local_port]
      Max-Forwards: 70
      Subject: Performance Test
    ]]>
  </send>

  <recv response="100" optional="true">
  </recv>

  <recv response="180" optional="true">
  </recv>

  <recv response="200" rtd="true" crlf="true">
  </recv>

  <send>
    <![CDATA[
      ACK sip:[service]@[remote_ip]:[remote_port] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      From: sipp <sip:sipp@[local_ip]:[local_port]>;tag=[pid]SIPpTag09[call_number]
      To: sut <sip:[service]@[remote_ip]:[remote_port]>[peer_tag_param]
      Call-ID: [call_id]
      CSeq: 1 ACK
      Contact: sip:sipp@[local_ip]:[local_port]
      Max-Forwards: 70
      Subject: Performance Test
    ]]>
  </send>

  <send>
    <![CDATA[
      OPTIONS sip:[service]@[remote_ip] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      Max-Forwards: 70
      To: <sip:[service]@[remote_ip]>
      From: sipp <sip:sipp@[local_ip]:[local_port]>;tag=[call_number]
      Call-ID: [call_id]
      CSeq: 1 OPTIONS
      Contact: <sip:sipp@[local_ip]:[local_port]>
      User-Agent: Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Aenean commodo
          ligula eget dolor. Aenean massa. Cum sociis natoque penatibus et magnis dis parturient
          montes, nascetur ridiculus mus. Donec quam felis, ultricies nec, pellentesque eu, pretium
```

```
            quis, sem. Nulla consequat massa quis enim. Donec pede justo, fringilla vel, aliquet nec
            , vulputate eget, arcu. In enim justo, rhoncus ut, imperdiet a, venenatis vitae, justo.
            Nullam dictum felis eu pede mollis pretium. Integer tincidunt. Cras dapibus. Vivamus
            elementum semper nisi. Aenean vulputate eleifend tellus. Aenean leo ligula, porttitor eu,
            consequat vitae, eleifend ac, enim. Aliquam lorem ante, dapibus in, viverra quis,
            feugiat a, tellus. Phasellus viverra nulla ut metus varius laoreet. Quisque rutrum.
            Aenean imperdiet. Etiam ultricies nisi vel augue. Curabitur ullamcorper ultricies nisi.
            Nam eget dui. Etiam rhoncus. Maecenas tempus, tellus eget condimentum rhoncus, sem
            quam semper libero, sit amet adipiscing sem neque sed ipsum. N
    ]]>
  </send>
  <recv response="200" crlf="true">
  </recv>

  <send>
    <![CDATA[
      INFO sip:[service]@[remote_ip] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      To: <sip:[service]@[remote_ip]>
      From: sipp <sip:sipp@[local_ip]:[local_port]>;tag=[call_number]
      Call-ID: [call_id]
      CSeq: 1 INFO
    ]]>
  </send>
  <recv response="200" crlf="true">
  </recv>

  <send>
    <![CDATA[
      BYE sip:[service]@[remote_ip]:[remote_port] SIP/2.0
      Via: SIP/2.0/[transport] [local_ip]:[local_port];branch=[branch]
      From: sipp <sip:sipp@[local_ip]:[local_port]>;tag=[pid]SIPpTag09[call_number]
      To: sut <sip:[service]@[remote_ip]:[remote_port]>[peer_tag_param]
      Call-ID: [call_id]
      CSeq: 2 BYE
      Contact: sip:sipp@[local_ip]:[local_port]
      Max-Forwards: 70
      Subject: Performance Test
    ]]>
  </send>

  <recv response="200" crlf="true">
  </recv>

  <!-- definition of the response time repartition table (unit is ms) -->
  <ResponseTimeRepartition value="10,␣20,␣30,␣40,␣50,␣100,␣150,␣200"/>

  <!-- definition of the call length repartition table (unit is ms) -->
  <CallLengthRepartition value="10,␣50,␣100,␣500,␣1000,␣5000,␣10000"/>
</scenario>
```

Výpis 34: SIPp scénář CLIENT STEG 3

D Příloha na CD/DVD

D.1 Obsah přiloženého CD

- SIPp scénáře
- zaznamenaný (BASE) model chování sítě
- predikované modely chování sítě