VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

ANALÝZA A DETEKCE TYPU MULTIMEDIÁLNÍCH DAT V PROVOZU RTP

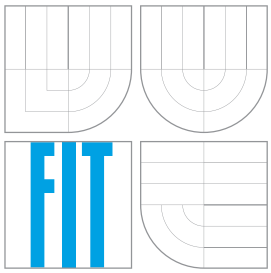
DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. MARTIN KMEŤ

BRNO 2014

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INFORMAČNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INFORMATION SYSTEMS

ANALÝZA A DETEKCE TYPU MULTIMEDIÁLNÍCH DAT V PROVOZU RTP

ANALYSIS AND DETECTION OF MULTIMEDIA TYPES IN RTP TRAFFIC

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

Bc. MARTIN KMEŤ

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. PETR MATOUŠEK, Ph.D.

BRNO 2014

Abstrakt

Tato práce se věnuje problematice detekce kodeku použitého při kódování hlasových dat nesených protokolem RTP bez dostupnosti informací nesených signalizačními protokoly ve VoIP aplikacích. Jejím hlavním cílem je navrhnout a implementovat rychlý algoritmus na detekci kodeků používaných pro přenos hlasu protokolem RTP. Tento algoritmus by měl být dostatečně rychlý pro použití jak na off-line analýzu zachycených dat, tak na on-line analýzu v reálném čase. Pro průzkum možností byly porovnány dva přístupy detekce. Pro samotné řešení problému byla zvolená detekce pomocí charakteristických znaků kodeku. V rámci řešení byla provedena analýza dat, implementace aplikace a následné testování na datech.

Abstract

This thesis deals with the issues of detecting the codec used for the encoding of voice data carried by the RTP protocol without having access to the information carried by signalisation protocols in VoIP applications. Its main goal is to create and implement a fast algorithm for detecting the codec used for voice transfer via the RTP protocol. This algorithm should be fast enough to be used for offline analysis of captured data as well as for real-time online analysis. For research of possibilities were compared two approaches of detection. Detection by the characteristics of the codecs was chosen to solve the problem itself. Within the solution was performed data analysis and implementation of the application followed by testing on data.

Klíčová slova

audio, detekce, kodek, stream, multimédiá, RTP, RTCP, real-time přenos

Keywords

audio, codec, detection, stream, multimedia, RTP, RTCP, real-time transfer

Citace

Martin Kmeť: Analýza a detekce typu multimediálních dat v provozu RTP, diplomová práce, Brno, FIT VUT v Brně, 2014

Analýza a detekce typu multimediálních dat v provozu RTP

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením pana Ing. Petra Matouška, Ph.D. a že jsem uvedl všechny použité literární prameny a publikace.

.....
Martin Kmeť
28. května 2014

Poděkování

Týmto by som chcel poďakovať Ing. Petrovi Matouškovi, Ph.D. za pomoc a rady, ktoré viedli ku vzniku tejto práce a doc. Ing. Miroslavi Vozňákovi, Ph.D. za umožnenie absolvovania školenia a prístup k sieťovému testeru Ixia.

Obsah

Kapitola 1

Úvod

Protokol RTP bol navrhnutý pre prenos rôzneho druhu dát s potrebou prenosu v reálnom čase. V dnešnej dobe je najviac používaný na prenos zvuku a videa pri VoIP telefónii, streamovaní videa alebo videokonferenciách.

Jeden z problémov pri použití tohto protokolu však nastáva pri pokuse o detekciu prenosu dát pomocou protokolu RTP v sieťovej komunikácii. Potreba rozpoznať RTP však môže nastať z viacerých dôvodov. Potrebné môže byť napríklad povoliť prenos cez porty na firewalle, klasifikovať prenos pre zaistenie kvality služieb, rekonštruovať hovor z odchytenej sieťovej komunikácie atď. Tomuto problému sa venuje práca [3].

Pri pokuse o rekonštrukciu hovorov z odchytených dát môžeme naraziť aj na ďalší výrazný problém, a to je identifikácia kodeku použitého na kompresiu prenášaného hlasu alebo videa. V prípade použitia jedného zo staticky mapovaných kodekov je možné toto určiť podľa čísla v položke `payload type` v RTP hlavičke, ale v prípade niektorých kodekov sa používa dynamické mapovanie kodeku na túto hodnotu. V tomto prípade sa informácie o kodeku prenášajú len prostredníctvom jedného zo signalizačných protokolov a bez jeho použitia sa musí použitý kodek identifikovať na základe charakteru prenášaných dát.

Práca teda popisuje teoretické základy o prenose pomocou protokolu RTP a možnosti jeho detekcie. V ďalších kapitolách sa venuje možnostiam voľby kodekov a ich vplyvu na RTP prevádzku. Práca sa však venuje najmä vypracovaniu postupu pre identifikáciu kodeku neseného protokolom RTP bez použitia dát nesených signalizačnými protokolmi, ale aj vývojom nástroja na automatickú detekciu RTP prenosu a jeho identifikáciu, vrátane identifikácie použitého kodeku.

V práci sú využité informácie zo semestrálneho projektu, na ktorý táto práca naväzuje. Prevzatá je veľká časť teórie t.j. kapitoly 2 a 3 a výskum ohľadom možností detekcie kodeku neseného protokolom RTP t.j. kapitola 5.

U čitateľa sa predpokladá základná znalosť počítačových sietí a pojmov s nimi súvisiacich.

1.1 Ciele práce

Cieľom tejto práce je rozšíriť existujúci spôsob detekcie RTP o ďalšie vlastnosti, a to o extrakciu dodatočných informácií dostupných z protokolov RTP a RTCP. Najmä však vypracovať metódu na automatickú detekciu kodeku bez použitia informácií nesených signalizačnými protokolmi.

1.2 Riešenie

V rámci tejto práce bude vyvinutý algoritmus, ktorý bude následne implementovaný do aplikácie pre experimentálne overenie funkčnosti algoritmu. Táto aplikácia bude založená na existujúcom algoritme na detekciu RTP streamov a bude zameraná na rýchlosť implementácie, ktorá by mala byť dostatočná aj pre detekciu v reálnom čase.

Kapitola 2

Protokoly RTP a RTCP

Oba protokoly boli prvýkrát publikované v januári 1996 v dokumente RFC 1889, ktorý bol v júli 2003 nahradený dokumentom RFC 3550[6]. Z druhého z uvedených dokumentov je vypracovaná veľká časť tejto kapitoly.

V dokumente sú popisované oba protokoly, a to RTP (Real-time Transport Protocol) na prenos dát v reálnom čase a RTCP (RTP Control Protocol) na monitorovanie kvality a šírenie informácií o účastníkoch streamu.

RTP streamy väčšinou využívajú služby transportného protokolu UDP[4], ktorý je vhodný najmä vďaka podpore multicastu a malej veľkosti hlavičky. Toto je pri RTP streamoch dôležitá vlastnosť, keďže RTP streamy majú väčšinou charakter veľkého množstva pomerne malých paketov. RTP však môže využiť služby ľubovoľného transportného protokolu, ktorý mu umožní oddeliť viac sedení a tok RTP a RTCP dát, čo UDP umožňuje vďaka použitiu portov.

Komunikácia prostredníctvom protokolu RTP prebieha v takzvaných sedeniach. V jednom sedení je vždy prenášaný len jeden stream jedného typu média, ale jeden účastník sa môže v jednom okamihu podieľať na viacerých sedeniach, ktoré sú navzájom nezávislé. Teda napríklad v prípade videokonferencií je video a audio prenášané väčšinou v dvoch samostatných streamoch, z ktorých každý vytvorí vlastné sedenie.

V RTP sedení sa môže vyskytovať viac typov zariadení:

Synchronizačný zdroj (SSRC – Synchronization Source) – Zdroj streamu (alebo RTP mixer), ktorý určuje synchronizáciu paketov pre príjemcov streamu. V jednom RTP sedení sa môže vyskytovať len jeden synchronizačný zdroj.

Prispievajúci zdroj (CSRC – Contributing Source) – Jeden zo zdrojov zasielajúcich dáta mixéru, ktorý z nich skladá výsledný stream. Mixér vkladá jednotlivé identifikátory synchronizačných zdrojov každého streamu, z ktorých skladá výsledný stream a vkladá ich do odosielaných paketov ako zoznam prispievajúcich zdrojov.

Koncový systém – Zariadenie, ktoré odosiela alebo prijíma stream. Do tejto skupiny patria aj SSRC a CSRC.

Mixér – Prechodný systém, ktorý prijíma RTP pakety od viacerých zdrojov a kombinuje ich, pričom môže meniť ich formát. Keďže jednotlivé zdroje nemusia byť synchronizované, stará sa aj o synchronizáciu dát a vytvára vlastné časovanie. Po skombinovaní jednotlivých paketov preposiela výsledný paket ďalej, pričom je uvedený ako synchronizačný zdroj.

Prekladač – Zariadenie, ktoré preposiela RTP pakety bez zmeny synchronizačného zdroja. Môže napríklad meniť formát dát bez kombinovania viacerých zdrojov, prevádzať multicast na unicast atď.

Monitor – Zariadenie prijímajúce RTCP dáta, zisťujúce kvalitu hovoru a zostavujúce štatistiku. Býva vstavané v aplikácii využívajúcej RTP, ale môže byť implementované v samostatnej aplikácii alebo môže existovať aj ako samostatné zariadenie.

RTP podporuje komunikáciu typu „many-to-many“, čo znamená, že v jednom okamihu, môže mať jedno sedenie viac odosielateľov aj príjemcov. O možnosť príjmu sa musí postarať protokol transportnej vrstvy (prevažne UDP pomocou multicastu). Možnosť viacerých odosielateľov je riešená pomocou RTP mixérov.

RTP nebolo navrhnuté pre konkrétny typ dát, ako napríklad prenos audia alebo videa, pre ktoré je v súčastnosti najviac využívaný, ale pre prenos všetkých typov dát s potrebou prenosu v reálnom čase. Konkrétny typ použitia špecifikujú RTP profily, ktoré musia byť špecifikované v samostatných dokumentoch. V týchto dokumentoch musia byť uvedené prípadné modifikácie hlavičky, dodefinované ďalšie typy RTP a RTCP správ a mapovanie typov obsahu z hlavičky RTP na konkrétne typy obsahu (napr. kódovanie audia a videa).

Medzi takéto profily patrí napríklad Zabezpečený RTP (SRTP) [1] alebo Profil pre Audio a Video Konferencie s Minimálnou Kontrolou [5], ktorý je dnes suverénne najpoužívanejším RTP profilom a podrobnejšie sa mu budeme venovať v kapitole 3.

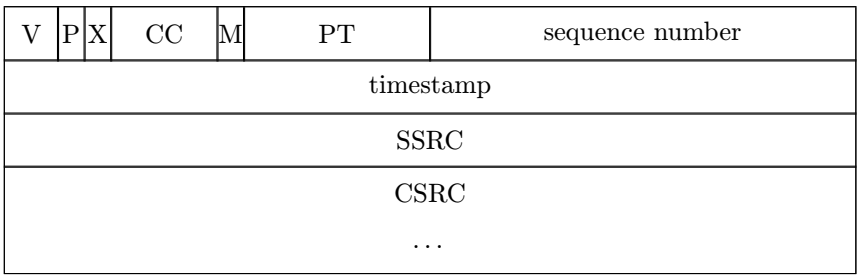
2.1 Protokol RTP

Protokol RTP slúži k samotnému prenosu dát medzi jednotlivými uzlami po sieti.

2.1.1 Formát paketu RTP

Paket protokolu RTP sa skladá z povinnej hlavičky, voliteľného rozšírenia hlavičky a samotných prenášaných dát.

Formát hlavičky, ktorá je v pakete umiestnená bezprostredne na začiatku, je ukázaný na obrázku 2.1.



Obr. 2.1: Hlavička RTP paketu

Popis jednotlivých položiek hlavičky:

V (Version) – Verzia protokolu. Pri aktuálne využívanej verzii je to vždy 2.

P (Padding) – Ak je tento bit nastavený, dáta nesené paketom obsahuje jeden alebo viacero bajtov, ktoré nie sú užitočnou časťou dát. V tomto prípade posledný bajt nesie počet bajtov na zahodenie.

X (eXtension) – Ak je nastavený, za fixnou časťou hlavičky bude nasledovať jedno rozšírenie hlavičky.

CC (CSRC Count) – Počet prispievajúcich zdrojov uvedených v hlavičke.

M (Marker) – Interpretácia tohto bitu je ponechaná na jednotlivých profiloch.

PT (Payload Type) – Identifikuje formát obsahu neseného RTP paketom a jeho následnú interpretáciu aplikáciou. Definícia jednotlivých formátov obsahu je ponechaná na jednotlivé profily. Typ môže byť mapovaný na číslo staticky alebo aj dynamicky pomocou iných metód ako RTP (napr. pomocou SDP).

sequence number – Číslo paketu v streame. Čísla jednotlivých paketov sa postupne zvyšujú o jednotku. Počiatočné číslo sekvencie by malo byť volené náhodne.

timestamp – Určuje sa podľa vzorkovania signálu prenášaného pomocou streamu. Musí byť odvodené od hodín, ktoré sa inkrementujú monotónne a lineárne. Dva za sebou vyslané pakety však nemusia mať monotónny priebeh časovej pečiatky. Táto situácia môže nastať, ak dáta vznikli v inom poradí, ako boli vysielané. Príkladom môže byť video s prekladaným snímkovaním.

SSRC identifier – Identifikačné číslo synchronizujúceho zdroja. Táto 32-bitová hodnota však vôbec nesúvisí s IP adresou zariadenia. Je generovaná spôsobom popísaným v [6] a musí byť v sedení unikátna. V prípade kolízie identifikátorov v sedení musí jeden zo zdrojov pre seba vygenerovať nové identifikačné číslo.

CSRC identifiers – Identifikačné čísla jednotlivých prispievajúcich zdrojov. Pre hodnoty identifikačného čísla platia rovnaké pravidlá, ako pre synchronizačný zdroj.

V prípade použitia rozšírenia hlavičky, musí toto nasledovať bezprostredne za povinnou časťou. Táto môže obsahovať doplnkové informácie špecifické pre jednotlivé profily. Formát hlavičky nie je jednotný, jedinou spoločnou časťou je druhých 16-bitov rozšírenia, určujúcich dĺžku rozšírenia hlavičky v 32-bitových slovách.

Za prípadným rozšírením hlavičky nasledujú samotné dáta. Ich súčasťou môžu byť aj informácie potrebné pre spracovanie konkrétneho typu dát aplikáciou. Aj keď tieto informácie môžu byť obsiahnuté v rozšírení hlavičky, odporúča sa, aby boli obsiahnuté až v tejto sekcii.

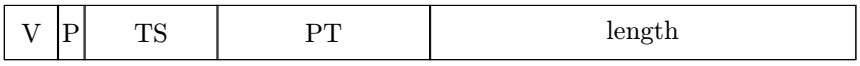
2.2 Protokol RTCP

Protokol RTCP slúži na obsluhu a podporu služieb poskytovaných protokolom RTP. Medzi jeho hlavné funkcie patrí kontrola kvality služby, pre poskytnutie prostriedkov, pre úpravu rýchlosti odosielania dát zo zdroja a poskytovanie informácií o streame pre príjemcov. RTCP definuje 5 typov správ: SR (Sender Report), RR (Reciever Report), SDES (Source DEScription), BYE a APP, ale norma povoľuje dodefinovanie nových typov profilom v prípade potreby.

RTCP pakety sa prenášajú vo forme zlúčených paketov. Jeden zlúčený paket sa vytvorí spojením viacerých paketov za sebou a až táto výsledná skupina paketov sa zabalí do protokolu nižšej vrstvy. Na začiatku zlúčeného paketu sa musí vždy nachádzať paket typu Sender Report alebo Reciever Report. V prípade, že nie je potrebné preniesť žiadnu

z informácií nesených jedným z týchto paketov, na začiatok paketu sa priloží prázdny paket Sender Report. Za týmto úvodným paketom môžu v prípade potreby nasledovať ďalšie pakety týchto typov. Po týchto musí vždy nasledovať paket typu Source Description, ktorý musí obsahovať minimálne položku CNAME a v prípade potreby môže byť nasledovaný ďalšími paketmi tohto typu. Ako posledné prídu všetky ostatné voliteľné typy paketov, vrátane typov dodefinovaných profilom, ale v prípade potreby paketov typu BYE, musia byť tieto v rámci zloženého paketu umiestnené ako posledné. Všetky pakety nasledujú bezprostredne za sebou. V prípade použitia šifrovania sa pred začiatok paketu priloží ešte náhodné 32-bitové číslo. Ak je potrebné pre šifrovanie, prípadne pre potreby protokolov nižších vrstiev zarovnať dáta priložením neužitočných dát, položka padding sa nastaví len v poslednom pakete a nadbytočné dáta sa priložia až za tento paket. Položka padding teda nesmie byť v ostatných paketoch nastavená.

Formát jednotlivých paketov je závislý na jeho type. Podrobný výpis a vysvetlenie formátov jednotlivých paketov je nad rámec tohto dokumentu a je dostupný v [6]. Z tohto dôvodu bude u jednotlivých paketov uvedený hrubý náčrt štruktúry a niekoľko pre detekciu potenciálne podstatných položiek. Všetky pakety majú však spoločných prvých 32-bitov, ktoré sú vyobrazené na obrázku 2.2. Význam jednotlivých položiek je nasledujúci:



Obr. 2.2: Prvých 32-bitov RTCP paketu

V (Version) – Rovnako ako pri hlavičke RTP táto položka určuje verziu použitého protokolu.

P (Padding) – Táto položka je tiež zhodná s protokolom RTP. Určuje, či sa na konci paketu vyskytujú bajty slúžiace pre zarovnanie. Podrobný popis je možné nájsť pri popise hlavičky RTP.

TS (Type Specific) – Význam tejto položky je rozdielny pre rôzne typy paketov a preto bude uvedený pri popise jednotlivých typov.

PT (Packet Type) – Položka určuje, o ktorý z 5 typov RTCP paketu sa jedná.

length – Určuje dĺžku paketu v počte 32-bitových slov mínus 1, vrátane prvých 32-bitov.

2.3 Typy RTCP paketov

2.3.1 SR: Sender Report

Hlásenia zdrojov. Pakety tohto typu slúžia na prenos štatistík o streame od účastníkov, ktorí zasielajú dáta do sedenia.

V položkách spoločnej časti má `SR` paket v položke `Packet Type` hodnotu 200 a v položke `Type Specific`, je uvedený počet hlásení obsiahnutých v pakete. Za spoločnou časťou nasleduje identifikátor SSRC. Za ňou nasleduje blok informácií od odosielateľa RTCP paketu, nasledovaný niekoľkými blokmi obsahujúcimi samotné hlásenia zdrojov, ktorých počet bol uvedený v spoločnej časti paketu. Ako posledné nasleduje rozšírenie špecifické pre jednotlivé profily.

2.3.2 RR: Reciever Report

Hlásenia príjemcov. Používajú sa pre prenos štatistík od účastníkov, ktorí nezasielajú dáta do streamu. Používa sa prípadne aj pre účastníkov zasielajúcich dáta, ak ich počet presiahne 31.

Formát tohto typu paketu je veľmi podobný formátu typu `SR`. Jedinými rozdielmi sú: v položke `Packet Type` je hodnota 201 a blok informácií od odosielateľa je vynechaný. Zvyšné položky aj ich poradie je zachované.

2.3.3 SDES: Source DEScription

Popis zdroja. Obsahuje jednotlivé položky na popis zdroja. Pakety tohto typu sú informačne najhodnotnejšie, čo sa týka informácií o samotnom streame, preto bude popísaný podrobnejšie.

V spoločnej časti paketu sa v položke `Packet Type` nachádza hodnota 202 a v položke `Type Specific` počet zhlukov informácií. Za spoločnou časťou nasledujú jednotlivé zhluky informácií, ktorých počet bol uvedený.

Jeden zhluk obsahuje v prvých 32 bitoch identifikátor SSRC alebo CSRC, za ktorým nasledujú jednotlivé informačné položky, ktorých formát je veľmi jednoduchý. Prvých 8 bitov tvorí typ položky, nasledovaný 8-bitovou hodnotou dĺžky reťazca v bajtoch, za ktorou nasleduje text v kódovaní UTF-8, s maximálnou dĺžkou 255 bajtov. Začiatok každej položky však musí byť zarovnaný na 32-bitov.

Paket môže obsahovať nasledujúce typy informačných položiek (v zátvorke bude uvedená hodnota v položke `type` na začiatku každej informačnej položky):

CNAME (1) – Zástupné meno zdroja. Identifikuje jednoznačne zdroj streamu, pričom musí byť pre stream jedinečné. Doporučené je plné doménové meno odosielajúceho zariadenia prípadne doplnené o užívateľa, vo formáte `user@host`. Keďže pri kolízii identifikátorov jednotlivých zdrojov dôjde k zmene identifikátora, zástupné meno sa nesmie meniť po celú dobu sedenia.

NAME (2) – Názov reálne opisujúci stream. Môže byť v ľubovoľnom formáte podľa volieb používateľa.

EMAIL (3) – Emailová adresa v štandardnom formáte.

PHONE (4) – Telefónne číslo. Je odporúčaný medzinárodný formát začínajúci znakom „+“.

LOC (5) – Geografická lokácia odosielateľa streamu. V rôznych podmienkach je možné očakávať rôzne stupne detailnosti informácie (napr. číslo izby a poschodie alebo mesto a štát).

TOOL (6) – Názov a prípadne verzia nástroja generujúceho stream.

NOTE (7) – Položka, ktorej obsah môže definovať profil. Nemala by byť obsahom každého paketu RTCP z dôvodu narastajúceho množstva odosielaných dát, ale mala by byť používaná pri výskyte jednorazových udalostí.

PRIV (8) – Položka používaná pre aplikačne špecifické rozšírenia SDES paketov. Jej formát je mierne odlišný: Za 8-bitovou hodnotou dĺžky nasleduje 8-bitová dĺžka prefixu v zhodnom formáte, za ňou nasleduje samotný prefix, nasledovaný hodnotou. Prefix slúži na označenie určené osobou definujúcou obsah položky.

2.3.4 BYE

Paket zasielaný pri opustení streamu niektorým z účastníkov.

Paket v spoločnej časti v položke `Packet Type` obsahuje hodnotu 203 a v položke `Type Specific` počet identifikátorov SSRC alebo CSRC opúšťajúcich sedenie. Za ním nasledujú samotné identifikátory. Poslednou časťou paketu je voliteľný dôvod odchodu zo sedenia, pred ktorým ešte nasleduje 8-bitová hodnota dĺžky.

2.3.5 APP

Paket prenášajúci informácie špecifické pre aplikáciu. Slúžia pre možnosť implementácie vlastností špecifických pre konkrétne aplikácie.

Jeho hodnota v položke `Packet Type` je 204. V položke `Type Specific` obsahuje podtyp paketu umožňujúci zhlukovanie viacerých typov paketov pod jedno meno. Za povinnou časťou nasleduje identifikátor SSRC alebo CSRC, za ktorým nasleduje meno zvolené autorom aplikácie (32-bitov) a ako posledné sú obsiahnuté samotné aplikačné dáta.

2.4 Použitie

Dnes sa RTP a RTCP používa primárne na prenos audia a videa cez siete postavené na modely TCP/IP a to pomocou profilu RTP/AVP. V oblasti VoIP je to primárne využívaný prostriedok na prenos samotného hlasu cez sieť. Tak isto je využívaný pri potrebe real-time streamovania videa najmä v oblasti videokonferencií, kde momentálne najviac využívaný štandard H.323 využíva RTP na prenos audio a video dát.

Existuje však aj mnoho aplikácií prenášajúcich audio a video cez sieť inými protokolmi. Medzi takéto patrí napríklad program Skype, ktorý využíva vlastný, uzavretý protokol. Je to hlavne z toho dôvodu, že Skype nemusí dodržiavať kompatibilitu medzi viacerými aplikáciami a vývojári mohli použiť protokol „ušitý na mieru“ ich potrebám.

Ďalším príkladom môžu byť v dnešnej dobe veľmi populárne a využívané video-servery typu `www.youtube.com`. Tie väčšinou používajú na prenos protokol HTTP. Tento bol zvolený z dvoch hlavných dôvodov. Prvým z nich je, že dáta nevznikajú v reálnom čase a môžu byť čo najrýchlejšie a v čo najväčšej kvalite prenesené k príjemcovi, kde sú po dobu prehrávania uložené v cache pamäti. Druhým z nich je potreba dostupnosti služby čo najväčšej skupine ľudí. Ak sa užívatelia dostanú k stránke, znamená to, že bezpečnostná politika siete povoľuje komunikáciu protokolom HTTP, a teda aplikácia nebude mať problém preniesť video.

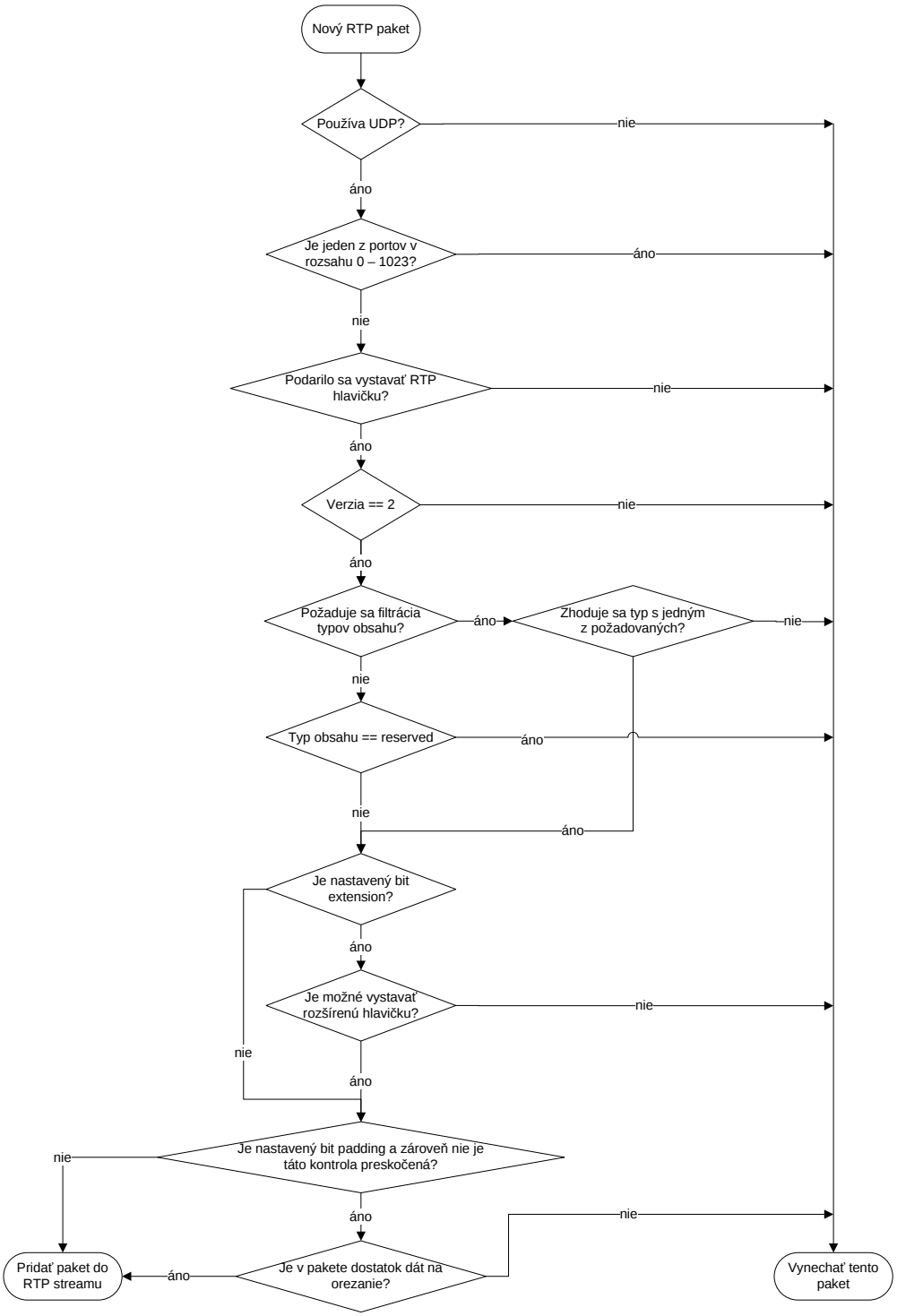
2.5 Detekcia RTP

Vývoju algoritmu pre detekciu RTP a RTCP prenosov v sieťovej komunikácii a jeho následnej implementácii v jazyku Python sa venuje [3], a preto bude tento len stručne popísaný.

Samotný algoritmus je založený na dvojfázovom filtri. V prvej fáze, prebieha takzvaná „per-packet“ filtrácia. V tomto kroku sa na základe viacerých parametrov, vyfiltrujú pakety, ktoré nespĺňajú podmienky potrebné pre RTP paket. Kompletná schéma algoritmu je zobrazená na obrázku 2.3. V prípade, že paket prejde filtráciou, je tento zaradený do toku (streamu) na základe jeho zdrojovej a cieľovej IP adresy, zdrojového a cieľového portu, identifikátoru synchronizačného zdroja a hodnoty položky `payload type`. Druhá fáza zabezpečuje jednoduchý „per-flow“ filter. Jeho úlohou je po skončení detekcie, alebo prípadne

v určitých časových intervaloch, kontrolovať počet paketov v jednotlivých tokoch. V prípade, že tok neobsahuje dostatočné množstvo paketov, je tento odstránený. V prípade kontroly v určitých časových intervaloch však treba zabezpečiť, aby bola kontrola vynechaná u príliš nových tokov, u ktorých je možnosť, že sa ešte nazbieral dostatok paketov. Táto druhá fáza filtru zabezpečí veľmi dobrú odolnosť proti falošne detekovaným streamom.

Podobný algoritmus bol publikovaný aj v článku [2], ktorý sa zaoberá porovnávaním prevádzky programu Skype a iných VoIP programov používajúcich protokol RTP. K získaniu potrebných dát k výskumu vytvorili detekčný program na zachytenie RTP komunikácie podobný vyššie uvedenému algoritmu. Tento algoritmus však pozostával len z „per-packet“ filtra, ktorý na rozdiel od nášho algoritmu používal menšiu množinu filtrovacích parametrov. V tomto prístupe nemusí byť medzi obomi algoritmami podstatný rozdiel, ale vďaka absencii „per-flow“ filtra masívne narastie množstvo falošne detekovaných streamov, ako už ukázali testy v [3]. Táto skutočnosť by nemusela byť veľký problém pri neskoršom spracovávaní výsledkov, kde sa dajú takéto streamy jednoducho dodatočne odstrániť, avšak v prípade použitia vo firewalle by mohla táto skutočnosť spôsobovať veľké množstvo zbytočne otváraných portov.



Obr. 2.3: Algoritmus pre detekciu jednotlivých RTP paketov

Kapitola 3

Multimediálne kodeky používané v RTP

3.1 RTP profil pre audio a video konferencie s minimálnou kontrolou

Tento profil, taktiež často označovaný ako RTP Audio Video Profil (RTP/AVP), bol prvý krát definovaný v dokumente RFC 1890, ktorý bol nahradený dokumentom RFC 3551 [5]. Profil sa v súčastnosti používa na prenos ľubovoľného audio a video obsahu, či sa už jedná o použitie v oblasti VoIP, videokonferencií, alebo streamingu videa po počítačovej sieti.

Profil žiadnym spôsobom neovplyvňuje stavbu RTP hlavičky a samotný nepoužíva jej rozšírenie. Aplikácie však nesmú ignorovať bit `extension`, no musia byť prípadne pripravené rozšírenie hlavičky ignorovať.

Tento profil podporuje viacero druhov kódovaní samostatného audia, videa, alebo kombinácie oboch. Ich mapovanie na položku `payload type (PT)` v hlavičke RTP paketu môže byť buď statické alebo dynamické. Statické čísla `PT` pre audio kodeky sú uvedené v tabuľke 3.1 a pre video a kombinované kodeky v tabuľke 3.2. Každému kodeku je pridelené aj krátke meno (encoding name), ktoré môže byť použité v signalizačných protokoloch (napr. SDP). Ďalšie kodeky môžu byť staticky mapované po registrácii v organizácii IANA (Internet Assigned Numbers Authority), z dôvodu zachovania unikátneho mapovania pre jednotlivé kodeky. Pri použití dynamického mapovania kodeku sa použije jedna z hodnôt v rozsahu 96–127 a o mapovanie konkrétneho kodeku sa postará jeden z použitých signalizačných protokolov.

3.2 Prieskum mapovania kodeku na číslo položky `payload type`

Pre potreby tejto práce a overenia súčasnej reálnej situácie bol vypracovaný výskum, ktorý skúmal podporu kodekov v niekoľkých aplikáciách používajúcich prenos cez RTP. Ako vzorka boli vybraté softvérové telefóny Ekiga, SJphone a X-Lite, ktoré sú dostupné zadarmo, hardvérového telefónu Well SIP-T20, domáceho smerovača s VoIP bránou Linksys WRP400-G2 a videokonferenčného softvéru Polycom PVX.

Výsledné zistenia týkajúce sa softvéru Ekiga sú zaznamenané v tabuľke 3.3. Je v nich možné pozorovať, že v rámci tohto softvéru sú hodnoty `payload type`, ktoré sú prideľované

dynamicky, rozdielne aj medzi verziou pre operačný systém Windows a Linux. Dodržuje však všetky hodnoty statického mapovania v oboch týchto verziách.

Z výsledkov získaných z programu X-Lite zaznamenaných v tabuľke 3.4 je vidieť, že napriek väčšiemu množstvu kodekov zhodných s predchádzajúcim programom, nie je zhodná ani jedna hodnota pridelená kodekom používajúcim dynamické mapovanie. Avšak aj tento program dodržuje všetky hodnoty používané pri statickom mapovaní.

Výsledky získané z programu SJphone sú zaznamenané v tabuľke 3.5. V tomto prípade sa jedna hodnota `payload type` pre kodek iLBC zhodovala s hodnotou, ktorú používal softvér X-Lite. Keďže SJphone používa aj iné čísla pre dynamicky mapované typy obsahov, ktoré sa opäť nezhodujú so žiadnym z predchádzajúcich nástrojov, môžeme predpokladať, že toto je spôsobené náhodou, a teda nie je pravidlom. Všetky hodnoty použité pre kodeky používajúce statické mapovanie ostali opäť zachované.

Hardvérový telefón Well SIP-T20, používal iba kodeky so statickým mapovaním. Ich číslovanie ostalo zachované. Výsledky sú zaznamenané v tabuľke 3.6.

Domáci smerovač s integrovanou VoIP bránou Linksys WRP400-G2, bol v tomto teste zvláštnosťou. V menu pre konfiguráciu umožňoval zmenu čísel používaných pre kodeky s dynamickým mapovaním. V tabuľke 3.7 sú preto uvedené hodnoty zistené zo signalizačného protokolu SIP pri východiskových nastaveniach zariadenia a kodeky, ktoré používali statické mapovanie. V prípade týchto sa číslovanie v menu meniť nedalo a ostalo zachované v porovnaní s ostatnými nástrojmi. Toto zariadenie taktiež podporovalo ako jediné ďalší protokol pre doručovanie udalostí a to Cisco NSE a zapuzdrené RTP.

Videokonferenčný softvér Polycom PVX rovnako ako predchádzajúce nástroje dodržiava hodnoty pre statické mapovanie, avšak opäť nenastala žiadna zhoda, týkajúca sa čísel používaných pre dynamické mapovanie kodekov. Výsledky sú zaznamenané v tabuľke 3.8.

Podľa výsledkov je zjavné, že všetky aplikácie striktne dodržujú statické mapovanie. U dynamického mapovania sa nemožno spoliehať na žiadne ustálené hodnoty. Tieto sa značne líšili nielen medzi aplikáciami samotnými, ale napríklad aj medzi vydaniami pre operačný systém Linux a Windows s rovnakým číslom verzie (Ekiga). Jedinou výnimkou bol typ obsahu označovaný ako `telephone-event`, slúžiaci na prenos tónovej voľby využívanej analógovými telefónmi, podľa normy DTMF. Tento typ bol vo všetkých programoch mapovaný na číslo 101.

Pri samotnej detekcii môžu byť tieto zistenia užitočné, pretože môžeme prehlásiť, že statické mapovanie sa v praxi skutočne dodržuje, ale u dynamického mapovania sa nemôžeme spoľahnúť na žiadne konkrétne kodeky pri niektorých hodnotách. Tieto zistenia sa použijú pre mapovanie hodnôt pre určenie kodeku u detekovaných streamov.

PT	encoding name	clock rate (Hz)
0	PCMA	8 000
1	reserved	
2	reserved	
3	GSM	8 000
4	G723	8 000
5	DVI4	8 000
6	DVI4	16 000
7	LPC	8 000
8	PCMA	8 000
9	G722	8 000
10	L16	44 100
11	L16	44 100
12	QCELP	8 000
13	CN	8 000
14	MPA	90 000
15	G728	8 000
16	DVI4	11 025
17	DVI4	22 050
18	G729	8 000
19	reserved	
20	unassigned	
21	unassigned	
22	unassigned	
23	unassigned	
dyn	G726-40	8 000
dyn	G726-32	8 000
dyn	G726-24	8 000
dyn	G726-16	8 000
dyn	G729D	8 000
dyn	G729E	8 000
dyn	GSM-EFR	8 000
dyn	L8	var.
dyn	RED	
dyn	VDVI	var.

Tabuľka 3.1: Zodpovedajúce hodnoty `payload type` pre audio kodeky. [5]

PT	encoding name	media type	clock rate (Hz)
24	unassigned	V	
25	CelB	V	90 000
26	JPEG	V	90 000
27	unassigned	V	
28	nv	V	90 000
29	unassigned	V	
30	unassigned	V	
31	H261	V	90 000
32	MPV	V	90 000
33	MP2T	AV	90 000
34	H263	V	90 000
35–71	unassigned	?	
72–76	reserved	N/A	N/A
77–95	unassigned	?	
96–127	dynamic	?	
dyn	H263-1998	V	90 000

Tabuľka 3.2: Zodpovedajúce hodnoty `payload type` pre video a kombinované kodeky [5]

encoding name	PT (Windows)	PT (Linux)	media type	clock rate (Hz)
Speex	125	113	A	16 000
iLBC	110	116	A	8 000
PCMU	0	0	A	8 000
PCMA	8	8	A	8 000
GSM	3	3	A	8 000
CELT	95	115	A	48 000
CELT	99	114	A	32 000
G722	9	9	A	16 000
Speex	124	112	A	8 000
G726-16	105	122	A	8 000
G726-24	104	121	A	8 000
G726-32	103	120	A	8 000
G726-40	102	119	A	8 000
MS-GSM	106	123	A	8 000
h264	109	N/A	V	90 000
theora	126	99	V	90 000
h261	31	31	V	90 000
h263	34	N/A	V	90 000
h263-1998	108	N/A	V	90 000
MP4V-ES	114	N/A	V	90 000
telephone-event	101	101	-	8 000

Tabuľka 3.3: kodeky podporované programom Ekiga a zodpovedajúce čísla pre `payload type`

encoding name	PT	media type	clock rate (Hz)
BV32	107	A	16 000
BV32-FEC	119	A	16 000
DVI4	5	A	8 000
DVI4	6	A	16 000
PCMA	8	A	8 000
PCMU	0	A	8 000
GSM	3	A	8 000
iLBC	98	A	8 000
L16	102	A	16 000
SPEEX	97	A	8 000
SPEEX-FEC	105	A	8 000
SPEEX	100	A	16 000
SPEEX-FEC	106	A	16 000
h263	34	V	90 000
h263-1998	115	V	90 000
telephone-event	101	-	8 000

Tabuľka 3.4: kodeky podporované programom X-Lite a zodpovedajúce čísla pre `payload type`

encoding name	PT	media type	clock rate (Hz)
GSM	3	A	8 000
iLBC	97	A	8 000
iLBC	98	A	8 000
speex	110	A	8 000
PCMA	8	A	8 000
PCMU	0	A	8 000
telephone-event	101	-	8 000

Tabuľka 3.5: kodeky podporované programom SJphone a zodpovedajúce čísla pre `payload type`

encoding name	PT	media type	clock rate (Hz)
PCMA	8	A	8 000
PCMU	0	A	8 000
G729	18	A	8 000
G722	9	A	8 000
telephone-event	101	-	8 000

Tabuľka 3.6: kodeky podporované hardvérovým telefónom WELL SIP-T20 a zodpovedajúce čísla pre `payload type`

encoding name	PT	media type	clock rate (Hz)
G726-32	98	A	8 000
G729ab	99	A	8 000
PCMA	8	A	8 000
PCMU	0	A	8 000
telephone-event	113	-	8 000
encaprtp	100	-	8 000
telephone-event	101	-	8 000

Tabuľka 3.7: kodeky podporované routerom Linksys WRP400-G2 a zodpovedajúce čísla pre `payload type`

encoding name	PT	media type	clock rate (Hz)
SIREN14	99	A	16 000
SIREN14	98	A	16 000
SIREN14	97	A	16 000
G7221	102	A	16 000
G7221	101	A	16 000
G7221	103	A	16 000
G722	9	A	8 000
G728	15	A	8 000
G729	18	A	8 000
PCMU	0	A	8 000
PCMA	8	A	8 000
h264	109	V	90 000
h263	34	V	90 000
h263-1998	96	V	90 000
h261	31	V	90 000
telephone-event	101	-	8 000

Tabuľka 3.8: kodeky podporované programom Polycom PVX a zodpovedajúce čísla pre `payload type`

Kapitola 4

Hlasové kodeky

Existuje veľké množstvo kodekov pre prenos hlasu. Vo väčšine prípadov sa používajú kodeky, ktoré sú štandardizované niektorou z organizácií ako napríklad ITU-T alebo IETF. Vďaka tomuto kroku je zabezpečená kompatibilita medzi zariadeniami alebo softvérom od rôznych výrobcov. Používajú sa však aj uzavrené riešenia, ktoré budú fungovať len medzi zariadeniami jedného výrobcu a pri hovore so zariadením iného výrobcu ich opäť nahradí štandardizovaný kodek. U týchto uzavretých riešení tradične výrobca sľubuje lepšiu kvalitu audia pri nižšom zaťažení prenosových liniek.

Používané kodeky sa líšia vo viacerých aspektoch, ako napríklad použitým algoritmom a jeho zložitosťou, veľkosťou potrebného prenosového pásma pre dátový tok, výslednou kvalitou audia alebo rozsahom frekvencií, ktorý je kodek schopný zakódovať. Ako príklad môžeme uviesť viac druhov posledného zmieneného, a teda veľkosti frekvenčného prenosového pásma. Podľa tohto môžeme deliť kodeky na štyri typy uvedené v tabuľke 4.1. V tabuľke je vidieť ako zvýšenie rozsahu prenášaného frekvenčného pásma vyžaduje aj zvýšenie vzorkovacej frekvencie a veľkosti dátového toku. Odmenou za toto zvýšenie však bude aj vyššia kvalita audia, ktoré je schopné preniesť vyššie frekvencie, a teda aj vhodnosť na prenos hudby. Medzi ďalší pre VoIP zaujímavý parameter kodeku môžeme označiť aj algoritmické oneskorenie. Táto hodnota vyjadruje oneskorenie prenášaného zvuku, ktoré spôsobuje samotný kódovací algoritmus (napríklad čakaním na celý rámec alebo doprednú analýzu).

Typ	**Frekvenčný rozsah (Hz)**	**Vzorkovacia frekvencia (kHz)**	**Dátový tok (kbps)**
Narrowband (NB)	300–3400	8	2,4–64
Wideband (WB)	50–7000	16	6,6–96
Super-wideband (SWB)	50–14000	32	24–48
Fullband (FB)	20–20000	48	32–128

Tabuľka 4.1: Typy kodekov podľa frekvenčného prenosového pásma [7]

Nasleduje krátky popis niektorých z kodekov používaných pri prenose hlasu. Ide len o ilustračný prehľad štandardizovaných kodekov, umožňujúci čitateľovi urobiť si prehľad o možnej variabilite kodekov. Časti tejto kapitoly sú prevzaté z [7]. V prípade podrobnejšieho záujmu o problematiku hlasových kodekov odporúčam čitateľovi uvedenú publikáciu, prípadne konkrétnu normu vzťahujúcu sa k danému kodeku.

4.1 Prehľad vybraných hlasových kodekov

4.1.1 G.711

Kodek G.711 s dátovým tokom 64 kbps využívajúci PCM (Pulse Code Modulation) ITU-T prijalo v roku 1972 a ďalej upravilo v roku 1988. Ide o prvý ITU-T štandard pre kódovanie hlasu v sérií "G"kodekov. Jedná sa o „narrowband“ kodek, a teda podporuje frekvencie v rozsahu 300–3400 Hz. Z historických dôvodov boli definované dva varianty, líšiace sa použitými pravidlami pre logaritmický companding[1]. Jeden variant sa označuje μ-law a používa sa v Severnej Amerike a Japonsku. Druhý sa označuje A-law a predstavuje štandard pre Európu a zvyšok sveta. Enkóder G.711 konvertuje lineárne 14 bitové uniformné PCM na kód s 8 bitmi na vzorku, podľa μ-law alebo A-law, s jemným rozložením kvantizačných úrovní pre nižšie hlasitosti reči a redším rozložením pre hlasový signál s vyššími hlasitosťami. Dekóder sa stará o spätný prevod tohto kódu na uniformné PCM. Kodek pracuje s dátovým tokom 64 kbps a je založený na kódovaní vzoriek. Toto znamená, že algoritmické oneskorenie je len dĺžka jednej vzorky, čo pri vzorkovacej frekvencii G.711 8000 Hz predstavuje oneskorenie 0,125 ms.

Pri použití G.711 vo VoIP aplikáciach sa štandardne paket vytvorí a odošle raz za 20 ms (dĺžka paketizácie). Pôvodný štandard G.711 neobsahoval žiadny algoritmus pre riešenie problému straty paketov. Tento je však kriticky dôležitý pri prenose paketov cez stratové siete, a teda pri použití vo VoIP aplikáciách. Tento problém vyriešilo rozšírenie G.711 Appendix 1 z roku 1999, ktoré štandardizovalo algoritmus s nízkou komplexnosťou a vysokou kvalitou pre dopĺňanie výsledného signálu, na miestach kde došlo k strate paketu. Tento algoritmus je podla štandardu povinnou súčasťou každej VoIP aplikácie. Pred uvedením tohto rozšírenia kodek neobsahoval žiadne riešenie problému straty paketov.

Jedná sa spolu s G.729 a jeho variantami G.729a a G.729b o jeden z najpoužívanejších kodekov vo VoIP.

4.1.2 G.711.1

G.711.1 je rozšírením kodeku G.711 o prenos s vyšším frekvenčným rozsahom. Jedná sa teda o „wideband“ kodek. ITU-T tento kodek definovalo v roku 2008. Kodek podporuje jak „wideband“, tak „narrowband“ kódovanie hlasu. V prípade použitia „wideband“ kódovania môže zakódovaný dátový tok prenášať zvukový signál vo frekvenčnom rozsahu 50–7000 Hz. Do enkodéru v tomto prípade vstupuje signál so vzorkovacou frekvenciou 16 kHz. Tento je rozdelený na dva 8 kHz signály. Jeden obsahuje časť pôvodného signálu s nižšími frekvenciami a druhý obsahuje časť signálu s vyššími frekvenciami. Na signál obsahujúci nižšie frekvencie sa použije kódovanie kompatibilné s kodekom G.711. Na signál s vyššími frekvenciami sa použije modifikovaná diskrétna kosínusová transformácia (MDCT), ktorá je aplikovaná na rámce obsahujúce 5 ms dlhé časti signálu. Výsledný dátový tok sa delí do troch vrstiev:

Layer 0 – Základný tok obsahujúci obsahujúci nižšie frekvencie s dátovým tokom 64 kbps používajúci PCM. Tento teda obsahuje 320 bitov na každých 5 ms signálu.

Layer 1 – Rozširujúci tok obsahujúci nižšie frekvencie. Jeho dátový tok je 16 kbps, čo predstavuje 80 bitov na každých 5 ms signálu.

[1] Jedná sa o techniku zmenšovania dynamického rozsahu. Názov techniky vznikol spojením slov compression a expanding

Layer 2 – Rozširujúci tok obsahujúci vyššie frekvencie. Jeho dátový tok je 16 kbps, čo predstavuje 80 bitov na každých 5 ms signálu.

Celkový dátový tok pri použití G.711.1 závisí na tom, z koľkých vrstiev sa skladá výsledný signál a môže nadobudnúť veľkosť 64, 80 alebo 96 kbps. Algoritmické oneskorenie kodeku je 11,875 ms (5 ms rámec, 5 ms predikcia a 1,875 ms QMF (Quadrate-Mirror Filterbank) analýza/syntéza).

4.1.3 G.726

G.726 definovalo ITU-T v roku 1990. Kodek je založený na adaptívnom diferenčnom PCM (ADPCM). Jedná sa o „narrowband“ kodek fungujúci s veľkosťou dátového toku 40, 32, 24 alebo 16 kbps. G.726 zahŕňa predchádzajúce ITU-T štandardy založené na ADPCM a to G.721 pre dátový tok 32 kbps a G.723 pre dátové toky 24 kbps a 40 kbps. Kodek používa vzorky veľkosti 5, 4, 3 alebo 2 bity pri veľkostiach dátového toku 40, 32, 24 alebo 16 kbps pri vzorkovacej frekvencii 8000 Hz. Kodek G.726 bol pôvodne navrhnutý pre použitie v DCME[2] zariadeniach. V súčastnosti sa taktiež používa vo VoIP aplikáciách.

4.1.4 G.728

Kodek štandardizovalo ITU-T v roku 1992 ako „narrowband“ kodek s veľkosťou dátového toku 16 kbps. Kodek je založený na algoritme LD-CELP (Low Delay Code Excited Linear Prediction). Jedná sa o prvý štandardizovaný kodek, ktorý je založený na variante algoritmu CELP, ktorý používa prístup „analýzy syntézou“, pre hľadanie v kódovej knihe. V meraní kvality dosahuje MOS skóre blízko hodnoty 4 pri dátovom toku 16 kbps, čo je podobná kvalita ako kvalita dosahovaná pri 32 kbps ADPCM alebo 64 kbps PCM. Po zavedení G.728 sa navrhlo veľké množstvo štandardov založených na rôznych variantoch algoritmu CELP.

Pre zaistenie nízkeho oneskorenia, G.728 používa pri kódovaní len malé bloky. Tieto obsahujú 5 po sebe nasledujúcich vzoriek. Pri použitej vzorkovacej frekvencii 8000 Hz predstavuje 5 vzoriek algoritmické oneskorenie len 0,625 ms (5x0,125 ms). Kódová kniha má veľkosť 1024 vektorov, a teda vyžaduje 10 bitov pre index kódovej knihy. Príjemcovi je zasielaných len týchto 10 bitov pre každý rámec. To predstavuje dátový tok 16 kbps.

G.728 môže mať ďalej zmenšený dátový tok na 12,8 alebo 9,6 kbps, ktoré sú definované v dodatku annex H.

4.1.5 G.729

Kodek G.729 je ďalším štandardom definovaným ITU-T. Definovaný bol v roku 1996 a je založený na algoritme CS-ACELP (Conjugate Structure Algebraic Code Excited Linear Prediction). Používa dátový tok 8 kbps. Rámce pre kódovanie majú dĺžku 10 ms a používa sa 5 ms predikcia. Celkové algoritmické oneskorenie teda predstavuje 15 ms. Každý 10 ms rámec sa skladá z dvoch 5 ms podrámcov. Koeficienty LPC filtra sú určované na základe celého rámca, zatiaľ čo excitačné parametre signálu sú určované na základe jednotlivých podrámcov. Koeficienty LPC filtra sú ďalej transformované na LPS (Line Spectrum Pairs) pre zaistenie stability a efektivity prenosu. G.729 enkodér, každý 10 ms rámec analyzuje pre získanie spomínaných parametrov, ktoré sú zakódované do 80 bitov. Keďže vzniká 80 bitov

[2] Digital Circuit Multiplication Equipment – Zariadenie na kompresiu hlasu pred prenosom na veľké vzdialenosti.

na každý 10 ms rámec, získame dátový tok 8 kbps. G.729 podporuje tri typy odosielaných rámcov:

Normal frame – Normálne 80 bitové rámce prenášajúce hlas.

Silence Insertion Description (SID) frame – Rámce prenášajúce parametre vlastností generovaného ticha. Používajú sa len v prípade, že je aktivovaná funkcia detekcie ticha.

Null frame – Nulový rámec. Prázdne rámce s dĺžkou 0.

G.729 bola definovaná pre mobilné a sieťové aplikácie. Má vstavané mechanizmy pre nahradzovanie stratených paketov interpoláciou založenou na predchádzajúcich prijatých rámcoch. V rámci kódeku G.729 bolo vydaných viacero rozšírení. Najpoužívanejšie vo VoIP sú pôvodný variant G.729, ktorá má všetky vyššie zmienené vlastnosti a deaktivovanú detekciu ticha. Ďalším vo VoIP používaným variantom je G.729 annex A (G.729a), ktorý má naviac zníženú komplexnosť kódovania pri zachovaní rovnakého dátového toku za cenu zníženia kvality hovoru. Posledným vo VoIP veľmi používaným variantom je G.729 annex B (G.729b), čo je variant G.729a rozšírený o aktiváciu funkcie detekcie ticha.

Tento kodek využíva patentované algoritmy a na jeho používanie je treba zakúpenie licencie. Jedná sa spolu s G.711 o jeden z najpoužívanejších kodekov vo VoIP.

4.1.6 G.723.1

ITU-T štandardizovalo kodek G.723.1 v roku 1996. Kodek ma dva varianty. Prvý variant má veľkosť dátového toku 5,3 kbps a je založený na algoritme ACELP (Algebraic Code Excited Linear Prediction). Druhý variant využíva dátový tok o veľkosti 6,3 kbps a využíva algoritmus MP-MLQ (Multi Pulse Maximum Likelihood Quantisation). Bol navrhnutý pre multimediálne aplikácie ako napríklad vizuálnu telefóniu s nízkym dátovým tokom. Dva varianty dátového toku sa poskytujú pre zvýšenú flexibilitu. Každý 30 ms rámec je rozdelený do štyroch podrámcov, z ktorých každý má 7,5 ms. Predikcia u kodeku G.723.1 má dĺžku jedného podrámca. Kombináciou týchto vlastností získame algoritmické oneskorenie veľké 37,5 ms.

4.1.7 GSM

GSM je kodek štandardizovaný organizáciou ETSI pre druhú generáciu mobilných sietí. GSM 06.10, vydaný v roku 1991, definuje takzvaný „full rate“ (FR) GSM, ktorý používa dátový tok veľkosti 13 kbps a je založený na RPE/LTP (Regular Pulse Excitation / Long Term Prediction) lineárnom predikčnom kóderi. Dĺžka využívaného rámca je 20 ms, čo predstavuje 160 vzoriek na rámec pri používanej vzorkovacej frekvencii 8000 Hz. Každý zakódovaný rámec je dlhý 260 bitov. Každý hlasový rámec je rozdelený na štyri podrámce, každý dlhý 5 ms. LP analýza je prevedená pre každý rámec. RPE analýza je založená na každom podrámci, kde LTP je založená na celých rámcoch. Zakódovaný rámec obsahuje parametre zo všetkých troch analýz.

GSM 06.20, definovaný v roku 1999, štandardizuje takzvanú „half rate“ (HR) GSM. Táto je založená na algoritme VSELP (Vector Sum Excited Linear Prediction), ktorá využíva dátový tok o veľkosti 5,6 kbps. Používa rámce dlhé 20 ms, rozdelené na štyri podrámce. Zkódovaný rámec má veľkosť 112 bitov a obsahuje parametre pre LPC filter a index do VSELP kódovej knihy.

GSM 06.60 bol definovaný v roku 2000 a štandardizuje rozšírenú FR GSM (EFR GSM). Je založená na algoritme ACELP a používa dátový tok o veľkosti 12,2 kbps. Rovnako ako najvyšší variant AMR.

4.1.8 AMR

AMR (Adaptive Multi Rate) je „narrowband“ kodek, založený na algoritme ACELP (Algebraic Code Excited Linear Prediction) a bol definovaný organizáciou ETSI v roku 2000. Bol taktiež zvolený skupinou 3GPP ako povinný kodek pre siete štandardu UMTS (mobilné siete tretej generácie). AMR podporuje viacero módov s rozdielnymi veľkosťami dátového toku a to 4,75, 5,15, 5,9, 6,7, 7,4, 7,95, 10,2 a 12,2 kbps. Dĺžka rámca je 20 ms. Pre získanie LP koeficientov sa analyzuje každý rámec zvlášť. Pri móde s veľkosťou dátového toku 12,2 kbps sa rámec analyzuje dva krát, pre ostatné módy sa analyzuje len raz. Každý rámec je delený na štyri podrámce po 5 ms. Analýza výšky tónu prebieha na každom podrámci. Veľkosti zakódovaných rámcov sa líšia podľa zvoleného módu a majú nasledovné veľkosti: 95, 103, 118, 134, 148, 159, 204 a 244 bitov.

4.1.9 iLBC

iLBC (Internet Low Bitrate Codec) je open-source hlasový kodek a bol definovany v IETF RFC 3951 v roku 2004. Tento kodek je navrhnutý pre internetové aplikácie a má dobrú odolnosť voči strate paketov. Je založený na blokovo nezávislom variante algoritmu CELP. V prípade straty paketov je teda schopný zabezpečiť lepšiu kvalitu hovoru ako kodeky založené na iných variantoch algoritmu CELP. Dĺžka rámca je 20 ms (15,2 kbps a 304 bitov na zakódovaný rámec), alebo 30 ms (13,33 kbps a 400 bitov na zakódovaný rámec). Každý rámec je rozdelený na štyri alebo šesť podrámcov. Na každom rámci sa urobí LPC analýza, pričom na 30 ms rámcoch sa robí dva krát. Vyhľadávanie v kódovej knihe prebieha na každom podrámci. Obsah zakódovaných rámcov závisí len na konkrétnom kódovanom rámci, a teda nezávisí na žiadnom z predchádzajúcich. Toto umožňuje zlepšiť techniku nahradzovania stratených paketov, a teda zvýšiť kvalitu hovoru pri ich strate.

4.1.10 SILK

SILK je kodek používaný programom Skype a bol navrhnutý IETF v roku 2009. Kodek podporuje štyri operačné módy a to:

Narrowband (NB) – Vzorkovacia frekvencia 8 kHz.

Mediumband (MB) – Vzorkovacia frekvencia 8 alebo 12 kHz.

Wideband (WB) – Vzorkovacia frekvencia 8, 12 alebo 16 kHz.

Super Wideband (SWB) – Vzorkovacia frekvencia 8, 12, 16 alebo 24 kHz.

Základný rámec kodeku ma 20 ms. Veľkosť odosielaného paketu je premenná, môže obsahovať jeden až päť zakódovaných rámcov. Kodek tak isto obsahuje možnosť prerušenia vysielania v prípade detekcie ticha.

4.1.11 G.722

Kodek štandardizovalo ITU-T v roku 1988. Jedná sa o kompresný kódovací štandard pre audio do 7 kHz so vzorkovaciu frekvenciou 16 kHz. Kodek je založený na algoritme SB-ADPCM(Sub Band Adaptive Differential Pulse Code Modulation), ktorý vstupný signál rozdelí na dva podľa frekvenčného pásma, prvý obsahuje nižšie frekvencie v rozsahu 0–4 kHz a druhý vyššie frekvencie 4–8 kHz. Každý z týchto signálov je potom samostatne zakódovaný algoritmom ADPCM. G.722 môže používať dátový tok o veľkosti 64, 56 alebo 48 kbps.

Vďaka vlastnostiam ADPCM môže byť G.722 použitý ako na kódovanie hlasu, tak na kódovanie hudby.

4.1.12 G.722.1

Štandard ITU-T schválený v roku 1999. Kodek je určený na prenos 7 kHz audia s dátovým tokom 24 alebo 32 kbps. Použitá vzorkovacia frekvencia je 16 kHz. Kodek je založený na algoritme MLT (Modulated Lapped Transform). Dĺžka audio rámca na spracovanie je 20 ms s 20 ms predikciou, čo nám určuje algoritmické oneskorenie na 40 ms. Každý rámec je transformovaný na 320 MLT koeficientov a tie sú následne transformované na 480 alebo 640 bitov pre dátový tok 24 alebo 32 kbps. Vďaka tomu, že sa každý rámec spracováva samostatne, je kodek pomerne odolný voči strate paketov.

V novej verzii G.722.1 je okrem pôvodného kodeku definovaný aj doplnok annex C, v ktorom je uvedený ďalší variant kodeku, ktorý zdvojnásobuje frekvenčné prenosové pásmo zo 7 na 14kHz. Súčasne s prenosovým pásmom bolo potrebné zvýšiť aj vzorkovaciu frekvenciu z 16 na 32 kHz a zvýšiť počet vzoriek na rámec z 320 na 640. Annex C podporuje dátové toky o veľkosti 24, 32 alebo 48 kbps. Pri použití tohto variantu kodeku sa často stretneme s pojmom „High Definition Voice“ alebo len „HD“. Tento kodek sa používa vo videokonferenčných jednotkách, alebo telefónoch od firmy Polycom[3].

4.1.13 AMR-WB/G.722.2

Tento kodek bol štandardizovaný ako AMR-WB u organizácie 3GPP a zároveň u ITU-T ako kodek G.722.2. Napriek rôznemu označeniu sa jedná o absolútne totožný kodek. Jedná sa o „wideband“ kodek s prenášaným frekvenčným pásmom do 7 kHz a vzorkovacou frekvenciou 16 kHz. Kodek je schopný produkovať dátový tok s veľkosťou 6,6, 6,85, 12,65, 14,25, 15,85, 18,25, 19,85, 23,05 respektíve 23,85 kbps. Veľkosť rámca je 20 ms audia. AMR-WB poskytuje audio vo vysokej kvalite, a teda je vhodným kodekom nie len na prenos hlasu, ale aj na prenos hudby, či použití v konferenčných jednotkách.

4.1.14 G.719

Kodek schválený ako štandard ITU-T v roku 2008 je jedným z posledných štandardov pre „fullband“ kodek. Podporuje dátové toky s veľkosťami od 32 do 128 kbps a prenáša frekvenčné pásma až do 20kHz. Vznikol ako spoločný projekt firiem Polycom[4] a Ericsson[5] s cieľom vytvoriť kodek vhodný na prenos hlasu, hudby alebo všeobecného audia vo vysokej kvalite. Jeho frekvenčný rozsah 20 Hz–20 kHz pokryje celý frekvenčný rozsah počuteľný ľudským

[3] http://www.polycom.com/
[4] http://www.polycom.com/
[5] http://www.ericsson.com/

uchom. Použitá vzorkovacia frekvencia je 48 kHz. Veľkosť použitého rámca je 20 ms s 20 ms predikciou, a teda celkové algoritmické oneskorenie predstavuje 40 ms. Kodek je založený na transformačnom kódovaní. Pri transformácii používa veľa adaptívnych vlastností, a teda umožňuje prenos širokej škály audia. Kodek preskenuje 20 ms rámec audia a na základe zisteného typu zvolí správny typ transformačného kódovania a príslušnú veľkosť bitového toku.

G.719 môže byť použitý pre high-end videokonferenčné a teleprezenčné aplikácie, kde môže poskytovať HD prenos hlasu s doprovodom HD videa.

4.1.15 Zhrnutie

Pre lepší prehľad jednotlivých kodekov a ich základných vlastností slúži tabuľka. Je zjavné, že pri výbere audio kodeku je veľa rozmanitých možností a každý kodek má svoje výhody aj nevýhody. Treba si však uvedomiť, že nie všetky uvedené kodeky sa vo VoIP reálne používajú. V praxi sa stretávame len s pomerne obmedzenou množinou kodekov, a ani u tých používaných sa nemusíme stretnúť so všetkými jeho variantmi a vlastnosťami. Medzi najpoužívanejšie kodeky v bežnej VoIP prevádzke sa radia G.711 a G.729, kde prvý menovaný je dostupný zadarmo a algoritmicky nenáročný na spracovanie, ale potrebuje vyššie prenosové pásmo. Na druhej strane G.729 vyžaduje platenú licenciu a poskytuje dobrú kvalitu audia pri vyžadovaní veľmi malého dátového toku.

Kodek	**Štandard /rok**	**Typ**	**NB/WB /SWB/FB**	**Dátový tok (kbps)**	**Rámec (ms)**	**Bitov na rámec(vz.)**	**Predik. (ms)**	**Alg. onesk. (ms)**
G.711	ITU/1972	PCM	NB	64	0,125	8	0	0,125
G.726	ITU/1990	ADPCM	NB	40 32 24 16	0,125	5 4 3 2	0	0,125
G.728	ITU/1992	LD-CELP	NB	16	0,625	10	0	0,625
G.729	ITU/1996	CS-ACELP	NB	8	10	80	5	15
G.723.1	ITU/1996	ACELP MP-MLQ	NB NB	5,3 6,3	30	159 189	7,5	37,5
GSM	ETSI/1991 ETSI/1999 ETSI/2000	(FR) RPE-LTP (HR) VSELP (EFR) ACELP	NB NB NB	13 5,6 12,2	20	260 112 244	0 0 0	20 20 20
AMR	ETSI/2000	ACELP	NB	4,75 5,15 5,9 6,7 7,4 7,95 10,2 12,2	20	95 103 118 134 148 159 204 224	5 0	25 20
iLBC	IETF/2004	CELP	NB	15,2 13,33	20 30	304 400	0	20 30
G.711.1	ITU/2008	PCM-WB (MDCT)	NB/WB	64 80 96	5	320 400 480	5	11,875
G.722	ITU/1988	SB-ADPCM	WB	64 56 48	0,125	8 7 6	0	0,125
G.722.1	ITU/1999 ITU/2005	Transform coding	WB SWB	24 32 24/32/48	20	480 640 480–960	20	40
G.719	ITU/2008	Transform coding	FB	32–128	20	640–2560	20	40
AMR-WB (G.722.2)	ETSI/ITU /2003	ACELP	WB	6,6–23,85	20	132–477	0	20
SILK	IETF/2009	CELP	WB	6–40	20	120–800	0	20

Tabuľka 4.2: Zhrnutie prehľadu kodekov [7]

Kapitola 5

Detekcia kodeku

Z predchádzajúcich výsledkov je zjavné, že pri použití staticky mapovaného kodeku nenastane žiadny väčší problém. Keďže všetky aplikácie statické mapovanie dodržovali, môžeme jednoducho určiť použitý kodek podľa čísla v položke `payload type (PT)` v RTP hlavičke. Avšak v prípade použitia niektorého z kodekov, ktoré sa mapujú dynamicky nastáva problém. Číslo v položke `payload type (PT)` sa mapuje na jednotlivé kodeky cez niektorý zo signalizačných protokolov a pretože sa táto práca venuje detekcii bez ich použitia, nemáme možnosť tieto informácie získať. V samotnej RTP a RTCP hlavičke sa v tomto prípade, na rozdiel od signalizačných protokolov, nenachádzajú žiadne užitočné informácie o použitom kodeku. Budeme sa preto musieť zamerať na sledovanie správania samotných dát nesených protokolom RTP a ďalšími charakteristickými vlastnosťami RTP paketov, podľa toho, ktorým kodekom sú nesené dáta. Medzi dve skúmané metódy patrí detekcia klasifikátorom s použitím neurónovej siete a sledovanie opakujúcich sa znakov v nesených dátach, dĺžky paketov a použitej vzorkovacej frekvencie. Na mieste klasifikátora použili klasifikátor založený na neurónovej sieti.

5.1 Detekcia pomocou klasifikátora

Táto možnosť detekcie je založená na použití štandardnej schémy klasifikátora. Možnosť tejto formy detekcie skúmali aj na Ankara University v Tureckej Ankare [8]. Klasifikátor je založený na steganografickej báze. Je založený na myšlienke, že keď sa pomocou štatistických metód môžu hľadať vzory, ktoré sa obyčajne v danom type dát nenachádzajú, prečo by sa nemohla tá istá metóda použiť na hladanie charakteristických vzorov dát? Na mieste charakteristík streamu použili algoritmus FNN (výsledný počet) a najdlhší Laponov exponent.

Tento klasifikátor teda neskúma špecifické, opakujúce sa vzory, ale štatistiku bitového streamu ako celku. Tieto štatistiky by mali byť u jedného streamu nemenné, a teda by sa pri zmene veľkosti bitového toku, ani pri poprehadzovaní poradia nemali výsledky meniť. Táto metóda teda funguje aj na poškodených streamoch s poprehadzovanými paketmi, bez potreby opätovného preskladania. Tým z Ankara University sa rozhodol testovacie dáta rozdeliť na dve časti. Vo všetkých zakódovaných audio streamoch náhodne poprehadzovali dáta, aby zabezpečili nezávislosť výsledkov na poradí a z prvej polovice extrahovali potrebne štatistiky. Tie boli pre zvýšenie presnosti normalizované a výsledky použité pre natrénovanie neurónovej siete. Druhá polovica dát bola takisto poprehadzovaná, prešla štatistickou analýzou a následnou normalizáciou. Pri následnom použití klasifikátoru na normalizované

štatistiky z druhej polovice dát, dostali výsledky uvedené v tabuľke 5.1.

	AMR 4,5 k	G.726 24 k	G.726 32 k	G.729 32 k	G.726 16 k	PCM 128 k	GSM 13,2 k	MELP 2400	R. S.
AMR 4,5 k	1,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000	0,0000
G.726 24 k	0,0000	0,9707	0,0000	0,0000	0,0000	0,0000	0,0049	0,0244	0,0000
G.726 32 k	0,0000	0,0000	0,9956	0,0000	0,0000	0,0029	0,0010	0,0000	0,0005
G.729 32 k	0,0005	0,0000	0,0000	0,9990	0,0000	0,0000	0,0000	0,0005	0,0000
G.726 16 k	0,0005	0,0005	0,0000	0,0005	0,9961	0,0000	0,0005	0,0010	0,0010
PCM 128 k	0,0000	0,0000	0,0024	0,0000	0,0000	0,9976	0,0000	0,0000	0,0000
GSM 13,2 k	0,0010	0,0151	0,0005	0,0000	0,0020	0,0015	0,9653	0,0142	0,0005
MELP 2400	0,0000	0,0250	0,0005	0,0000	0,0059	0,0000	0,0093	0,9639	0,0000
R. S.	0,0000	0,0000	0,0000	0,0020	0,0024	0,0005	0,0005	0,0000	0,9946

Tabuľka 5.1: Výsledky detekcie klasifikátorom vyvinutým na Ankara University (V zvislom smere je skutočný kodek dát, vo vodorovnom pravdepodobnosť pri klasifikácii). [8]

Z výsledkov je jasné, že metóda ktorú vyvinuli na Ankara University má naozaj vysokú presnosť. Jej ďalšou výhodou je, že bude fungovať aj pri poškodených alebo poprehadzovaných častiach streamu bez výrazného zníženia presnosti. Táto metóda má však aj veľké nevýhody. Jednou z nich je potreba neurónovú sieť natrénovať pomocou veľkého množstva testovacích dát so známymi kodekmi, predtým ako ich bude schopná klasifikovať. Ďalším, väčším problémom tejto metódy je pomerne veľká výpočtová náročnosť. Oproti nasledujúcej metóde, ktorá je značne jednoduchšia, bude beh analýzy výrazne dlhší a jej hardvérové urýchľovanie náročnejšie. Toto síce nie je veľký problém pri analýze už odchytených dát, ale môže nastať problém pri potrebe behu aplikácie v reálnom čase pri analýze dát tečúcich cez sieťové rozhranie. Tento problém je taktiež čiastočne riešiteľný najmä obmedzením analýzy kodekov na malé časti odchytených dát.

5.2 Detekcia pomocou charakteristických znakov

Táto metóda je založená na filtrovaní jednotlivých možných kodekov na základe ich charakteristických znakov. Medzi takéto môžu patriť napríklad dĺžka paketu, doba medzi odosielaním jednotlivých paketov, ale aj charakteristické vzory v samotných nesených dátach.

Pri odchytávaní dát je možné okrem dĺžky paketu zistiť aj rozdiely medzi jednotlivými časovými razítkami jednotlivých paketov a vypočítať tak dobu medzi jednotlivými odoslanými paketmi. Na základe tejto informácie a informácie o dĺžke paketov už môžeme od seba niektoré kodeky odlíšiť.

Ďalšou možnosťou sú opakujúce sa vzory dát, napríklad opakujúca sa sekvencia na začiatkoch paketov alebo periodicky sa opakujúca sekvencia počas celého streamu (napríklad. GSM). Tieto sekvencie však bude potrebné identifikovať z odchytených dát a následne porovnať so štandardami, v ktorých sú definované jednotlivé kodeky. Ďalším problémom tejto časti je, že nie je zaručené, že sa u každého kodeku takéto sekvencie budú objavovať.

Kombináciou skúmania oboch týchto znakov by mohla vzniknúť dostatočne presná metóda pre detekciu použitého kodeku, ktorá by mohla byť ešte obohatená o rozpoznávanie podľa statických typov obsahu. Táto metóda síce nemusí dosahovať vysoké presnosti pri veľkej škále kodekov ako predchádzajúca metóda, ale môže byť dostatočne presná pre aktuálne reálne používané kodeky. Napríklad väčšina VoIP operátorov podporuje na svojich ústredniach len kodeky G.711 a G.729 (aj s kompatibilným variantom A).

5.3 Porovnanie metód

V predchádzajúcich kapitolách sme si predstavili dve potenciálne fungujúce možnosti detekcie multimediálneho kodeku v RTP dátach. Každý má svoje výhody a nevýhody. V prípade

detekcie klasifikátorom medzi najväčšie výhody patrí vysoká presnosť a odolnosť voči chybám, keďže je možné dáta detegovať aj na streamoch, ktoré majú poprehadzované poradie bajtov. Medzi jeho najväčšie nevýhody patrí ale potreba veľkej množiny trénovacích dát, oproti druhej metóde väčšia algoritmická náročnosť a potreba väčšej vzorky dát pre detekciu, z dôvodu potreby nazbierania štatistík o streame. Bez dostatočne dlhých dát pre vytvorenie štatistík totiž bude presnosť detekcie týmto algoritmom rapídne klesať.

Druhá metóda má tak isto svoje výhody aj nevýhody. Medzi výhody patrí veľmi malá algoritmická náročnosť. Keďže je metóda implementovateľná ako jednoduchý konečný automat, táto bude jednoducho akcelerovateľná v hardvéri. Ďalej je metódu možné jednoducho paralelizovať spracovaním viacerých paketov naraz, prípadne skenovaním viacerých vlastností naraz. Ďalej je veľmi príjemnou vlastnosťou schopnosť detekcie na základe jednotiek paketov, ktoré predstavujú len niekoľko málo milisekúnd hovoru. Medzi nevýhody v porovnaní s predchádzajúcou metódou patrí nižšia presnosť a potreba preskúmania a implementácie detekovateľných znakov samostatne pre každý požadovaný kodek. Táto metóda si tak isto neporadí ak sú niektoré z charakteristických znakov poškodené.

Keď zoberieme do úvahy všetky výhody aj nevýhody oboch spomenutých metód a zohľadníme cieľové použitie na VoIP, vychádza ako vhodnejšia metóda detekcie pomocou charakteristických znakov. Jej nevýhody sú akceptovateľné, keďže vo VoIP sfére sa používa len pomerne obmedzené množstvo kodekov a naopak jej výhody sú veľmi užitočné. Vyššia jednoduchosť algoritmu je potrebná pre implementáciu na sieťach s veľkými dátovými tokmi, kde by zložitý algoritmus nezvládal pracovať pri dostatočnej rýchlosti. Schopnosť detekcie na základe veľmi krátkej vzorky dát je zase dôležitá pri využití vo firewalloch. V tejto implementácii by použitie detekcie klasifikátorom spôsobilo odrezanie priveľkej časti začatého hovoru a značne by znížilo komfort používateľa. V krízových situáciach by sme toto dokonca mohli označiť za nebezpečné. Ako príklad krízovej situácie môžeme uvažovať napríklad bezpečnostnú pohotovosť (požiar, vlámanie, atď.) v organizácii využívajúcej VoIP telefóniu. V prípade, že by bola detekcia použitá pre otváranie portov na firewalle, k otvoreniu portov by mohlo dôjsť až niekoľko sekúnd po započatí hovoru. Toto by mohlo mať v prípade stresového vypätia volajúceho fatálne následky.

Z vyššie uvedených dôvodov bude praktická realizácia a nasledujúce testy vypracované s použitím detekcie na základe charakteristických znakov.

Kapitola 6

Vzorové dáta a analýza charakteristických znakov

6.1 Vytvorenie vzorky testovacích dát

Pre potreby analýzy dát a následného testovania výslednej aplikácie vzniklo niekoľko vzoriek dát, ktoré sú popísané v tejto kapitole.

Prvá testovacia sada je vytvorená open-source softvérovým telefónom Ekiga[1]. Zameriava sa teda na širšiu vzorku kodekov so slobodnými implementáciami. Hovor prebiehal medzi dvoma fyzickými počítačmi, ktoré boli spolu prepojené cez prepínač. Pri hovore nebola použitá žiadna ústredňa, ale hovor prebiehal priamo medzi počítačmi, adresovanými IP adresami, bez použitia ústredne. Komunikácia bola odchytená programom Wireshark na jednom z dvoch použitých počítačov. Pri každom hovore boli na jednej strane zakázané všetky kodeky, až na zachytávaný. Zoznam zachytených súborov v tejto testovacej vzorke je zaznamenaný v tabuľke 6.1.

Názov súboru	**Veľkost (kB)**	**Počet paketov**	**Kodek (počet streamov)**
amr-wb.pcap	101,5	736	AMR-WB (2)
g722.pcap	175,3	751	G.722 (2)
g726-16.pcap	85,3	625	G.726 16kbps (2)
g726-24.pcap	99,4	598	G.726 24kbps (2)
g726-32.pcap	117,9	604	G.726 32kbps (2)
g726-40.pcap	148,1	651	G.726 40kbps (2)
g7221.pcap	116,9	867	G.722.1 (2)
gsm.pcap	87,9	809	GSM 06.10 (2)
pcma.pcap	195,1	837	G.711 *A*-law (2)
pcmu.pcap	188,8	810	G.711 μ-law (2)
silk8.pcap	89,1	678	Silk 8kHz (2)
silk16.pcap	82,6	642	Silk 16kHz (2)
speex8.pcap	80,4	822	Speex 8kHz (2)
speex16.pcap	131	1038	Speex 16kHz (2)

Tabuľka 6.1: Zoznam testovacích súborov vytvorených pomocou programu Ekiga

Druhá vzorka dát je vytvorená pomocou komunikácie medzi dvomi smerovačmi Cisco

[1] http://ekiga.org/

2811. Oba smerovače používali vlastnú implementáciu VoIP ústredne firmy Cisco a to Call-Manager Express[2]. Oba smerovače mali na ústredni prihlásené po jednom telefóne od firmy Cisco a ústredne boli nastavené na vzájomnú komunikáciu s použitím signalizačného protokolu SIP a použitím jednotlivých odchytávaných kodekov. Smerovače boli medzi sebou prepojené manažovateľným prepínačom, na ktorom bol nastavený monitorovací port, odosielajúci komunikáciu medzi smerovačmi na zachytávajúci počítač. Na tomto bežal program Wireshark zachytávajúci komunikáciu v promiskuitnom móde. Zoznam zachytených súborov v tejto testovacej vzorke je zaznamenaný v tabuľke 6.2.

Názov súboru	**Veľkost (kB)**	**Počet paketov**	**Kodek (počet streamov)**
g711alaw.pcap	293,6	1267	G.711 *A*-law (2)
g711ulaw.pcap	325,3	1405	G.711 μ-law (2)
g729br8.pcap	115,1	1330	G.729b (2)
g729r8.pcap	262,3	2835	G.729 (2)

Tabuľka 6.2: Zoznam testovacích súborov vytvorených pomocou smerovačov Cisco 2811

Tretia vzorka vznikla za použitia sieťového testera od svetovo uznávanej firmy Ixia[3]. Tester po zakúpení príslušnej licencie umožnuje použiť aj VoIP modul. Tento slúži na testovanie VoIP ústrední, telefónov, brán a podobne. Obsahuje možnosť simulovať hovor s použitím používateľsky definovaného zvukového súboru, ktorý sa následne zakóduje, paketizuje a odošle pomocou RTP streamu s potrebnou signalizáciou rovnako, ako keby na mieste testera bol reálny telefón. Aj keď toto zariadenie nie je primárne určené na generovanie VoIP komunikácie, mohli sme možnosti VoIP modulu využiť pri takzvanom zapojení „wire-under-test“, v ktorom sa simulujú obe strany hovoru. Pri tomto zapojení sa prepojili dva porty testera cez manažovateľný prepínač, na ktorom bol aktivovaný monitorovací port, ktorý odosielal komunikáciu medzi portami testera na zachytávajúci počítač. Na oboch portoch testera bol nakonfigurovaný test, ktorý na jednej strane začínal zahájením hovoru pomocou signalizačného protokolu SIP, nadviazania hovoru a odosielania súboru s vloženou dátovou stopou. Na druhom porte boli nastavenia obdobné s rozdielom, že druhá strana hovor neinicializovala, ale odpovedala na prichádzajúcu požiadavku na vytvorenie hovoru. Pre obe strany bol vždy použitý rovnaký vstupný audio súbor. Obe strany mali zakázané všetky kodeky, s výnimkou aktuálne zachytávaného kodeku. Na zachytávajúcom počítači zachytával komunikáciu program Wireshark v promiskuitnom móde. Tento test bol pre všetky zachytávané kodeky vykonaný dva krát, s dvomi rôznymi vstupnými zvukovými stopami (baapi-a, babgk-a) s rovnakou dĺžkou. Zoznam zachytených súborov v tejto testovacej vzorke je zaznamenaný v tabuľke 6.3.

6.2 Zistené charakteristické znaky

Prvým z použiteľných charakteristických znakov je položka `payload type` v samotnej RTP hlavičke. Ako už bolo spomenuté v kapitole 3, tieto sa delia na statické a dynamické pričom prieskum medzi nimi ukázal, že všetky testované aplikácie a hardvérové zariadenia statické mapovanie dodržiavali, a naopak u dynamických profilov sa nevyskytuje žiadna opakovaná hodnota. Jedinou výnimkou je používaný `payload type` číslo 101. Tento ale neslúži na

[2] http://www.cisco.com/c/en/us/products/collateral/unified-communications/unified-communications-manager-express/product_data_sheet0900aecd8041c303.html

[3] http://www.ixiacom.com/

Názov súboru	Veľkost (MB)		Počet paketov		Kodek (počet streamov)
	baapi-a	babgk-a	baapi-a	babgk-a	
alaw.pcap	11,1	11,1	48264	48265	G.711 *A*-law (2)
amr-12.pcap	5	5	48254	48252	AMR 12,2kbps (2)
g726-16.pcap	8,7	8,7	96287	96269	G.726 16kbps (2)
g726-24.pcap	9,6	9,6	96255	96267	G.726 24kbps (2)
g726-32.pcap	10,6	10,6	96259	96279	G.726 32kbps (2)
g726-40.pcap	11,5	11,5	96275	96277	G.726 40kbps (2)
g729.pcap	4,3	4,3	48254	48259	G.729 (2)
g729a.pcap	4,3	4,3	48257	48252	G.729a (2)
g729b.pcap	3,7	3,6	42511	41127	G.729b (2)
g7231-5k.pcap	2,9	2,9	32399	32241	G.723.1 5,3kbps (2)
g7231-6k.pcap	3,0	3,0	32249	32247	G.723.1 6,3kbps (2)
ulaw.pcap	11,1	11,1	48242	48247	G.711 μ-law (2)

Tabuľka 6.3: Zoznam testovacích súborov vytvorených pomocou sieťového testeru firmy Ixia

prenos samotného hovoru ale tónovej voľby (DTFM) podľa RFC 4733. Táto sa využíva preto, že kodeky používajúce komprimáciu môžu prenášať tónovú voľbu nesprávne.

Analýza odchytených dát potvrdila skôr zistené správanie hodnôt položky `payload type`, a preto môžeme prehlásiť hodnoty využívajúce statické mapovanie kodekov za charakteristický znak. Preto kodeky využívajúce toto mapovanie máme už v prvom kroku detekované a ďalšie znaky budú už teda slúžiť len na spresnenie detekcie, napríklad detekciou konkrétnej variácie kodeku. Bohužiaľ táto položka je pre dynamicky mapované kodeky absolútne nepoužiteľná a detekcia týchto sa bude musieť spoliehať na iné charakteristické znaky.

Po preskúmaní zachytených dát sa ukázalo, že aj keď sa dĺžka paketov pri rovnakom kodeku naozaj mení aj v základných nastaveniach, jej pomer voči rozdielu medzi časovými razítkami po sebe nasledujúcich paketov tvorí pomerne unikátnu vlastnosť. Toto správanie je spôsobené tým, že väčšina kodekov používaných na prenos hlasu má statickú vzorkovaciu frekvenciu a aj šírku pásma. Ďalším faktorom je, že časové razítka súvisia so vzorkovacou frekvenciou. U všetkých uvedených kodekov označovali počet vzoriek signálu nesených jednotlivými paketmi. Z uvedeného vyplýva, že jediná premenná hodnota je doba paketizácie. Preto pri jej zvyšovaní, resp. znižovaní sa rozdiel medzi časovými razítkami a veľkosť paketov mení stále v tom istom pomere. Tieto výsledky sú zhrnuté do tabuľky 6.4.

Táto metóda však bude nedostačujúca v prípade požiadavky na detekciu kodekov s variabilným dátovým tokom. Vo VoIP aplikáciach je kódovanie hlasu takýmto kodekom pomerne vzácne. Hlas je totižto pomerne kritický s ohľadom na oneskorenie a vyžaduje sa držanie prenosového pásma. Kodeky s konštantným dátovým tokom nám zaručujú, že pri náhlom zvýšení požiadaviek na prenosové pásmo sa nezvýši aj oneskorenie alebo zvuk dokonca nevypadne. Naviac pri dnešných kapacitách prenosových liniek nie je úspora dosiahnutá vďaka použitiu zvukových kodekov s variabilnými dátovými tokmi vo VoIP podstatná. Jediným kodekom využívajúcim variabilný dátový tok, ktorý bol skúmaný v rámci našej testovacej vzorky bol kodek Silk. Tento ale nie je vo VoIP aplikáciach bežne využívaný. Využíva ho najmä aplikácia Skype, ktorá nevyužíva pre prenos hlasu protokol RTP. Existuje jeho implementácia, ktorá je použiteľná v iných softvérových riešeniach. Jej licencia však zakazuje použitie pre iné ako testovacie účely.

Codec	**Payload type**	**Δ time**	**Data len**	**Pomer**
AMR-WB	dyn	320	62	160:31
AMR-12k	dyn	160	33	160:33
GSM	3	160	33	160:33
G.722	9	160	160	1:1
G.722.1	dyn	320	60	16:3
G.723.1 5 kbps	4	240	20	12:1
G.723.1 6 kbps	4	240	24	10:1
G.726 16 kbps	dyn	80/240	20/60	4:1
G.726 24 kbps	dyn	80/240	30/90	8:3
G.726 32 kbps	dyn	80/240	40/120	2:1
G.726 40 kbps	dyn	80/240	50/150	8:5
G.729	18	160	20	8:1
G.729a	18	160	20	8:1
G.729b	18	160*	20*	var.
G.711 *A*-law	8	160	160	1:1
G.711 μ-law	0	160	160	1:1
Speex 16 kHz	dyn	320	52	80:13
Speex 8 kHz	dyn	160	20	8:1
Silk 16 kHz	dyn	640	var.	var.
Silk 8 kHz	dyn	320	var.	var.

* – skrátené v prípade detekcie ticha

Tabuľka 6.4: Zoznam zistených vlastností kodekov v RTP

Kapitola 7

Detekčná aplikácia

Ako vzor aplikácie bola použitá aplikácia pre detekciu RTP streamov v jazyku Python, ktorá vznikla na základe práce [3]. Táto však bola priveľmi pomalá a preto sa ukázalo, že bude vhodné jej prepísanie do iného programovacieho jazyka a až následné rozšírenie o detekcie kodeku.

Voľba padla na jazyk C++ vďaka jeho prenositeľnosti na úrovni zdrojových kódov medzi viacerými platformami, výkonnosti výsledných aplikácií a dobrej podpory paralelizmu. Ďalej bola použitá knižnica `libpcap` pre Linux/FreeBSD a `winpcap` pre Windows. Tieto umožňujú odchytávanie paketov alebo načítanie zo súborov a sú používané naprieč širokej škále aplikácií. Pre zjednodušenie implementácie, zlepšenie prenositeľnosti a jednoduchému rozšíreniu o grafické používateľské rozhranie v prípade potreby bola použitá aj knižnica `Qt`, ktorá je tak isto dostupná na Linux, FreeBSD aj Windows.

7.1 Detekčný algoritmus

Detekcia protokolu RTP je založená na algoritme popisovanom v [3] a nie je predmetom tejto práce.

Detekcia kodeku použitého pri zakódovaní dát nesených RTP tokom je založená na viacprechodovom filtri. Tento využíva tabuľku, ktorá obsahuje jednotlivé podporované kodeky. Tabuľka nesie pre každý kodek informáciu o jeho internom identifikátore, názve, používanom `payload type` a pomere medzi veľkosťou nesených dát v jednom pakete a rozdiele medzi časovými razítkami dvoch po sebe nasledujúcich paketov. Tieto sú zjednodušené tak, že všetky dynamické hodnoty položky `payload type` sú mapované na jednotnú hodnotu. Tak isto premenné hodnoty rozdielu časových razítok a veľkosti dát nesených paketom sú v prípade premenných hodnôt mapované na jednu hodnotu.

Vstupom algoritmu je hodnota `payload type`, veľkosť dát nesených RTP paketom a rozdiel medzi časovými razítkami. Po vstupe sa hodnoty upravujú nasledovne:

1. V prípade, že je hodnota položky `payload type` zodpovedajúca niektorej z dynamicky mapovaných hodnôt, premapuje sa táto na hodnotu označujúcu všetky dynamické kodeky v internej tabuľke algoritmu.

2. Pre hodnoty veľkosti dát nesených RPT paketom a rozdielu medzi časovými razítkami sa nájde najväčší spoločný deliteľ.

3. Obe hodnoty sa vydelia ich najväčším spoločným deliteľom a ďalej budeme pracovať s touto hodnotou.

Týmto postupom dosiahneme jednoduché porovnávanie k položkám v internej tabuľke kodekov.

Po spracovaní vstupov môžeme pristúpiť k samotnému filtrovaniu. Toto využíva pomocnú tabuľku, ktorá označuje, či kodek patrí alebo nepatrí do množiny možných kodekov. Samotné filtrovanie prebieha nasledovne:

1. Inicializuje sa pomocná tabuľka, kde sa všetky kodeky pridajú do množiny možných kodekov.
2. Pre každý kodek sa skontroluje:
   (a) Vyhovuje vstupná hodnota `payload type` hodnote uvedenej v pomocnej tabuľke? Ak nie, kodek sa vylúči z množiny možných kodekov.
   (b) Vyhovuje rozdiel medzi časovými razítkami paketov hodnote uvedenej v tabuľke? Ak nie, kodek sa vylúči z množiny možných kodekov.
   (c) Vyhovuje veľkosť dát nesených paketmi hodnote uvedenej v pomocnej tabuľke? Ak nie, kodek sa vylúči z množiny možných kodekov.
3. Prejde sa množina možných kodekov a algoritmus vráti zoznam možných kodekov.

V prípade pomerne obmedzenej množiny kodekov ako napríklad našej, je pravdepodobné, že táto množina filtrov bude postačujúca, avšak v prípade potreby doimplementovania kontroly ďalších špecifických znakov bude toto veľmi jednoduché pridanie ďalších filtrácií na spresnenie detekcie pred krok č. 3.

7.2 Implementácia

Implementácia aplikácie je napísaná, ako bolo už spomenuté, v jazyku C++ za použitia knižníc `Qt` a `libpcap`. Implementácia je založená na sade navzájom komunikujúcich objektov.

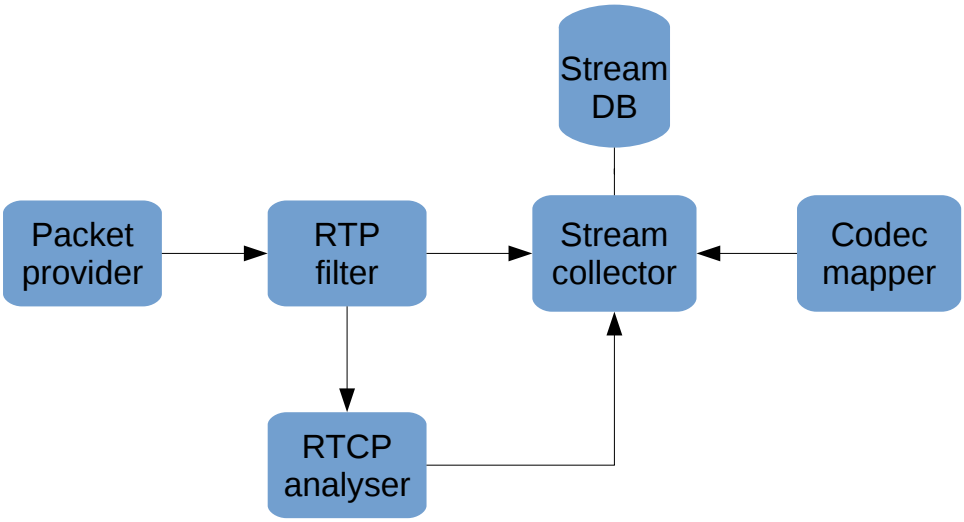
Prvý na radu prichádza objekt triedy `PacketProvider`. Tento sa stará o načítavanie paketov a spracovanie jednotlivých linkových vrstiev. Linkové vrstvy sa jednoducho preskakujú a dáta nimi nesené sa nespracovávajú. Rozhranie knižnice `libpcap` poskytuje informáciu o zachytenom linkovom protokole na danom zariadení. Pred odoslaním je teda len potrebné skontrolovať použitý linkový protokol a nastaviť veľkosť preskakovaných dát. Vďaka tomuto je pridanie podpory ďalších linkových protokolov veľmi jednoduché. Aktuálna verzia programu však podporuje výhradne protokol `ethernet`.

Príjemcom zachytených paketov je objekt triedy `RtpFilter`. Tento na základe „per-packet“ časti RTP detekčného algoritmu, vychádzajúceho z algoritmu na obrázku 2.3 vylúči všetky pakety, ktoré nezodpovedajú požiadavkám na RTP paket. Na rozdiel od algoritmu na obrázku, je v implementácii vynechaná kontrola položky `padding` a následnej možnosti odrezať zarovnávajúce bajty, ktoré sa ukázali ako problematické. Vylepšením algoritmu je, že v prípade detekcie hodnoty položky `payload type` zodpovedajúcej jednej z hodnôt vyhradeným pre zamedzenie konfliktu s RTCP, je tento zaslaný objektu triedy `RtcpAnalyser`. Týmto sa eliminuje potreba dvojitej kontroly niektorých hodnôt pre detekciu RTCP paketov. V prípade, že vstupný paket vyhovuje požiadavkám na RTP paket, je tento zaslaný objektu triedy `StreamCollector`.

Objekt triedy `RtcpAnalyser` sa stará o spracovanie detekovaných RTCP paketov. RTCP zložený paket rozdelí na jednotlivé elementy, informácie z elementárnych paketov sa spracujú a informácie o zdrojoch sa zašlú objektu triedy `StreamCollector`.

Trieda `CodecMapper` implementuje detekčný algoritmus popísaný v kapitole 7.1. Objekt tejto triedy poskytuje objektu triedy `StreamCollector` službu mapovania informácií zistených o streame na reťazec s názvom jedného alebo viacerých kodekov, ktoré mohli byť použité na zakódovanie dát nesených streamom.

Objekt triedy `StreamCollector` tvorí srdce samotnej aplikácie. Zhromažďuje informácie o zachytených RTP paketov do databázy streamov, zhromažďuje informácie o zdrojoch pochádzajúce z RTCP paketov, filtruje streamy s nedostatočným počtom paketov a po ukončení detekcie vypisuje zistené informácie. Pri výpise informácií okrem iného využíva služby objektu triedy `CodecMapper`, a teda vypisuje zoznam možných kodekov.



Obr. 7.1: Schéma implementácie aplikácie

Kapitola 8

Testovanie na dátach

Pre testovanie a porovnanie implementácie boli použité dáta z kapitoly 6.1, v ktorých boli odstránené pakety obsahujúce signalizačné protokoly. Týmto bolo zaručené, že žiadna z aplikácií nepoužije na detekciu kodeku informácie zo signalizačného protokolu, ale bude sa musieť spoľahnúť na analýzu samotných RTP dát.

Pre porovnanie bola zvolená open-source aplikácia Wireshark[1] a aplikácia PacketScan od spoločnosti Gl Communications[2]. Pri aplikácii Wireshark bolo oproti pôvodným nastaveniam povolené hľadanie RTP paketov mimo signalizačných konverzácií. Aplikácie PacketScan bola ponechaná v základných nastaveniach. Tak isto pri aplikácii rtpinfo neboli pridané žiadne prepínače meniace základné správanie.

Aplikácie boli postupne spustené na vyššie uvedených súboroch a výsledky boli zaznamenané v nasledujúcich kapitolách.

8.1 Sada číslo 1

Sada číslo 1 pozostáva zo súborov uvedených v tabuľke 6.1 a namerané výsledky sú uvedené v tabuľke 8.1 pre aplikáciu rtpinfo, tabuľke 8.2 pre aplikáciu Wireshark a tabuľke 8.3 pre aplikáciu PacketScan.

Z výsledkov je zjavné, že žiadna z aplikácií nemala problém s detekciou streamov. Toto bolo predpokladateľné na základe obsahu súborov, ktoré okrem samotného hovoru obsahovali len servisné protokoly, a teda aplikácie nemali veľa priestoru na nesprávnu detekciu RTP. Je však vidieť podozrivo znížený počet detekovaných paketov u aplikácie PacketScan oproti aplikáciam rtpinfo a Wireshark.

Pri pohľade na výsledky detekcie kodekov je vidieť, že v tomto ohľade je najúspešnejšia aplikácia rtpinfo. Problém mala iba s kodekom Silk. Po preskúmaní bola príčina okamžite zjavná. V RTP hlavičkách sa v položke `payload type` namiesto čísla z rozsahu pre dynamicky mapované kodeky použila hodnota 92, ktorá zodpovedá jednej z nepridelených hodnôt pre statické mapovanie, a teda tento stream nerešpektuje dokument [5]. Bez tohto problému by bol kodek identifikovaný, keďže sa jedná o jediný kodek s premenlivým pomerom medzi rozdielom časových razítok a dĺžky dát nesených paketom s použitím dynamického mapovania na hodnotu `payload type` v mapovacej tabuľke.

Aplikácia Wireshark rozpoznala všetky staticky mapované kodeky správne, avšak pri výskyte kodeku s dynamicky mapovanou hodnotou `payload type` nedokázala kodek určiť.

[1] http://www.wireshark.org/
[2] http://www.gl.com/packetscan.html

Aplikácia PacketScan dokázala taktiež správne určiť len staticky mapované kodeky. U zvyšných kodekov sa pokúsila o detekciu kodeku, avšak neúspešne. Kodek bol vo všetkých prípadoch s dynamickou hodnotou `payload type` buď detekovaný nesprávne, alebo nebol detekovaný vôbec.

Súbor	**Streamov**	**Paketov**	**Kodek**
amr-wb.pcap	2	717	AMR-WB
g722.pcap	2	736	G.722
g726-16.pcap	2	609	G.726 16kbps
g726-24.pcap	2	583	G.726 24kbps
g726-32.pcap	2	588	G.726 32kbps
g726-40.pcap	2	623	G.726 40kbps
g7221.pcap	2	846	G.722.1
gsm.pcap	2	796	GSM 06.10
pcma.pcap	2	822	G.711 *A*-law
pcmu.pcap	2	791	G.711 μ-law
silk8.pcap	2	663	Neznámy
silk16.pcap	2	631	Neznámy
speex8.pcap	2	804	Speex 8kHz
speex16.pcap	2	1015	Speex 16kHz

Tabuľka 8.1: Výsledky testov sady č. 1 aplikáciou rtpinfo

Súbor	**Streamov**	**Paketov**	**Kodek**
amr-wb.pcap	2	717	Neznámy
g722.pcap	2	736	G.722
g726-16.pcap	2	609	Neznámy
g726-24.pcap	2	583	Neznámy
g726-32.pcap	2	588	Neznámy
g726-40.pcap	2	623	Neznámy
g7221.pcap	2	846	Neznámy
gsm.pcap	2	796	GSM 06.10
pcma.pcap	2	822	G.711 *A*-law
pcmu.pcap	2	791	G.711 μ-law
silk8.pcap	2	663	Neznámy
silk16.pcap	2	631	Neznámy
speex8.pcap	2	804	Neznámy
speex16.pcap	2	1015	Neznámy

Tabuľka 8.2: Výsledky testov sady č. 1 aplikáciou Wireshark

8.2 Sada číslo 2

Sada číslo 2 pozostáva zo súborov uvedených v tabuľke 6.2 a namerané výsledky sú uvedené v tabuľke 8.4 pre aplikáciu rtpinfo, tabuľke 8.5 pre aplikáciu Wireshark a tabuľke 8.6 pre aplikáciu PacketScan.

Súbor	Streamov	Paketov	Kodek
amr-wb.pcap	2	707	G.726 32kbps
g722.pcap	2	726	G.722
g726-16.pcap	2	599	G.726 40kbps
g726-24.pcap	2	573	EVRCC
g726-32.pcap	2	578	G.711 μ-law
g726-40.pcap	2	615	G.711 A-law
g7221.pcap	2	838	AMR-WB
gsm.pcap	2	786	GSM 06.10
pcma.pcap	2	812	G.711 A-law
pcmu.pcap	2	781	G.711 μ-law
silk8.pcap	2	653	Neznáme
silk16.pcap	2	621	Neznáme
speex8.pcap	2	794	Speex 16kHz
speex16.pcap	2	1005	iSAC

Tabuľka 8.3: Výsledky testov sady č. 1 aplikáciou PacketScan

Z pohľadu detekcie RTP sa opakovala situácia z predchádzajucich testov. Aplikácie rtpinfo a Wireshark mali detekované všetky streamy aj pakety správne. Aplikácia PacketScan hlásila nižší počet RTP paketov.

Detekciu kodeku v tejto sade zvládli všetky aplikácie správne. Toto je spôsobené použitím kodekov s výhradne statickým mapovaním. Aplikácia rtpinfo však naviac dokázala rozlíšiť kodek G.729b, ktorý nie je kompatibilný s kodekmi G.729 a G.729a. Táto informácia by bola dôležitá napríklad pri rekonštrukcii hovoru, pretože dekódovanie dát dekóderom pre kodek G.729 by v tomto prípade skončilo neúspešne.

Súbor	Streamov	Paketov	Kodek
g711alaw.pcap	2	1245	G.711 A-law
g711ulaw.pcap	2	1379	G.711 μ-law
g729br8.pcap	2	1291	G.729b
g729r8.pcap	2	2797	G.729/G.729a

Tabuľka 8.4: Výsledky testov sady č. 2 aplikáciou rtpinfo

Súbor	Streamov	Paketov	Kodek
g711alaw.pcap	2	1245	G.711 A-law
g711ulaw.pcap	2	1379	G.711 μ-law
g729br8.pcap	2	1291	G.729
g729r8.pcap	2	2797	G.729

Tabuľka 8.5: Výsledky testov sady č. 2 aplikáciou Wireshark

8.3 Sada číslo 3

Sada číslo 3 pozostáva zo súborov uvedených v tabuľke 6.3 a namerané výsledky sú uvedené v tabuľke 8.7 pre aplikáciu rtpinfo, tabuľke 8.8 pre aplikáciu Wireshark a tabuľke 8.9 pre

Súbor	Streamov	Paketov	Kodek
g711alaw.pcap	2	1235	G.711 *A*-law
g711ulaw.pcap	2	1369	G.711 μ-law
g729br8.pcap	2	1281	G.729
g729r8.pcap	2	2787	G.729

Tabuľka 8.6: Výsledky testov sady č. 2 aplikáciou PacketScan

aplikáciu PacketScan. Keďže táto sada obsahuje súbory založené na dvoch audio stopách, bolo v skutočnosti vykonané dvojnásobné množstvo testov. Výsledky sa však vo všetkých ohľadoch, až na počet detekovaných paketov zhodovali, a teda boli výsledky z každého programu agregované do jednej tabuľky.

V prípade RTP detekcie sa opäť opakovala situácia z predchádzajúcich testov. Aplikácie rtpinfo a Wireshark mali detekované všetky streamy aj pakety správne. Aplikácia PacketScan hlásila nižší počet RTP paketov.

Aplikácia rtpinfo v tejto testovacej sade predviedla 100 % úspešnosť. Všetky kodeky detegovala správne vrátane variant kodekov G.723.1, G.729 a G.726. Aplikácia Wireshark detegovala opäť správne len staticky mapované kodeky, ale u kodekov G.729 a G.723.1 nevedela rozlíšiť ich varianty. Dynamicky mapované kodeky nedokázala detegovať vôbec. Podobná situácia nastala aj u aplikácie PacketScan s tým rozdielom, že u niektorých dynamických kodekov sa aplikácia pokúsila o detekciu, ale vždy s nesprávnym výsledkom.

Súbor	Streamov	Paketov		Kodek
		baapi-a	babgk-a	
alaw.pcap	2	48010	48016	G.711 *A*-law
amr-12.pcap	2	48010	48016	AMR 12,2kbps
g726-16.pcap	2	96020	96032	G.726 16kbps
g726-24.pcap	2	96020	96032	G.726 24kbps
g726-32.pcap	2	96020	96032	G.726 32kbps
g726-40.pcap	2	96020	96032	G.726 40kbps
g729.pcap	2	48010	48016	G.729/G.729a
g729a.pcap	2	48010	48016	G.729/G.729a
g729b.pcap	2	42262	40882	G.729b
g7231-5k.pcap	2	32008	32012	G.723.1 5,3kbps
g7231-6k.pcap	2	32008	32012	G.723.1 6,3kbps
ulaw.pcap	2	48809	48016	G.711 μ-law

Tabuľka 8.7: Výsledky testov sady č. 3 aplikáciou rtpinfo

8.4 Test rýchlosti

Rýchlosť aplikácie bola testovaná opäť na troch vyššie uvedených sadách dát. Z dôvodu problematického porovnávania programov s grafickým užívateľským rozhraním a aplikácie rtpinfo, ktorá funguje ako konzolová aplikácia nebude táto porovnávaná s aplikáciami Wireshark a PacketScan. Pre tento test neboli z testovacích sád odstránené dáta signalizačných protokolov. Test prebehol spustením aplikácie na jednom súbore 10 krát a v nasledujúcich tabuľkách sú uvedené priemerné časy behov na jednotlivých súboroch. Všetky testy prebie-

Súbor	Streamov	Paketov		Kodek
		baapi-a	babgk-a	
alaw.pcap	2	48010	48016	G.711 *A*-law
amr-12.pcap	2	48010	48016	Neznámy
g726-16.pcap	2	96020	96032	Neznámy
g726-24.pcap	2	96020	96032	Neznámy
g726-32.pcap	2	96020	96032	Neznámy
g726-40.pcap	2	96020	96032	Neznámy
g729.pcap	2	48010	48016	G.729
g729a.pcap	2	48010	48016	G.729
g729b.pcap	2	42262	40882	G.729
g7231-5k.pcap	2	32008	32012	G.723.1
g7231-6k.pcap	2	32008	32012	G.723.1
ulaw.pcap	2	48809	48016	G.711 μ-law

Tabuľka 8.8: Výsledky testov sady č. 3 aplikáciou Wireshark

Súbor	Streamov	Paketov		Kodek
		baapi-a	babgk-a	
alaw.pcap	2	48000	48006	G.711 *A*-law
amr-12.pcap	2	48000	48006	G.726 24kbps
g726-16.pcap	2	96010	96022	Neznámy
g726-24.pcap	2	96010	96022	Neznámy
g726-32.pcap	2	96010	96022	Neznámy
g726-40.pcap	2	96010	96022	GSM-EFR
g729.pcap	2	48000	48006	G.729
g729a.pcap	2	48000	48006	G.729
g729b.pcap	2	42252	40872	G.729
g7231-5k.pcap	2	31998	32002	G.723.1
g7231-6k.pcap	2	31998	32002	G.723.1
ulaw.pcap	2	48799	48006	G.711 μ-law

Tabuľka 8.9: Výsledky testov sady č. 3 aplikáciou PacketScan

hali na stolnom počítači s procesorom Intel Core i5-4670K s taktom 3,40 GHz, 8 GB DDR3 a SSD diskom. Ako operačný systém bola použitá Linuxová distribúcia Ubuntu 14.04 skompilovaná pre architektúru x86_64.

Výsledky testov na prvej testovacej sade sú uvedené v tabuľke 8.10. V prvej testovacej sade boli rozdiely medzi jednotlivými súbormi veľmi malé a boli skôr spôsobené chybou merania a rozdielnym zaťažením operačného systému. Pri takto malých súboroch sa pravdepodobne viac prejavila réžia pri spustení a ukončení aplikácie ako záťaž spôsobená samotným algoritmom.

Na druhej testovacej sade sa nám výsledky, uvedené v tabuľke 8.11, viac-menej opakovali. Doby behu sa celkovo nepatrne zvýšili, ale stále nemôžeme vylúčiť, že toto bolo spôsobené chybou merania.

Pri testoch na sade číslo 3 boli výsledné rýchlosti, medzi vzorkami z rôznych audio súborov, opäť veľmi podobné a rozdiely medzi nimi boli skôr spôsobené chybou merania a rozdielnym zaťažením operačného systému. Preto do tabuľky 8.12 bol použitý priemer

Názov súboru	Veľkosť (kB)	Počet paketov	Čas behu aplikácie (s)
amr-wb.pcap	101,5	736	0,011
g722.pcap	175,3	751	0,013
g726-16.pcap	85,3	625	0,012
g726-24.pcap	99,4	598	0,009
g726-32.pcap	117,9	604	0,011
g726-40.pcap	148,1	651	0,011
g7221.pcap	116,9	867	0,011
gsm.pcap	87,9	809	0,010
pcma.pcap	195,1	837	0,011
pcmu.pcap	188,8	810	0,012
silk8.pcap	89,1	678	0,011
silk16.pcap	82,6	642	0,010
speex8.pcap	80,4	822	0,012
speex16.pcap	131	1038	0,013

Tabuľka 8.10: Výsledky testov rýchlosti na sade č. 1

Názov súboru	Veľkost (kB)	Počet paketov	Čas behu aplikácie (s)
g711alaw.pcap	293,6	1267	0,014
g711ulaw.pcap	325,3	1405	0,013
g729br8.pcap	115,1	1330	0,013
g729r8.pcap	262,3	2835	0,013

Tabuľka 8.11: Výsledky testov rýchlosti na sade č. 2

z oboch hodnôt. Z týchto výsledkov si už ale môžeme urobiť obraz o rýchlosti aplikácie. Na rýchlosť behu aplikácie má najmä vplyv počet paketov v súbore a nie celková veľkosť súboru. Táto skutočnosť je vidieť pri výraznom zvýšení počtu paketov, napríklad pri kodeku G.726 oproti ostatným kodekom. Avšak pri zmene veľkosti jednotlivých súborov s podobným počtom paketov sa doba spracovania prakticky nemenila. Takéto správanie bolo očakávateľné, keďže aplikácia spracováva hlavičky paketov a nie dáta nimi nesené.

8.5 Zhodnotenie

Aplikácia rtpinfo dokázala dobré schopnosti detekcie kodekov z testovacej sady dát. Aplikácie Wireshark a PacketScan však dokázali správne detegovať len staticky mapované kodeky. Je nutné však podotknúť, že obe aplikácie správne pracovali v prípade detekcie zo signalizačných dát.

Najviac nespoľahlivou aplikáciou sa ukázala byť aplikácia PacketScan. Táto nielenže detegovala dynamicky mapované kodeky nesprávne, ale aj nehlásila správny počet detekovaných paketov. Tento bol vo väčšine prípadov menší presne o 10 a preto je na mieste podozrenie na chybu v implementácii ich počítadla.

Rýchlosť aplikácie sa ukázala byť dostatočná. Pri teste s najväčším počtom paketov v teste a teda súbore `g726-16.pcap`, ktorý sa priemerne spracoval za 0,078 s. Pri jeho veľkosti 8,7 MB sa aplikácia dostala na rýchlosť 111,5 MB/s. Takáto rýchlosť by bola dostatočná aj na detekciu v reálnom čase. Reálna rýchlosť aplikácie však môže byť ešte vyššia, keďže väčšina z paketov v súboroch bola skutočne RTP paketov, ktoré spôsobovali modifiká-

Názov súboru	**Veľkost (MB)**		**Počet paketov**		**Čas behu aplikácie (s)**
	baapi-a	babgk-a	baapi-a	babgk-a	
alaw.pcap	11,1	11,1	48264	48265	0,051
amr-12.pcap	5	5	48254	48252	0,047
g726-16.pcap	8,7	8,7	96287	96269	0,078
g726-24.pcap	9,6	9,6	96255	96267	0,079
g726-32.pcap	10,6	10,6	96259	96279	0,079
g726-40.pcap	11,5	11,5	96275	96277	0,078
g729.pcap	4,3	4,3	48254	48259	0,048
g729a.pcap	4,3	4,3	48257	48252	0,043
g729b.pcap	3,7	3,6	42511	41127	0,041
g7231-5k.pcap	2,9	2,9	32399	32241	0,036
g7231-6k.pcap	3,0	3,0	32249	32247	0,034
ulaw.pcap	11,1	11,1	48242	48247	0,050

Tabuľka 8.12: Výsledky testov rýchlosti na sade č. 3

cie v databáze RTP streamov. V skutočnej prevádzke by bola väčšina paketov vylúčená zo spracovania ešte v procese filtrovania paketov. Rýchlosť by tak isto mohla byt naďalej zvyšovaná podporou paralelizácie, hardvérovou akceleráciou, prípadne lepšou optimalizáciou interných štruktúr.

Kapitola 9

Záver

Cieľom tejto práce bol výskum možností detekcie multimediálnych kodekov, ktoré na prenos medzi komunikujúcimi stranami používaju protokol RTP.

V tejto problematike boli uvažované dve potenciálne funkčné metódy, každá so špecifickými výhodami a nevýhodami. V prípade detekcie pomocou klasifikátora sú medzi hlavnými výhodami schopnosť detekcie aj na veľmi poškodených dátach, ale jeho najväčšou nevýhodou je pomerne vysoká výpočtová náročnosť. V prípade detekcie pomocou charakteristických znakov máme na jednej strane výhodu veľmi jednoduchých výpočtových úloh, a teda aj vyššej výkonnosti a jednoduchšej implementácie v hardvéri, ale aj nezanedbateľnú nevýhodu potreby ručnej analýzy dát a špecifických znakov, ako aj toho, že nájdené znaky nebudú dostatočné na identifikáciu kodeku.

Pre implementáciu bola zvolená metóda detekcie na základe charakteristických znakov. Táto bola implementovaná spolu s algoritmom pre detekciu RTP streamov v jazyku C++ do nástroja rtpinfo. Aplikácia bola otestovaná na sade dát a prejavila funkčnosť na sade kodekov používaných vo VoIP komunikácii. Bola schopná detegovať kodeky a ich varianty pri všetkých testovaných kodekoch s konštantným bitovým tokom, ktoré tvoria väčšinu použitých kodekov pri prenose hlasu vo VoIP aplikáciach pomocou protokolu RTP. Aplikácia je v aktuálnom stave plne vyhovujúca pre detekciu RTP streamov a detekcie vyššie uvedených kodekov, bude však ďalej vylepšovaná.

Medzi ďalšie kroky k zlepšeniu algoritmu bude patriť spresnenie detekcie pridaním možnosti detekcie kodekov s pomocou hľadania charakteristických znakov v samotných dátach nesených protokolom RTP. Týmto by sa mala umožniť aj detekcia kodekov pre kódovanie videa. Tieto totiž zväčša obsahujú samostatné hlavičky, ktoré umožnia ich detekciu. Medzi ďalšie vylepšenia aplikácie bude patriť ďalšia optimalizácia a zavedenie paralelizácie pre zrýchlenie celej aplikácie. V neposlednom rade sa zavedie použitie grafického používateľského rozhrania a automatizácie dekódovania streamov pre zjednodušenie obsluhy aplikácie operátorom.

Výsledky tejto práce budú zahrnuté v rámci výsledkov v projekte „Moderní prostředky pro boj s kybernetickou kriminalitou na Internetu nové generace“. Článok založený na tejto práci bol publikovaný na študentskej konferencii STUDENT EEICT 2014 a ďalší článok založený na tejto práci bol odoslaný na medzinárodnú konferenciu ICDF2C 2014 (International Conference on Digital Forensics & Cyber Crime).

Literatúra

[1] BAUGHER, M.; MCGREW, D.; NASLUND, M.; aj.: The Secure Real-time Transport Protocol (SRTP), RFC 3711. Technická zpráva, March 2004.
URL http://tools.ietf.org/html/rfc3711

[2] COSTEUX, J.-L.; GUYARD, F.; BUSTOS, A.-M.: Detection and Comparison of RTP and Skype Traffic and Performance. In *GLOBECOM*, San Francisco, CA: IEEE, 2006, ISSN 1930-529X.

[3] KMEŤ, M.: *Nástroj pro sledování RTP streamů*. Bakalářská práce, FIT VUT v Brně, Brno, 2012.

[4] POSTEL, J.: User Datagram Protocol, RFC 768. Technická zpráva, August 1980.
URL http://tools.ietf.org/html/rfc768

[5] SCHULZRINNE, H.; CASNER, S.: RTP Profile for Audio and Video Conferences with Minimal Control, RFC 3551. Technická zpráva, Júl 2003.
URL http://tools.ietf.org/html/rfc3551

[6] SCHULZRINNE, H.; CASNER, S.; FREDERICK, R.; aj.: RTP: A Transport Protocol for Real-Time Applications, RFC 3550. Technická zpráva, Júl 2003.
URL http://tools.ietf.org/html/rfc3550

[7] SUN, L.; MKWAWA, I.-H.; JAMMEH, E.; aj.: *Guide to Voice and Video over IP: For Fixed and Mobile Networks*. London: Springer, 2013, ISBN 978-1-4471-4904-0, 269 s.

[8] YARGICOGLU, A. U.; ILK, H. G.: Speech Coder Identification Using Chaotic Features Based on Steganalyzer Models. In *Communications*, Žilina: Žilinská univerzita v Žiline, 2012, ISSN 1335-4205.

Dodatok A

Obsah DVD

Priložený dátový nosič DVD obsahuje:

- text semestrálnej práce vo formáte PDF
- zdrojové súbory diplomovej práce pre systém LaTeX
- zdrojové súbory aplikácie rtpinfo
- domumentácia aplikácie rtpinfo generovaná systémom Doxygen
- aplikácia rtpinfo skompilovaná pre systém Linux (x86_64)
- sada testovacích dát
- sada testovacích dát s odstánenou signalizáciou

Dodatok B

Prehľad niektorých multimediálnych kodekov prenášaných pomocou RTP

encoding name	Názov	Ďalšie informácie
PCMU	G.711 PCM μ-Law	RFC 3551
GSM	GSM	RFC 3551
G723	G.723.1	RFC 3551
DVI4	IMA ADPCM	RFC 3551
LPC	Linear Predictive Coding	RFC 3551
PCMA PCMU	G.711 PCM *A*-Law	RFC 3551
G722	G.722	RFC 3551
L16	Linear PCM 16-bit	RFC 3551
QCELP	Qualcomm Code Excited Linear Prediction	RFC 2658
MPA	MPEG-1 or MPEG-2 Audio	RFC 2250
G728	G.728	RFC 3551
G729	G.729	RFC 3551
G726	G.726	RFC 3551
GSM-EFR	GSM-EFR	RFC 3551
L8	Linear PCM 8-bit audio	RFC 3551
RED	Redundant audio payload format	RFC 2198
VDVI	Variable rate DVI4	RFC 3551
CelB	Sun's CellB Video Encoding	RFC 2029
JPEG	JPEG Video	RFC 2435
nv	Xerox PARC's Network Video	RFC 3551
H261	H.261	RFC 4587
MPV	MPEG-1 or MPEG-2 Video	RFC 2250
MP2T	MPEG-2 transport stream	RFC 2250
H263	H.263	RFC 2190
H263-1998	H.263, second version	RFC 4629
speex	Speex Codec	RFC 5574
iLBC	Internet low Bitrate Codec	RFC 3951
CELT	CELT	www.celt-codec.org/
H264	H.264	RFC 3984
theora	Theora	www.theora.org/
MP4V-ES	MPEG-4 Visual	RFC 3016
BV32	BroadVoice Speech Codec	www.broadcom.com/support/broadvoice/
SIREN14	Polycom Siren 14/G 722.1C	http://www.polycom.com/company/about_us/technology/siren14_g7221c/
G7221	G.722.1	RFC 5577

Tabuľka B.1: Prehľad niektorých multimediálnych kódekov prenášaných pomocou RTP