



ТЕХНОЛОГИЧНО УЧИЛИЩЕ ЕЛЕКТРОННИ СИСТЕМИ
към ТЕХНИЧЕСКИ УНИВЕРСИТЕТ - СОФИЯ

ДИПЛОМНА РАБОТА

по професия код 523050 „Техник на компютърни системи“
специалност код 5230502 „Компютърни мрежи“

Тема: Разработване на технологично решение, свързващо два VoIP
доставчика на услуги чрез външен ENUM сървър

Дипломант:
Викторио Миланов

Научен ръководител:
инж. Владислав Василев

СОФИЯ

2024



ТЕХНОЛОГИЧНО УЧИЛИЩЕ
ЕЛЕКТРОННИ СИСТЕМИ
към ТЕХНИЧЕСКИ УНИВЕРСИТЕТ - СОФИЯ

Дата на заданието: 15.11.2023 г.

Утвърждавам:.....

Дата на предаване: 15.02.2024 г.

/проф. д-р инж. П. Якимов/

ЗАДАНИЕ за дипломна работа

ДЪРЖАВЕН ИЗПИТ ЗА ПРИДОБИВАНЕ НА ТРЕТА СТЕПЕН НА ПРОФЕСИОНАЛНА КВАЛИФИКАЦИЯ

по професия код 523050 „Техник на компютърни системи“

специалност код 5230502 „Компютърни мрежи“

на ученика Викторио Красимир Миланов 12 г клас

1. Тема: Разработване на технологично решение, свързващо два VoIP доставчика на услуги чрез външен ENUM сървър
2. Изисквания:
 - 2.1. Да се опишат основните протоколи, устройства и понятия, свързани с VoIP мрежите
 - 2.2. Да се опишат основните характеристики на Oracle SBC
 - 2.3. Да се проектира връзката между VoIP оператори, с функционалност за маршрутизация на обаждания чрез външен ENUM сървър и чрез локални политики (LP)
 - 2.4. Да се имплементира резервираност на устройствата
 - 2.5. Да се опише и реализира механизъм за манипулиране на SIP заглавия (header) с цел скриване на топологията на мрежата на оператора
 - 2.6. Да се реализира връзката между доставчиците на гласови услуги
 - 2.7. Да се тества работоспособността на изградената мрежа и всички заложени функционалности
3. Съдържание
 - 3.1 Обзор
 - 3.2 Същинска част
 - 3.3 Приложение

Дипломант :.....

Ръководител:.....

Вр.и.д. Директор:.....

ст. пр. д-р Веселка Христова /

Становище на дипломния ръководител и предложение за рецензент

По време на дипломната си разработка, дипломантът Викторио Миланов се отнасяше отговорно към поставените му задачи.

В процеса на работа се справи с множество предизвикателства и успя да реализира дипломната си работа в срок спрямо заданието.

Дипломната работа е изпълнена изцяло и описана в детайли, отговарящи на нивото й на сложност.

Предлагам дипломантът Викторио Миланов да бъде допуснат до защита на дипломната си работа, като за рецензент предлагам маг. инж. Венцислав Петков от фирма "Смартком-България" АД.

Дипломен ръководител:

(инж. Владислав Василев)

Съдържание

ЗАДАНИЕ.....	2
Съдържание.....	4
Увод.....	7
Глава 1. Основни понятия и устройства във Voice over Internet Protocol мрежите.....	8
1.1. Основни сведения за Voice over Internet Protocol.....	8
1.2. Voice over Internet Protocol телефони.....	9
1.2.1. IP телефонен апарат.....	9
1.2.2. Softphone - "Софтфон".....	10
1.3. Session Initiation Protocol.....	11
1.3.1. Основни сведения.....	11
1.3.2. Алтернативи на Session Initiation Protocol.....	12
1.3.3. Основни функционалности и особености.....	13
1.3.4. Обекти в Session Initiation Protocol.....	14
1.3.5. Сигнален трафик и принцип на работа.....	19
1.3.6. Разклоняване в Session Initiation Protocol.....	28
1.4. Session Description Protocol.....	30
1.4.1. Основни сведения.....	30
1.4.2. Структура на съобщенията.....	31
1.4.3. Полета в Session Description Protocol header.....	32
1.4.4. Атрибути в Session Description Protocol header.....	36
1.5. Real-time Transport Protocol.....	38
1.5.1. Основни сведения.....	38
1.5.2. Системи в Real-time Transport Protocol.....	39
1.5.3. RTP Хедър.....	40
1.5.4. Real-time Transport Control Protocol.....	45
1.5.5. Кодеци - основни сведения.....	47
Глава 2. Основни характеристики на Session Border Controller. 49	
2.1. Функционални изисквания към дипломната работа.....	49
2.1.1. Основни изисквания.....	49
2.1.2. Изисквания към разработеното решение.....	49
2.2. Основни сведения за Session Border Controller.....	50
2.3. Основни функционалности и приложения на Session Border Controller.....	51
2.3.1. Основни функционалности на Session Border Controller....	51
2.3.2. Основни приложения на Session Border Controller.....	53
2.4. Oracle Session Border Controller.....	55

2.4.1. Основни сведения и ползи на Oracle Session Border Controller.....	55
2.4.2. Преимущества на Oracle Session Border Controller пред устройства на други производители.....	56
Глава 3. Маршрутизация чрез E.164 Number Mapping сървър и локални политики. Манипулация на Session Initiation Protocol заглавия.....	59
3.1. DNS - Name Authority Pointer записи.....	59
3.1.1. Основни сведения за Domain Name System.....	59
3.1.2. Видове сървъри в Domain Name System.....	60
3.1.3. Основни видове записи в Domain Name System.....	62
3.1.4. NAPTR запис.....	62
3.2. E.164 Number Mapping.....	65
3.2.1. Основни сведения за E.164 Number Mapping.....	65
3.2.2. Процес на търсене на услуги, свързани с домейн в DNS, образуван от E.164 номер.....	69
3.3. Локални политики.....	73
3.3.1. Основни сведения за локални политики.....	73
3.3.2. Маршрутизация чрез локални политики.....	74
3.3.3. Предпочитание за пътища, въведени чрез локални политики.....	75
3.4. Манипулация на Session Initiation Protocol header.....	76
3.4.1. Основни сведения за Header Manipulation Rules.....	76
3.4.2. Приложения и функции на Header Manipulation Rules.....	77
3.4.3. Видове Header Manipulation Rules според начина на приложение.....	78
Глава 4. Конфигурация и реализиране на свързаността и основната функционалност.....	81
4.1. Топологии на мрежата.....	81
4.1.1. Логическая топология.....	81
4.1.2. Физическая топология.....	82
4.2. ACLI Интерфейс.....	83
4.2.1. ACLI.....	83
4.2.2. Конфигурационна йерархия в Oracle SBC.....	85
4.3. Установяване на достъп до SBC и базова конфигурация.....	88
4.3.1. Установяване на достъп до SBC.....	88
4.3.2. Базова конфигурация.....	88
4.4. Конфигуриране на интерфейси.....	92
4.4.1. Физически интерфейси - Physical Interfaces.....	93
4.4.2. Мрежови интерфейси - Network Interfaces.....	95

4.5. Реализиране на резервираност между SBC устройствата.....	97
4.6. Реализиране на маршрутизацията.....	102
4.6.1. Пространства.....	102
4.6.2. Модул "sip-config".....	103
4.6.3. SIP интерфейси.....	103
4.6.4. Модул "steering-pool".....	104
4.6.5. Session Agents.....	105
4.6.6. DNS сървър и база данни MariaDB.....	105
4.6.7. ENUM функционалност на SBC.....	111
4.6.8. Локални политики.....	112
4.7. Конфигурация на софтфони.....	113
4.8. Реализиране на манипулация на Session Initiation Protocol хедър.	
115	
Глава 5. Тестване на свързаността и доказване на работоспособността.....	117
5.1. Елементи и параметри на конфигурацията.....	117
5.1.1. Интерфейси.....	117
5.1.2. Конфигурация за резервираност.....	119
5.1.3. Пространства.....	119
5.1.4. Модул "sip-config" и SIP интерфейси.....	120
5.1.5. Session Agents.....	120
5.1.6. ENUM функционалност.....	121
5.1.7. Модул "steering-pool".....	123
5.1.8. Локални политики.....	123
5.1.9. Правила за манипулация на SIP заглавия.....	124
5.2. Резервираност на устройствата.....	124
5.3. Реализиране на обаждане.....	126
5.3.1. Осъществяване на Layer 3 свързаност.....	126
5.3.2. Последователност от стъпки за реализиране на обаждане...	
127	
5.3.3. Обаждане преди и след манипулация на Session Initiation Protocol хедър.....	131
Заключение.....	133
Използвани съкращения.....	135
Използвана литература.....	138

Увод

В днешни дни технологиите и услугите са се внедрили толкова много във всекидневието на човека, че хората не могат да си представят дори един ден без да се възползват от предимствата на някоя технология, особено тези, които са обвързани по някакъв начин с комуникация - предаване на глас или видео. Те могат да варират - от обаждания по телефона до конферентни връзки онлайн, позволяващи само с няколко докосвания или движения да се установи връзка с някого дори той да бъде на хиляди километри разстояние от повикващия.

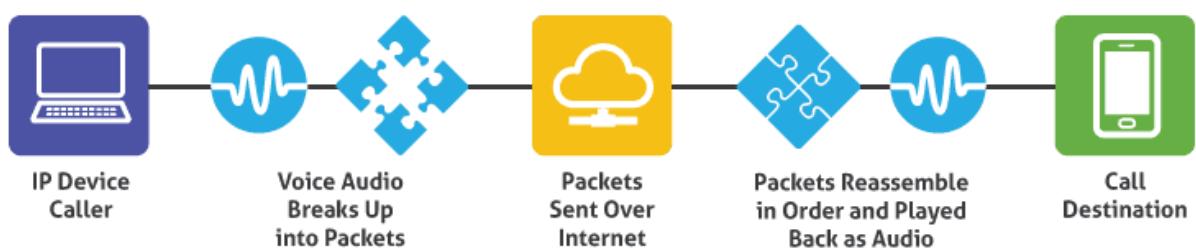
Именно за предоставянето на услуги като споменатите по-горе се използва VoIP - Voice over Internet Protocol. Използвайки като основа Интернет, "мрежата от мрежите" по света, VoIP осигурява услуги, които улесняват ежедневието на хората и ги свързват помежду им. С появата на все повече доставчици на услуги, свързаността и комуникацията им един с друг стават все по-важни и търсени.

Основната цел на настоящата дипломната работа е да се разработи технологично решение, свързващо два VoIP доставчика на услуги чрез външен E.164 Number Mapping (ENUM) сървър. Съответно трябва да се опишат основните протоколи, устройства и понятия, свързани с VoIP мрежите, както и основните характеристики на Session Border Controller (SBC). Нужно е да се имплементира функционалност за маршрутизация на обаждания чрез външен ENUM сървър и чрез локални политики (Local Policies), както и резервираност на устройствата. Реализацията на механизъм за манипулиране на SIP header (SIP заглавия) също е необходима. Целта на механизма е скриване на топологията на мрежата на оператора. Накрая трябва да се тества работоспособността на изградената мрежа и всички заложени функционалности.

Глава 1. Основни понятия и устройства във Voice over Internet Protocol мрежите

1.1. Основни сведения за Voice over Internet Protocol

Voice over Internet Protocol, широко известен като VoIP, е технология или протокол, отговорен за пренасяне на трафик, съдържащ глас (или звук под някаква форма), върху протокола IP. Използва голям набор от технологии и протоколи за изпълнение на своето предназначение. Намира своето приложение в широк спектър от приложения и платформи за видео и аудио връзки - пример за това могат да бъдат Skype, Discord, Google Meet, Zoom и още много други. Напоследък протоколът започва да се използва и за телефонни обаждания, като трансформира телефонните обаждания в данни, които могат да бъдат изпратени през интернет. VoIP всъщност действа, като чрез набор от технологии и други протоколи модифицира звук, като той най-често е човешки глас [1]. Трансформираният гласов сигнал се компресира и след това се изпраща през интернет под формата на пакети с помощта на IP протокола. VoIP сървърите могат да свързват обаждания към различни телефонни мрежи. При получаване данните се декомпресират за или от приемаща страна, за да се възприеме звука, който е генериран в рамките на разговора (Фиг.1.1).



Фиг.1.1 - Преработката на гласов сигнал и изпращане в интернет

1.2. Voice over Internet Protocol телефони

VoIP телефонът е всеки телефон, който използва интернет връзка за осъществяване и приемане на повиквания [2]. Съществуват най-общо два вида.

1.2.1. IP телефонен апарат

Хардуерното изражение на VoIP телефоните се. Алтернатива на традиционните стационарни телефони. За разлика от тях, при VoIP телефоните не е нужно присъствието и имплементирането на вече остарялата система PBX (Private Branch eXchange). Това дава множество преимущества на VoIP системите. Разликите във външния вид между VoIP телефон и стационарен телефон могат да се видят на *Фиг.1.2* и *Фиг.1.3*.



Фиг.1.2 - IP телефонен апарат на компанията "Nextiva"



Фиг.1.3 - Стационарен телефон KX-TS500 на марката *Panasonic*

Често срещана разлика между стационарните телефони и VoIP телефоните е, че при VoIP телефонните апарати обикновено върху самото устройство има дисплей и съответно потребителски интерфейс, позволяващ на клиентите да конфигурират и променят някои от неговите параметри. По-широк набор от параметрите на VoIP телефони се променят през достъпване на IP адреса на телефона чрез интернет.

1.2.2. Softphone - "Софтфон"

Софтуерното изражение на VoIP телефоните. Софтфоните не са ограничени до определена физическа локация. Функцията на VoIP телефоните е симулирана чрез използване на приложение за провеждане на разговори от настолен компютър или друго устройство [\[2\]](#).

1.3. Session Initiation Protocol

1.3.1. Основни сведения

Протоколът за иницииране на сесии (Session Initiation Protocol - SIP) се използва за сигнализиране и управление на интерактивни комуникационни сесии [3]. Тези комуникационни сесии могат да варират - могат да бъдат гласови, видео, чат или текстови съобщения. Протоколът SIP все повече се използва за осигуряване на VoIP и като цяло в телефонията, като в наши дни е неизменна част от тези архитектури. Протоколът работи на последния слой на Open Systems Interconnection (OSI) модела - апликационния. SIP е протокол, разработен предимно от работната група SIPCORE на Internet Engineering Task Force (IETF). SIP използва текстови съобщения, базирани на моделът "request - response" - заявка (от клиента) и отговор (от сървъра), следователно е подобен по действие на протокола HTTP - HyperText Transfer Protocol (или HTTPS - HyperText Transfer Protocol Secure). SIP може да се използва за управление на интернет конферентни, телефонни разговори или обмяна на съобщения през интернет.

SIP може да използва както Transmission Control Protocol (TCP), така и User Datagram Protocol (UDP) като протоколи на транспортния слой. Дори и при използването на UDP (който не е надежден, за разлика от TCP) за по-бързо предаване, VoIP гарантира надеждността на трафика и достигането на пакетите навреме с други протоколи и технологии, като например Real-time Transport Protocol (RTP).

1.3.2. Алтернативи на Session Initiation Protocol

SIP е алтернатива на препоръката на ITU (International Telecommunications Union) - H.323. H.323 е един от най-старите стандарти, използвани за VoIP телефония и видеоконферентни връзки и представлява набор от протоколи, позволяващи предаването на различни по вид комуникационни данни.

Разликите между двата протокола са показани в Табл.1.1 [4].

Протокол	SIP	H.323
Стандартизирано от	IETF	ITU
Произход	Интернет	Телефония
Архитектура	Модуларна	Изолирана
Мащабируемост	Голяма	Малка
Кодиране	Текстов формат	Двоичен формат
Протокол за сигнализация	TCP	TCP и/или UDP
Конферентни връзки	Хардуерно (чрез чипове)	Чрез IP Multicast

Табл.1.1 - Разлики между SIP и H.323

H.323 е по-малко мащабирам и гъвкав, и мрежите, изградени чрез H.323 са по-сложни за конструиране и адаптиране към промени от своя еквивалент SIP. Конферентните връзки се осъществяват чрез IP Multicast пакети при H.323, което не е особено гъвкав вариант.

Съществуват и други конкуренти на SIP (например H.248), но, заради споменатите по-горе предимства, SIP ги превъзхожда. Поради своите преимущества се е доказал като най-разпространения протокол за изграждане на VoIP сесии в момента. Протоколът SIP е избран за целите на дипломната работа именно заради тези специфики, както и заради своята популярност.

1.3.3. Основни функционалности и особености

SIP поддържа множество функции, които го правят предпочитан за широка гама решения. Най-важните функционалности на протокола са следните [3]:

- SIP поканите се използват за създаване на сесии и носят описания на параметри за сесиите (посредством протокола за описание на сесиите - Session Description Protocol), което позволява на участниците да се споразумеят за набор от съвместими типове трафик и други параметри. По този начин SIP не е ограничен до конкретен тип преносна среда (media) и следователно може да работи с разширяващия се набор от технологии и преносни среди.
- SIP дава възможност за мобилност на потребителите чрез механизъм, който позволява заявките да бъдат проксираны или пренасочени към текущото местоположение на потребителя, след като те биват регистрирани в SIP мрежа.
- SIP поддържа удостоверяване "end-to-end" - през целия път на пакетите в мрежата и "hop-by-hop" - на всеки hop, тоест всеки път когато пакета премине през дадено устройство и се обработи от него. Поддържа се и "end-to-end" криптиране с помощта на различни технологии.
- SIP е независим от транспортния протокол на по-ниско ниво, което му позволява да се възползва от предимствата на двата най-известни транспортни протокола (TCP и UDP), както и на бъдещи нови разработки на транспортния слой. Най-често за сигнализационен трафик се използва TCP на порт 5060 или 5061,

а за обаждания и гласова връзка - UDP, също на порт 5060 или 5061 [\[5\]](#).

- Протоколът SIP може да бъде използван за много различни видове комуникационни сесии, тоест не е лимитиран единствено до гласови или видео такива.

1.3.4. Обекти в Session Initiation Protocol

Един SIP обект може да работи в един от следните режими [\[6\]](#):

- User Agent - UA. Потребителският агент е крайната точка (устройство) на една SIP сесия или повикване, най-често VoIP телефон. Той изпраща и форматира SIP заявки съгласно инструкциите. Отговорен е и за кодирането и декодирането на информацията - записва и предава звук. UA могат да бъдат два типа:
 - **User Agent Client (UAC)** – Обектът, който изпраща заявка и получава отговор, тоест клиентът в сесията.
 - **User Agent Server (UAS)** – Обектът, който получава заявка и изпраща съответния отговор - явява се сървър в сесията.
- Proxy server - използва се за маршрутизиране на заявките и за прилагане на политики или защитни стени. Той приема заявки от името на един потребител и ги предава на друг потребител, като има възможност да ги променя при необходимост. Препраща с помощта на AOR - Address Of Record. Прокси сървърът се намира между два UA. Между източника на обаждането и дестинацията може да има максимум 70 прокси сървъра.

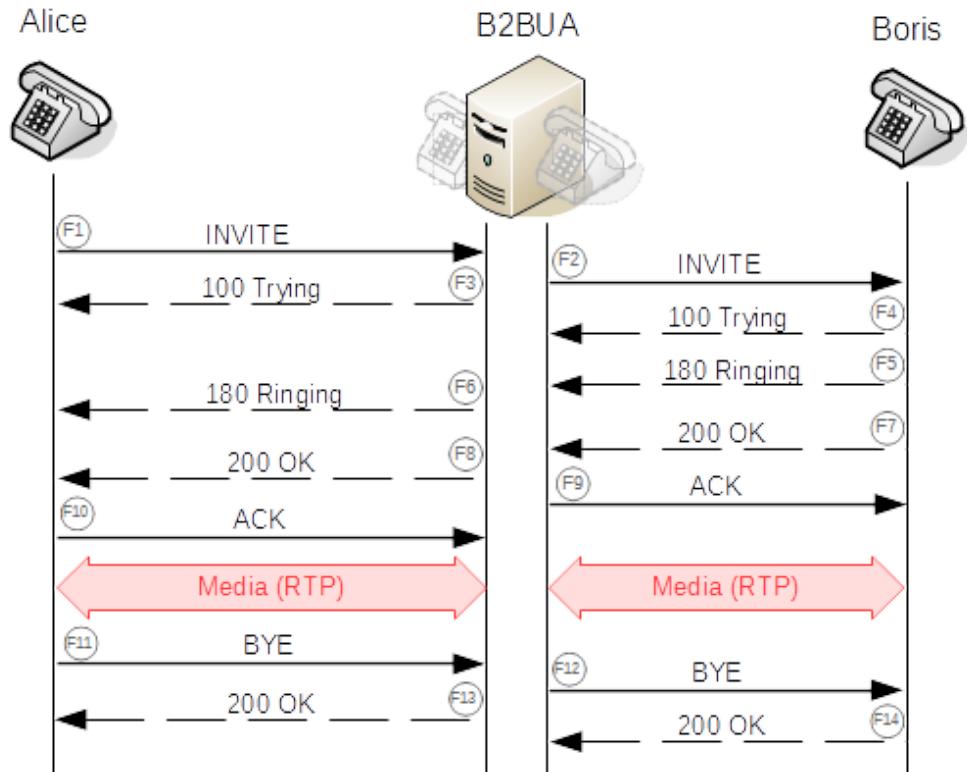
Съществуват два вида:

- **Stateless Proxy Server** - прокси сървър, който единствено препраща полученото съобщение и не съхранява никаква информация за повикване и/или SIP транзакции.
- **Stateful Proxy Server** - този тип прокси сървър проследява всяка получена заявка и отговор и може да ги използва в бъдеще, ако е необходимо. Може да препраща заявки отново, ако дълго време няма отговор от отсрецната страна.
- Registrar - регистраторът приема заявки за регистрация на потребители (UA) и ги асоциира с адрес, по който останалите да се свързват към тях. С други думи, той прави съпоставяне между AOR и "Contact" address. Регистраторите дават възможност на потребителите да актуализират информацията за своето местоположение. Предоставят и възможност за налагане на политики, които може да се използват за осигуряване на рутиране при различна от обичайната локация на потребител. Помагат на потребителите да се удостоверяват в мрежата и съхраняват URI (Uniform Resource Identifier - идентификатор за ресурс - в SIP това е адреса на потребителя) и местоположението на UA в база данни, за да послужат с тази информация на други SIP сървъри/обекти в рамките на същия домейн.
- Redirection server - за осигуряване на мобилност на потребителя се използва пренасочващ сървър. Той приема SIP заявки и връща нула или повече нови адреси, с които трябва да се осъществи връзка, за да се изпълни обаждането. Пренасочващият сървър не инициира SIP заявки и не приема SIP повиквания. Той получава заявки и търси техния желан получател в базата данни,

създадена от регистратора. Отговаря на потребителя с 3xx съобщения (отговори за пренасочване).

- Location server - сървърът за местоположение администрира информация за местоположението на повикващия към сървъра за пренасочване и SIP прокси сървъра. Той предоставя информация за възможните местоположения на клиентите на Redirection сървърите и прокси сървърите. Само тези два обекта от SIP архитектурата имат възможност да се свържат със сървър за местоположение.
- B2BUA - Back-To-Back User Agent - Потребителският агент от типа "back-to-back" е логически елемент в SIP архитектурите. Той е вид SIP UA, който получава SIP заявка, след това я преформулира и я изпраща като нова заявка. За разлика от Stateless прокси сървъра, той се интересува от състоянието на диалога и е като посредник във всички съобщения, изпратени по диалозите, които е установил. B2BUA нарушава "end-to-end" принципа на SIP, защото двамата потребители в нито един момент не препращат пакети помежду си, а си комуникират единствено с B2BUA. Двете страни на обаждането се срещат в литературата под названието "call legs". Интересна особеност е, че B2BUA действа веднъж като UAS (в първия call leg), а после като UAC (във втория), защото първо получава заявки от иницииращия обаждането, след това ги трансформира, и ги препраща от свое име към търсения потребител.

Две разделени от B2BUA страни на сесията са илюстрирани на **Фиг.1.4.**



Фиг.1.4 - SIP обаждане в сесия с B2BUA

Разликата между това обаждане и обикновено SIP обаждане между два UA е, че тук се изпраща съобщение TRYING от страната на Boris, защото той смята, че разговаря с Alice без да има нищо между тях - не знае за наличието на B2BUA, който вече е изпратил на Alice TRYING съобщение. B2BUA има пълен контрол върху сесиите. Може да се имплементират върху него множество политики. Те биха могли да му дадат права да управлява повикванията - следене и начисляване на такси, автоматично прехвърляне или прекъсване на повикванията, както и скриване на вътрешните елементи на мрежата (частни адреси, топология на мрежата и други) [6].

На Фиг.1.5 са представени ролите, които играят мрежови обекти в SIP при установяването на една сесия.



Фиг.1.5 - Роля на всеки елемент от една SIP мрежа

Потребителят с телефонния апарат иска да установи SIP комуникационна сесия с потребителя с лаптоп. При първоначално постъпване в мрежата, UA трябва да се регистрират в регистраторът, следователно първата стъпка е регистрацията на софтфона, инсталiran върху лаптопа. Регистраторът записва неговия адрес и съответно ги предоставя на сървъра за местоположение. След изпращане на INVITE съобщение от телефонния апарат към прокси сървъра, прокси сървърът прави запитване към сървъра за местоположение за адреса на търсения клиент. В случая, това е собственика на лаптопа със софтфона, и след като прокси сървърът получи адреса, той препраща INVITE съобщението в правилната посока.

1.3.5. Сигнален трафик и принцип на работа

Протоколът SIP работи на базата на двупосочна комуникация. Едното устройство в сесията изпраща заявка, а другото я получава и, връща отговор със съответен код, индикиращ дали заявката е била получени и/или обработена успешно. SIP съобщенията се състоят от няколко реда, които обясняват подробностите за повикването в текстов формат. Принципа на работа на протокола много наподобява този на HTTP или HTTPS.

В една SIP мрежа всеки обект или всяко устройство се идентифицира чрез SIP URI, известен още като "AOR" - уникален адрес [6]. Този адрес използва идентификационен низ от символи, с който се осъществява повикване към друго лице чрез SIP. Съществува и "sips" адресация, използваща TLS и TCP и съответно фокусирана към повече сигурност (sip secure). По същество AOR е SIP "телефонен номер" на потребителя и е във формат, подобен на този на електронната поща. Форматът е "sip:user@host" или понякога "sip:user@host:port". Host най-често е име на домейн или IP адрес. Когато даден SIP UA желае да комуникира с друг SIP UA, той попълва полето "To" с AOR на получателя и поставя своя собствен AOR в полето "From".

AOR е добър метод за маршрутизиране на високо ниво, но не е достатъчен, когато става въпрос за идентифициране на действителния получател на едно SIP съобщение. Това е така, защото формата "name@domain" не посочва устройството или устройства, които този AOR може да използва в момента. Един AOR би могъл да се използва за регистрация на множество устройства. Идентифицирането на конкретния получател на дадено съобщение се осъществява чрез Contact address, записан в полето "Contact". Полето за контакт позволява на потребителя да свърже едно или повече устройства с един AOR [7]. Таблиците с двата вида адреса се създават от Registrar и се използват за установяване на точния адрес, към когото е нужно да

се изпрати съобщение, посредством AOR адреса. Може AOR да се смята за public адрес от IP мрежите, а Contact - за private адрес.

A) Структура на съобщенията и видове

На *Фиг.1.6* е показана примерна SIP заявка [\[8\]](#).

```
INVITE sip:bob@biloxi.com SIP/2.0
Via: SIP/2.0/UDP pc33.atlanta.com;branch=z9hG4bK776asdhd
Max-Forwards: 70
To: Bob <sip:bob@biloxi.com>
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710@pc33.atlanta.com
CSeq: 314159 INVITE
Contact: <sip:alice@pc33.atlanta.com>
Content-Type: application/sdp
Content-Length: 142
```

Фиг.1.6 - Примерна заявка в SIP от Alice

Първият ред на кодираното в текст съобщение съдържа името на метода, ако съобщението е заявка или трицифрен код, ако съобщението е отговор. В случая, това е заявка от вида INVITE. В първия се съдържа и версията на SIP, която се използва. Следващите редове представляват списък на заглавните полета. Тази примерна заявка съдържа само необходимите (задължителните) хедър полета, без optionalните.

Полетата на хедърите от примерната заявка са следните:

- Via съдържа адреса (pc33.atlanta.com), на който Alice очаква да получи отговори на своята заявка. Полето съдържа също и "параметър на клона" (branch), който идентифицира конкретната SIP транзакция.

- To - "Към" съдържа име (в случая Bob) и SIP или SIPS URI - AOR (`sip:bob@biloxi.com`), към когото първоначално се отправя заявката.
- From - "От" - също съдържа име (Alice) и SIP или SIPS URI (`sip:alice@atlanta.com`), които този път посочват инициатора на заявката. Това поле на заглавието има и параметър "tag", съдържащ произволен низ (в случая `1928301774`), който е добавен от съответния VoIP телефон. Той се използва за идентификационни цели.
- Call-ID - Идентификатор на повикването. Съдържа уникален идентификатор за това повикване, генериран от комбинацията от случаен низ и име на хост или IP адрес на повикващия. Комбинацията от полетата To, From и Call-ID напълно определя SIP връзка от типа "peer-to-peer" между Alice и Bob и се нарича диалог.
- CSeq или Command Sequence - Поредица от команди. Съдържа цяло число и име на метод. Стойността в CSeq се увеличава за всяка нова заявка в рамките на диалога.
- Contact - полето съдържа SIP или SIPS URI, който представлява директен маршрут към контакт с Alice, обикновено съставен от потребителско име в FQDN - Fully Qualified Domain Name ("Напълно квалифицирано име на домейн"). Това представлява пълният адрес на един интернет обект. Предоставя точното му местоположение в системата DNS (Domain Name System), като посочва името на хоста, името на домейна и домейна от най-високо ниво -

Top-Level Domain (TLD) и завършва с "." [9]. FQDN е предпочитано, но много крайни системи нямат регистрирани имена на домейни, така че се допускат и IP адреси. Полето Via указва на другите елементи от сесията къде да изпратят отговор към Alice, докато полето Contact указва на други елементи къде да изпратят бъдещите си заявки към Alice. Незадължителен параметър.

- Max-Forwards поелто служи за ограничаване на броя на "скоковете" (hops), които една заявка може да направи по пътя към своята дестинация. То се състои от цяло число, което се намалява с едно на всеки скок. Един скок представлява преминаване през едно устройство. Това поле е начин за предотвратяване на зацикляне на трафика в мрежата. Незадължителен параметър.
- Content-Type съдържа описание на тялото на съобщението (самото тяло не е показано в този пример). Описанието най-често е реализирано чрез протокола SDP - Session Description Protocol и стойността на полето изглежда по следния начин: "application/sdp".
- Content-Length - Съдържа броя на октетите (байтовете) в тялото на съобщението. Незадължителен параметър.

Примерен отговор от Bob е показан на *Фиг.1.7.*

```
SIP/2.0 200 OK
Via: SIP/2.0/UDP server10.biloxi.com
      ;branch=z9hG4bKnashds8;received=192.0.2.3
Via: SIP/2.0/UDP bigbox3.site3.atlanta.com
      ;branch=z9hG4bK77ef4c2312983.1;received=192.0.2.2
Via: SIP/2.0/UDP pc33.atlanta.com
      ;branch=z9hG4bK776asdhd ;received=192.0.2.1
To: Bob <sip:bob@biloxi.com>;tag=a6c85cf
From: Alice <sip:alice@atlanta.com>;tag=1928301774
Call-ID: a84b4c76e66710@pc33.atlanta.com
CSeq: 314159 INVITE
Contact: <sip:bob@192.0.2.4>
Content-Type: application/sdp
Content-Length: 131
```

Фиг.1.7 - Примерен отговор от Bob на заявката от Alice

Първият ред в отговорите е резервиран за версията на протокола SIP и код от отговор, индикиращ как е възприета или обработена заявката и какъв е отговора на отсрецната страна.

Някои хедър полета могат да се срещнат и в своите съкратени, еднобуквени форми (*Табл.1.2*) [\[10\]](#).

Име на поле	Съкратен формат
Call-ID	I
Contact	M
Content-Length	L
Content-Type	C
From	F
Subject	S
To	T
Via	V

Табл.1.2 - Съответствие между някои полета и съкратените им форми

Съществуват общо шест вида основни методи за заявки [\[6\]](#):

- INVITE - INVITE се използва за иницииране/установяване на сесия между UA. INVITE може да съдържа информация за параметрите на сесията, предложени от повикващия, в тялото на съобщението. Сесията се счита за установена, ако на INVITE заявката е изпратен успешен отговор (код 2xx) или ACK. Успешната заявка INVITE създава диалог между два UA, който продължава до изпращането на BYE за прекратяване на сесията. INVITE, изпратен в рамките на установлен диалог, се нарича Re-INVITE. Re-INVITE се използва за промяна на характеристиките на сесията или за опресняване на състоянието на диалога.
- ACK - съобщение с ACK метод се използва за потвърждаване на отговори, из pratени след заявки (например след INVITE). ACK винаги се изпраща в същата посока като INVITE, тоест от същия UA. ACK може да съдържа параметри на сесията, ако те не са предоставени в INVITE заявката, но не може да се използва за промяна на характеристиките, ако те вече са из pratени в първоначалния INVITE.
- REGISTER - Заявката REGISTER извършва регистрация на UA. Тази заявка се изпраща от UA към сървъра на регистратора. Заявката REGISTER може да бъде препратена или проксирана, докато достигне до регистратор на посочения домейн. Тя носи AOR в полето "To" и Contact address в полето "Contact" на потребителя, който желае да се регистрира. Заявката REGISTER съдържа и информация за това колко време е валидна регистрацията (по

подразбиране 3600 секунди - 1 час). Един UA може да изпрати заявка REGISTER от името на друг. Това е известно като регистрация от трета страна - "Third-party-registration". При такъв тип регистрация полето "From" съдържа URI на страната, която изпраща регистрацията, а от името на кого се изпраща тя се идентифицира в заглавието "To".

- CANCEL - CANCEL се използва за прекратяване на сесия, която, поради някаква причина не е била установена успешно. Потребителите използват тази заявка, за да анулират "висящ" опит за установяване на SIP сесия, иницииран по-рано. CANCEL може да се изпрати както от UA, така и от прокси сървър или B2BUA.
- BYE - заявка с метод BYE, е начинът за прекратяване на установена сесия. Това е SIP заявка, която може да бъде изпратена както от едната, така и от другата страна на обаждането, за да се прекъсне сесията. BYE заявка не може да бъде изпратена от прокси сървър или от B2BUA. Заявката обикновено се насочва "от край до край", тоест заобикаля прокси сървъра, ако има такъв. BYE заявка не може да бъде изпратена към чакаща INVITE заявка или неустановена сесия.
- OPTIONS - Използва се рядко. Методът OPTIONS би могъл да се използва за запитване към друг UA или прокси сървър за техните възможности и/или за установяване на текущата му наличност. В отговора на заявката се изброяват възможностите на отсрецния UA или сървъра. Прокси сървърите никога не генерираят OPTIONS заявка.

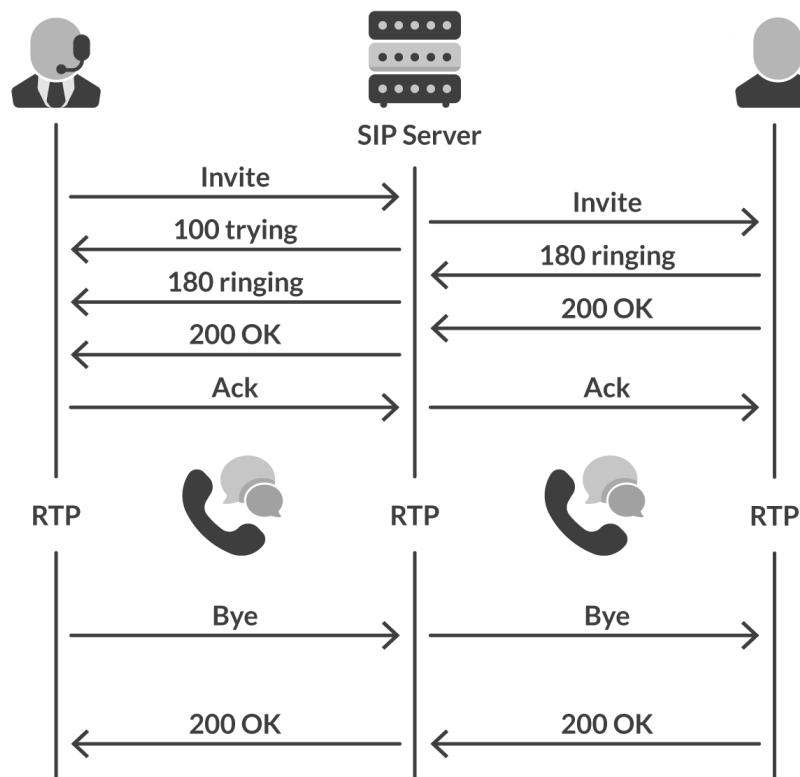
Б) Връщани кодове от отговори

Различните цифри в трицифрената последователност на върнатия код от отговора имат различно значение. Кодовете са разделени на 6 основни групи [\[11\]](#), като най често срещаните съобщения от всяка група са:

- 1xx - Информационни отговори
 - 100 - Trying - опит за свързване;
 - 180 - Ringing - UA получава Trying съобщение и известява потребителя, че се търси връзка с него (най-често се изразява в звънене).
- 2xx - Успешен отговор
 - 200 - OK - заявката е в правилен формат и е приета успешно.
- 3xx - Отговори за пренасочване
 - Най-често използвани от Redirection server за пренасочване на повикването.
- 4xx - Грешки при заявка (от клиент)
 - 400 - Bad Request - заявката не е разбрана, заради неправилен синтаксис/формат;
 - 404 - Not Found - обектът, в случаите UA, към който се търси връзка, не може да бъде намерен;
- 5xx - Грешки при сървъра
 - 503 Service Unavailable - Най-често, когато сървърът е претоварен не може да отговори или да обработи заявката на потребителя.
- 6xx - Глобални грешки
 - 603 Decline - потребител отказва или не може да осъществи обажддане.

В) Принцип на работа

Примерна сесия и съобщенията, които се изпращат, за да се установи тя, са илюстрирани на *Фиг.1.8*. За обмяната на служебна SIP информация е прието е названието "SIP Transaction" [6].



Фиг.1.8 - Примерен цикъл на една SIP сесия - "SIP Transaction"

За да се даде начало на изграждане на сесия се изпраща заявка от вида INVITE до SIP прокси сървър. Прокси сървърът изпраща отговор 100 Trying на повикващия (отляво в случая), за да не се налага повторното изпращане на заявки от тип INVITE. Прокси сървърът търси адреса на отсрецната страна в сървъра за местоположение и, след като го получи, препраща INVITE заявката по-нататък - към нейния адресат. При получаване на заявка с метод INVITE в нея, от дясната страна се генерира 180 Ringing отговор, заедно с което започва телефона да звъни. Този отговор се връща обратно на повикващия. В този момент от

транзакцията се изчаква отговор от дясната страна, тоест се изчаква вдигане на телефона, а когато това се случи съответно се генерира отговор 200 OK. Лявата страна, получила 200 OK съобщението, изпраща ACK и оттук нататък сесията е установена. RTP пакети започват да се разменят и това въщност представлява разговора между двамата потребители. След приключване на разговора всеки участник може да изпрати заявка BYE, за да прекрати сесията. Последното съобщение след BYE, ако всичко е минало успешно, е отговор 200 OK.

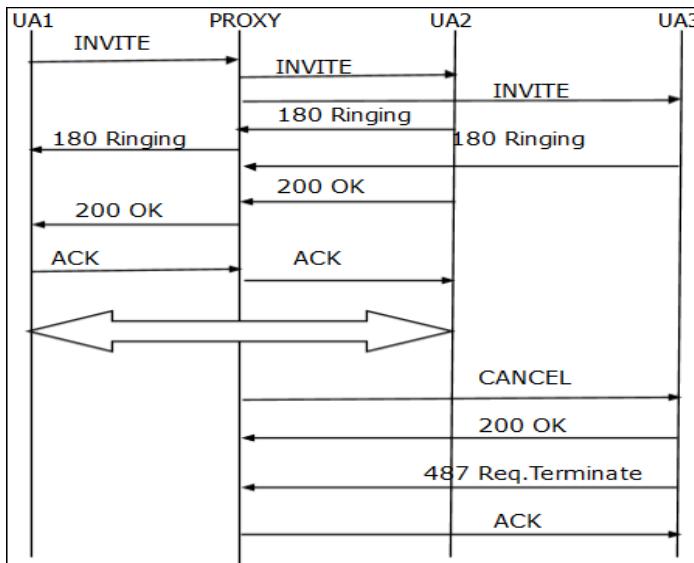
1.3.6. Разклоняване в Session Initiation Protocol

Понякога прокси сървърите препращат едно SIP повикване към няколко крайни точки. Този процес е известен като разклоняване (на английски - "forking"). По този начин едно повикване може да се свърже с много крайни точки едновременно. Разклоняването е възможно само ако има наличен Stateful proxy сървър, тъй като той трябва да отговори на много получени съобщения от различни крайни точки. Съществуват два вида разклоняване - паралелно и последователно [\[6\]](#).

A) Паралелно разклоняване

При този сценарий прокси сървърът разклонява INVITE към две устройства (UA2, UA3) едновременно. И двете устройства генерират 180 Ringing (звънене), а ако откликнат на повикването, генерират съответно 200 OK отговор. Отговорът, който пръв достигне до инициатора (в случая по-бързият отговор е този на UA2), ще установи сесия с него. За другия UA ще се изпрати заявка от типа CANCEL, за да се прекрати изграждането на сесия с него, на която заявка се отговаря с 200 OK и 487 "Request Terminate". Това разклоняване се нарича паралелно, защото

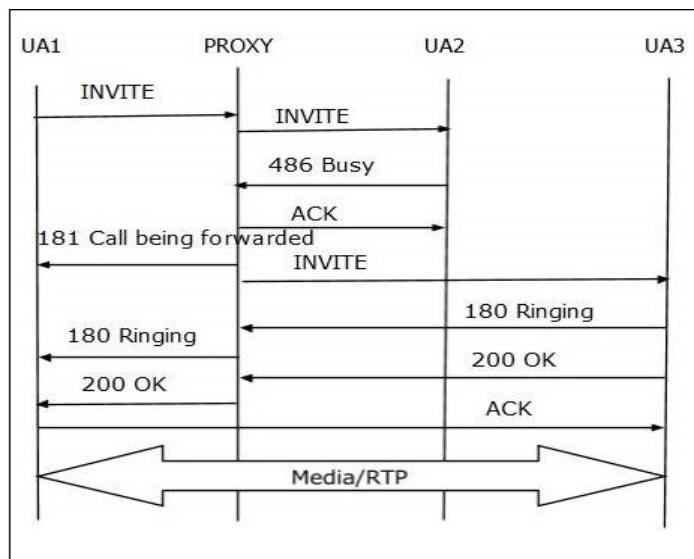
успоредно и едновременно се препраща INVITE заявката на няколко отдалечени UA и е показано на Фиг.1.9.



Фиг.1.9 - Паралелно разклоняване на едно SIP обажддане

Б) Последователно разклоняване

При последователното разклоняване сървърът изпраща INVITE до едно устройство (UA2). Ако това устройство е недостъпно или заето по това време, сървърът го препраща към друго устройство (UA3) и така нататък. Именно поради този тип разклоняване се нарича последователно (Фиг.1.10) [6].



Фиг.1.10 - Последователно разклоняване на едно SIP обажддане

1.4. Session Description Protocol

1.4.1. Основни сведения

Докато SIP най-общо се занимава с установяването и прекратяването на сесии, SDP - Session Description Protocol (Протокол за описание на сесии) се занимава единствено с описанието на трафика в рамките на тези сесии [12]. Когато се инициират гласови повиквания по IP, поточно видеопредаване или всяка друг вид сесии, съществува изискване за изясняване на информация за сесията - какъв трафик би се пренасял (има се предвид аудио, видео), по какъв начин, както и много други данни за описание на сесията. SDP осигурява стандартно представяне на такава информация, независимо от начина на нейното пренасяне.

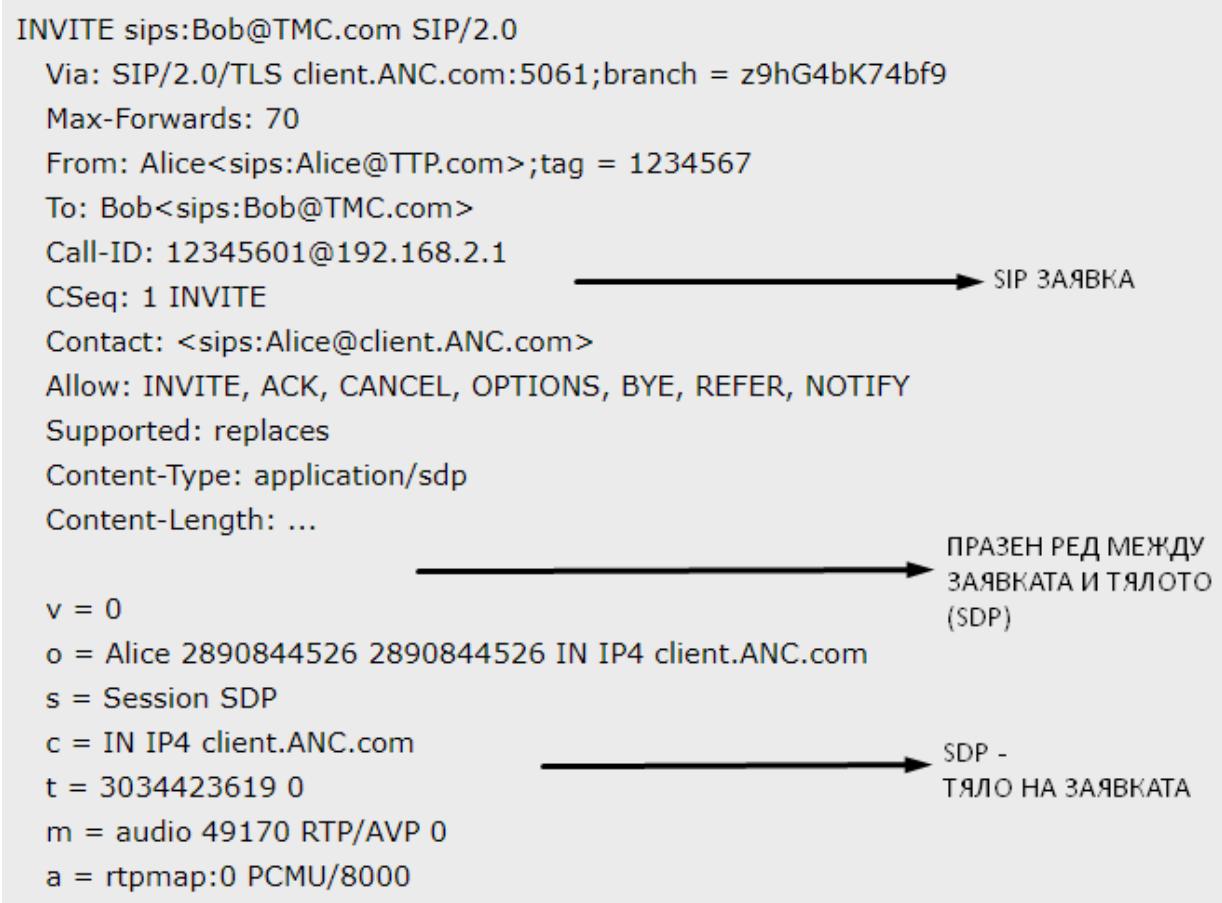
SDP е единствено формат за описание на сесии - той не включва транспортен протокол и съответно поддържа използването на различни транспортни протоколи. Фактът, че SIP пренася описанието на сесиите в друг протокол, не е случаен. Създателите на SIP са си поставили за цел да го направят "media agnostic" - независим от трафика, за който се инициира сесията. SDP е проектиран така, че да може да се разширява, за да поддържа нови типове и формати на носители. SDP намира приложения във връзка с протокола за пренос в реално време RTP (Real-time Transfer Protocol), протокола за стрийминг в реално време RTSP (Real-Time Streaming Protocol), SIP и дори като самостоятелен формат за описание на сесии.

Важно е да се разбере, че SDP не договаря параметрите за сесиите. Вместо това, едната страна от сесията казва на другата какви параметри за сесията поддържа и съответно предлага да се изберат и използват някои от тях. В зависимост от тази информация получателите на описанието вземат решение за участие в сесията - дали, кога и как

да вземат участие. SDP има общо предназначение, така че да може да се използва в широк спектър от мрежови среди и приложения [13].

1.4.2. Структура на съобщенията

Примерна SIP заявка с SDP тяло, както и означения за информацията, която се пренася в заявката, е показана на [Фиг.1.11 \[14\]](#).



Фиг.1.11 - Примерна SIP заявка, включваща в себе си SDP тяло

SDP частта на едно SIP съобщение (наричана често тяло на съобщението) се състои от поредица от редове (наречени полета), съдържащи "<символ>=<стойност>", където <символ> е единичен буквен знак, указващ името на даден параметър (с чувствителност към малките и големите букви - case sensitive, тоест не се толерират главни

букви), а <стойност> е структуриран текст, описващ стойността на съответния параметър.

1.4.3. Полета в Session Description Protocol header

Съобщението се състои от три основни раздела - описания на сесията, на времето и на носителя на трафика. Всяко съобщение може да съдържа множество описания на времето и трафика, но само едно описание на сесията [13]. Независимо дали незадължителни полета се използват или не, редът на всяко едно поле е строго определен. Параметрите (подредени) са показани по-долу, като незадължителните са обозначени с "*":

- Описание на сесията - описание на ниво "session"
 - v= (версия на протокола);
 - o= (идентификатор на инициатора и сесията);
 - s= (име на сесията);
 - i=*(информация за сесията);
 - u=*(URI на описанието);
 - e=*(имейл адрес);
 - p=*(телефонен номер);
 - c=*(информация за връзката - не се изисква, ако е включена във всички описания на трафика);
 - b=*(нула или повече редове с информация за ширината на честотната лента);
 - a=*(нула или повече редове с атрибути на сесията).
- Описание на времето (едно или повече)
 - t= (време, през което сесията е активна);
 - r=*(нула или повече повторения);
 - z=*(ред за отместване заради часова зона).

- Описание на носителя на трафика (нула или повече) - описание на ниво "media"
 - m= (информация за конкретен тип/вид трафик);
 - i=* (заглавие на носителя);
 - c=* (информация за връзката - по избор, ако е включена на ниво session);
 - b=* (нула или повече редове за информация за честотната лента);
 - a=* (нула или повече редове за атрибути на носителя).

Между буквата на съответния параметър и знака "=" няма интервали, а между всяка отделна стойност има точно един интервал. Всяко поле има определен брой дефинирани стойности. Пояснения към всяко поле са показани по-долу [\[13\]](#):

- Версия на протокола - Protocol version ("v=")
Редът "v=" дава версията на SDP, в момента е "0".
- Произход - Origin ("o=")
Полето "o=" (произход - "origin") описва инициатора на сесията (потребителско име и адреса на хоста на потребителя), заедно с идентификатор на сесията и номер на версията. Редът служи като уникален идентификатор за тази версия на описанието на сесията (с версията се следят модификации по сесията).
- Име на сесията - Session name ("s=")
Полето за име на сесията е текстовото име на сесията. В описанието на сесията трябва да има само един ред "s=" и той не трябва да бъде празен.

- Информация за сесията - Session information ("i=")

Това поле предоставя текстова информация за сесията, което е предназначено за четене от хора.

- URI ("u=")

Редът "u=" предоставя URI. В случая URI трябва да бъде указател към допълнителна информация за сесията, често е уеб страница. Този ред не е задължителен. Не се допуска повече от един ред "u=" за описание на сесия.

- Имейл адрес - Email address ("e=") и Телефонен номер - Phone number ("p=")

В тези редове се посочва информация за контакт. Включването на тези полета по желание.

- Информация за връзка - Connection information ("c=")

Редът "c=" съдържа информация, необходима за установяване на мрежова връзка, с други думи - IP адрес.

- Информация за честотната лента - Bandwidth information ("b=")

Полето за широчина на честотната лента е незадължително и обозначава предложената широчина на честотната лента, която да бъде използвана от сесията.

- Активно време - Active time ("t=") и Време на повторение - Repeat time ("r=")

Редът "t=" представлява описание на времето за начало и край на сесията. Редът "r=" е за определяне на времена за повторение на сесията. По подразбиране всички дефиниции на времена са в секунди (отправната точка е 1.01.1900 г. - UTC - Coordinated Universal Time).

- Настройване на часовата зона - Time Zone Adjustment ("z=")

Редът "z=" е незадължителен модификатор на полетата за повторение, които следват непосредствено след него. Представлява изражение на отместването във времева зона. Съдържа поредица от отправни точки и отмествания като стойности.

- Описания на носителя на трафика - Media descriptions ("m=")

m=<media> <port>/<number of ports> <proto> <fmt> ...

Описанието на сесията може да съдържа няколко описания на носители на трафик. Всяко такова описание започва с ред "m=" и завършва или със следващия ред "m=", или с края на описанието на сесията. Полето има няколко подполета:

<media> - Типът на трафика, например "audio".

<port> - Порта на транспортното ниво, към който се изпраща потока от данни.

<proto> - Транспортният протокол (или негов профил), отговорен за пренос на данните. Дефинирани са следните стойности (могат да бъдат разширени от IANA - Internet Assigned Numbers Authority):

- "udp": Данните се пренасят директно чрез UDP без нищо допълнително.
- "RTP/AVP": Означава RTP, използван в рамките на профила RTP/AVP за аудио и видеоконференции с минимален контрол, работи заедно с UDP.
- "RTP/SAVP": Този профил означава защищен (S - secure) RTP, също работи заедно с UDP.
- "RTP/SAVPF": Означава RTP с разширен SAVP профил за обратна връзка, базирана на RTCP.

<fmt>

Описание на формат на трафика. Интерпретацията на тези стойности зависи от стойността на подполето <proto>. Ако подполето <proto> е "RTP/AVP" или "RTP/SAVP", подполетата <fmt> съдържат номера на типа на RTP полезния товар (RTP payload). Когато е даден такъв списък, всички формати могат да се използват в сесията. Те са изброени по реда на предпочтение. Ако <proto> е със стойност "udp", информацията в <fmt> трябва да е за типа трафик - например "audio".

Примерно описание на сесия чрез SDP, включващо повечето полета, е илюстрирано на *Фиг.1.12* [13].

```
v=0
o=jdoe 3724394400 3724394405 IN IP4 198.51.100.1
s=Call to John Smith
i=SDP Offer #1
u=http://www.jdoe.example.com/home.html
e=Jane Doe <jane@jdoe.example.com>
p=+1 617 555-6011
c=IN IP4 198.51.100.1
t=0 0
m=audio 49170 RTP/AVP 0
m=audio 49180 RTP/AVP 0
m=video 51372 RTP/AVP 99
c=IN IP6 2001:db8::2
```

Фиг.1.12 - Описание на сесия чрез SDP

1.4.4. Атрибути в Session Description Protocol header

Незадължителното поле "a=" съдържа атрибути на предишното описание (най-често на ниво media). Това поле може да се използва за разширяване на SDP, за да се предостави повече информация за трафика и за сесията.

Атрибутите в SDP могат да бъдат:

- На ниво сесия - "session". Ниво на сесия означава, че атрибутът е посочен преди първия ред "m=" в SDP. В този случай атрибутът се прилага за всички описания на носители.
- На ниво носител - "media". Това ниво означава, че атрибутът е посочен след някой ред "m=". В този случай атрибутът се прилага само за това конкретно описание на носител.

SDP може да включва атрибути както на ниво сесия, така и на ниво носител. Ако един и същ атрибут се появи и в двете нива, този на ниво носител има предимство пред атрибута на ниво сесия за това конкретно описание на носител. Атрибутните полета могат да бъдат в две форми:

- Атрибут със свойство е с формата "a=<име на атрибут>". Това са двоични атрибути и наличието на такива атрибути означава, че има такова свойство на сесията. Пример за това може да бъде "a=recvonly".
- Атрибут със стойност е такъв, който има формата "a=<име на атрибут>:<стойност на атрибут>". Например, ако трябва да се описва бяла дъска, тя би имала атрибут за стойност със следното изражение "a=orient:landscape".

Най-често срещаният от атрибутите е "rtpmap". Той се използва на ниво "media" и съпоставя номер на тип RTP payload (използван в реда "m=") към име на кодек. Примерен ред с атрибут "rtpmap" може да бъде: "m=audio 49232 RTP/AVP 98 a=rtpmap:98 OPUS". Този ред указва, че RTP/AVP аудио трафик с номер 98, пренасян на порт 49232, ще използва кодек OPUS за кодиране и декодиране.

1.5. Real-time Transport Protocol

1.5.1. Основни сведения

Протоколът за пренос в реално време (Real-time Transport Protocol, RTP) е протокол за пренос на аудио и видео по интернет [15]. Той е проектиран така, че да осигурява функции за мрежов транспорт от край до край ("end-to-end"), подходящи най-вече за предаване на данни в реално време, като например аудио и видео.

RTP се използва заедно с протокола за контрол на преноса в реално време (Real-time Transport Control Protocol - RTCP), който е направен за наблюдение на качеството на предаване на данни. RTP осигурява доставката на пакетите, докато RTCP се използва за осигуряване на обратна връзка за качеството на предаването и за предоставяне на информация за контрол. RTP е протокол, базиран на пакети (packet-based protocol), което означава, че той разделя потоците от данни на пакети за предаване по мрежата. На всеки пакет се дава пореден номер (sequence number), който позволява на приемника да "сглоби" пакетите в правилния ред. RTP включва и "Timestamp", който позволява на приемника да синхронизира аудио и видеопотоците. RTP се използва широко в различни приложения, включително за VoIP, видеоконференции и предаване на трафик поточно. Поддържа се от много сървъри и често се използва в комбинация с други протоколи, като RTSP, SIP и SDP, за предоставяне на аудио и видеосъдържание през интернет.

RTP не включва никакви вградени мерки за сигурност. Затова той може да се използва заедно с други протоколи, като например Secure Real-time Transport Protocol (SRTP), за да се осигури криптиране и удостоверяване на трафика.

1.5.2. Системи в Real-time Transport Protocol

В една RTP мрежа съществуват най-общо три вида системи според тяхната функция [\[16\]](#).

A) End system - крайна система

Системата, която генерира някакво съдържание/трафик за изпращане в RTP пакети и/или консумира съдържанието на получени RTP пакети. Една крайна система може да действа като един или повече синхронизирани източници в определена RTP сесия, но обикновено действа като само един.

Б) Mixer

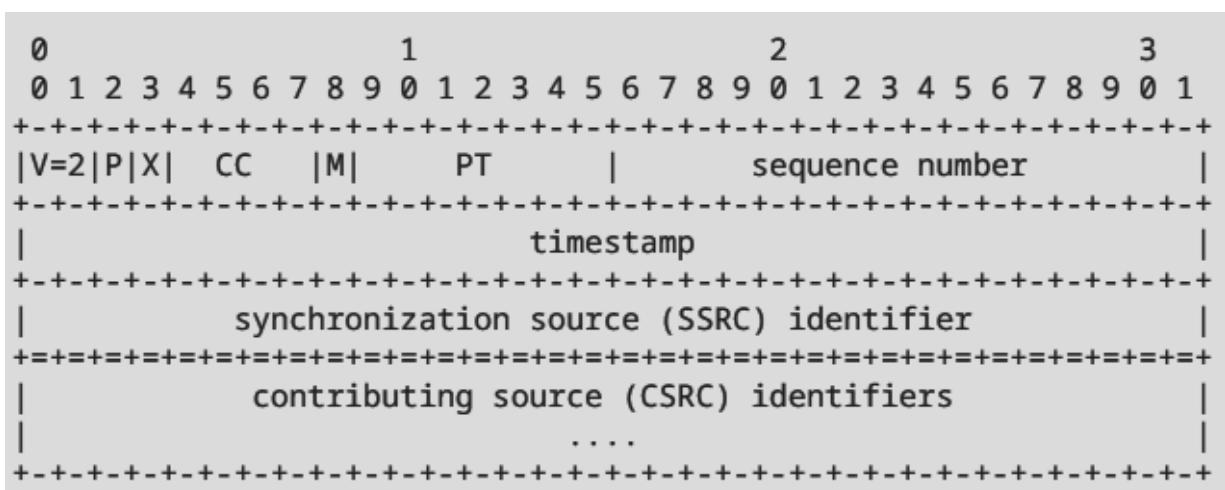
Междинна система, която получава RTP пакети от един или повече източници, евентуално променя формата на данните, комбинира ги и след това препраща нов RTP пакет. Тъй като времето между няколко входни източника обикновено не е синхронизирано, миксерът прави корекции на времето и генерира собствено време за комбинирания поток. По този начин всички пакети данни, произходящи от миксер, ще бъдат идентифицирани с него като техен източник на синхронизация. Те най-често се използват в случаи, при които участниците в една област са свързани чрез нискоскоростна връзка с участници, които имат достъп до високоскоростна мрежа. В такава ситуация миксер се поставя в близост до нискоскоростната област. Той ресинхронизира входящите аудиопакети, реконструира ги до постоянното времетраене на всеки пакет от 20 ms и "смесва" тези реконструирани аудиопотоци в един поток. В RTP хедъра са предвидени средства за идентификация на миксерите, които са допринесли за преправения пакет, така че да се осигури правилна индикация на говорещия при получателите.

В) Транслатор

Междинна система, която препраща RTP пакети, без да променя идентификатора на източника на синхронизация, тоест без да сменя синхронизацията на пакетите. Някои от участниците в даден разговор могат да бъдат свързани с високоскоростни връзки, но да не могат да бъдат достигнати чрез IP multicast пакет, защото се намират зад защитна стена на приложния слой (application-level firewall), която не пропуска този тип трафик, например. За подобни случаи може да се използва друг тип система на ниво RTP - транслатор. Два транслатора се инсталират, по един от двете страни на защитната стена, като по този начин външният насочва всички multicast пакети, получени от вътрешната мрежа, към транслатора от другата страна на защитната стена. Пакетите биха се върнали по обратния път по аналогичен начин.

1.5.3. RTP Хедър

Форматът на един RTP хедър е показан на Фиг.1.13 [16].



Фиг.1.13 - RTP хедър

Първите дванадесет октета присъстват във всеки един RTP пакет, а списъкът със CSRC (Contributing Source) идентификатори присъства само когато е вмъкнат от миксер.

Полетата в хедъра на RTP са следните:

- Версия - Version (V): 2 бита.

Това поле идентифицира версията на RTP. Версията, определена от RFC 3550 е "2". Стойността "1" се използва от първата версия на RTP.

- "Подложка" - Padding (P): 1 бит.

Ако този бит е зададен (стойността му е единица), пакетът съдържа един или повече допълнителни октети с padding в края, които не са част от полезния товар в пакета. Използването на padding може да се наложи при някои алгоритми за криптиране с фиксириани размери на блоковете или за пренасяне на няколко RTP пакета в данните на протокол от по-ниско ниво.

- "Разширение" - Extension (X): 1 бит.

Предвиден е механизъм за разширяване на хедъра, който позволява на отделните реализации да експериментират с нови, независими от формата на полезния товар функции. Ако битът е зададен, тогава фиксираният хедър трябва да бъде последвано от разширение с определен формат.

- Брой на CSRC - CSRC Count (CC): 4 бита.

Полето съдържа броя на CSRC полета, които следват в хедъра (ако има такива). Максималният им брой е 15, затова и полето CC е 4 бита.

- Маркер - Marker (M): 1 бит.

Интерпретацията на маркера се определя от така наречения "профил", което представлява начин за преработка/модификация на RTP хедъра, така че той да отговаря на спецификите на определени приложения. Примери за профили са "RTP/AVP" - RTP Audio/Video Profile. Маркирите биха могли да служат, за да показват важни събития, като например да маркират границите на всеки един от пакетите в даден поток.

- Тип на полезния товар - Payload Type (PT): 7 бита.

Това поле идентифицира чрез код формата на полезния товар на RTP и определя интерпретацията му от приложението. Един RTP източник може да промени типа полезен товар по време на сесията. Приемаща страна на един пакет, съдържащ типове полезен товар, които тя не разбира, игнорира пакета.

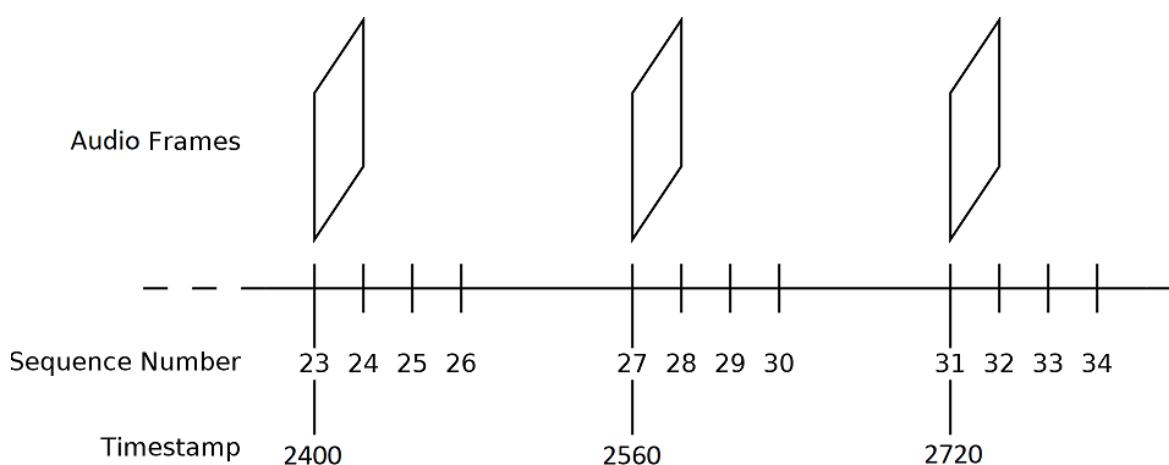
- Номер на последователността - Sequence number: 16 бита.

Номерът на последователността се увеличава с единица за всеки пакет с данни от RTP, който бъде изпратен, и може да се използва от приемника за откриване на загуба на пакети и за възстановяване на последователността на пакетите. Първоначалната стойност на номера на последователността трябва да бъде случайна (непредсказуема) от гледна точка сигурност - по този начин атаки към мрежата са по-трудни.

- Timestamp: 32 бита.

Използва се от приемника за възпроизвеждане на получените преби в подходящо време и интервал. За аудио трафик се вземат преби от данните веднъж на всеки 125 µs (тоест с 8 kHz честота на дискретизация/семплиране). Timestamp отразява момента на

вземане на преби от първия октет в RTP пакета, защото този момент е известен на предаващата крайна точка и има обща дефиниция за всички типове трафик, независимо от кодиране, закъснения или други преработки. Моментът трябва да бъде получен от часовник, който се увеличава монотонно във времето. Първоначалната стойност на Timestamp трябва да бъде произволна. Timestamp полета на RTP пакети от различни потоци данни могат да напредват във времето с различни скорости и обикновено имат независими, случайни отмествания (offsets). За всеки поток timestamp на RTP е свързан с момента на вземане на преби, както и с време от референтен часовник (среща се под названието "стенен часовник"), който представлява времето, в което от данните, отговарящи на даден RTP timestamp, е била взета преба. Референтният "стенен" часовник е общ за всички типове трафик, които трябва да бъдат синхронизирани. Един кадър (пакет в RTP) съответства на 20 ms от дадено съдържание – независимо дали то е видео или аудио. За 1 секунда има 1000/20, тоест 50 кадъра, следователно при 8kHz честота на семплиране инкременталната стойност на timestamp за аудио ще бъде 8000/50, което е равно на 160 (Фиг.1.14).



Фиг.1.14 - Timestamp в аудио пакети

- Източник на синхронизация - Synchronization Source (SSRC): 32 бита.

Полето SSRC идентифицира източника на синхронизация. Той трябва да бъде избран на случаен принцип, като целта е да няма два еднакви SSRC идентификатора в рамките на една и съща RTP сесия. Ако даден обект промени своя транспортен адрес, той трябва да избере и нов SSRC идентификатор, за да не бъде интерпретиран като зациклил източник. Транспортните адреси представляват комбинацията от мрежов адрес и порт, която идентифицира крайна точка на транспортния слой, като най-често комбинацията е от IP адрес и TCP или UDP порт. Пакетите се предават от транспортен адрес на източника (source transport address) до транспортен адрес на получателя (destination transport address).

- Списък на CSRC - CSRC List: от 0 до 15 елемента, 32 бита всеки. Списъкът CSRC идентифицира източниците, които по някакъв допринасят за полезния товар, съдържащ се в този пакет. Броят на идентификаторите се определя от полето CC. Идентификаторите на CSRC се въвеждат от миксери, като използват собствените си SSRC идентификаторите и ги попълват в един елемент от списъка. За аудиопакети се изброяват SSRC на всички източници, които са били смесени заедно, за да се създаде новия сигнал, което позволява правилното възпроизвеждане на говорещия при приемника.

1.5.4. Real-time Transport Control Protocol

Протоколът за контрол на RTP (Real-time Transport Control Protocol - RTCP) се основава на периодичното предаване на контролни пакети до всички участници в сесията. Пакетите се разпространяват заедно с тези от RTP, което се осигурява чрез мултиплексиране на пакетите с данни (от RTP) и контролните пакети (от RTCP). Най-често това се реализира чрез използване на отделни номера на портове при UDP за RTP и RTCP.

RTCP изпълнява три основни функции [\[16\]](#):

- Предоставяне на обратна връзка за качеството на разпространението на данни от RTP. Като транспортен протокол, изпращането на данни е неразделна част от RTP, а контролът върху това е свързан с функциите за контрол на други протоколи от транспортния слой.
- RTCP пренася постоянен идентификатор на транспортно ниво за RTP източник, наречен канонично име или Canonical name - "CNAME". Тъй като идентификаторът на SSRC може да се промени, приемниците се нуждаят от CNAME, за да следят всеки участник. Приемниците могат също така да изискват CNAME, за да асоциират свързани RTP сесии към даден участник, например за синхронизиране на аудио и видео. Синхронизацията изиска NTP (Network Time Protocol) и RTP timestamp да бъдат включени в RTCP пакетите.
- Пъrvите две функции изискват всички участници да изпращат RTCP, следователно скоростта трябва да се контролира, за да може RTP да има възможност да се мащабира до голям брой участници. Поради факта, че всеки участник изпраща своите

контролни пакети до всички останали, всеки може да наблюдава броя на участниците. Следователно RTCP се използва и за изчисляване на скоростта, с която да се изпращат пакетите.

Типовете RTCP пакети за пренос на контролна информация са следните:

- SR - Sender Report: Отчет от изпращачи - статистически данни (например загубени пакети, информация за синхронизация на пакети, информация за накъсване на потока от данни) за изпращане и получаване от участници, които са активни изпращачи;
- RR - Receiver Report: Отчет от получатели - статистически данни за получаване от участници които не са активни изпращачи, а са получатели;
- SDES - Source Description: Елементи за описание на източника, включително CNAME;
- BYE: Посочва края на участието на даден участник;
- APP: Специфични за приложението функции.

Всеки RTCP пакет започва с фиксирана част, подобна на тази при RTP пакетите. Следват структурирани елементи, които могат да бъдат с различна дължина в зависимост от типа на пакета. Изискването за фиксираната част на всеки пакет е включено, за да могат RTCP пакетите да се надграждат - да бъдат "stackable". Няколко RTCP пакета могат да бъдат обединени в "съставен пакет" - compound packet.

С изключение на BYE, изпращането на всеки друг тип от гореспоменатите типове пакети добавят стойността на SSRC полето на техния изпращач в таблица от участници (member table). В

последствие, при използването на пакет, съдържащ BYE съобщение, съответния участник се премахва от таблицата с участници.

1.5.5. Кодеки - основни сведения

Кодеците са от изключителна важност за функционирането на VoIP като цяло, по-конкретно RTP. "Кодек" идва от английската дума "codec" - комбинация между "code" и "decode" - кодиране и декодиране. Те биват хардуерни - устройства и чипове или софтуерни - програми и апликации [17]. Тяхната цел, независимо от реализацията, е да кодират и съответно да декодират сигнали. Кодирането на информация само по себе си намалява и нейния обем, тоест я компресира, под никаква форма. Съществуват различни по предназначение видове кодеки - разделят се най-често на две категории в това отношение - видео и аудио кодеки. На практика, за да се предаде върху IP, гласовия трафик се кодира и съответно компресира (намалява се обема на аналогия, човешки глас), а за да се възпроизведе обратно като такъв - се декомпресира.

Аудио кодеците се различават по функция - някои приоритизират ниска латенция (забавяне), други използват малко ресурси на мрежата, трети поддържат много различни звукови честоти. Доставчиците на VoIP услуги обикновено поддържат няколко стандарти за кодеки. Големината на честотната лента и пробите за секунда в кодеците са правопропорционални - колкото по-голяма е честотната лента на съответния кодек, толкова повече преби за секунда се вземат от сигнала и обратно. Пробите за секунда показват колко преби се вземат от аналогият сигнал (гласа) за една секунда - повече означава по-добро качество на звука.

Кодеците са неизменна част от VoIP архитектурата и са от ключово значение за разговорите през интернет, защото отговарят както за качеството на звука, така и за бързината на предаване на данните.

Съществуват огромен брой аудио кодеци, но тези, които VoIP архитектурите най-вече използват, са пет (Табл.1.3).

Наименование	Честотна лента	Проби за секунда	Bitrate	Латенция/ забавяне	Предимства
G.711	300-3400 Hz	8000	64 Kbps	125 µs	Ниско забавяне
G.722	50-7000 Hz	16000	32 Kbps	4 ms	Добро качество, ниско забавяне
G.722.2	50-7000 Hz	16000	48-64 Kbps	25 ms	Компромис между честотна лента, bitrate и забавяне
G.729	300-3400 Hz	8000	8 Kbps	15 ms	Малка консумация на енергия
Opus	50-20000 Hz	48000	6-510 Kbps	26,5 ms	Голяма честотна лента - звуци освен човешки глас

Табл.1.3 - Петте основни кодека, които VoIP решенията използват

Различните видове честотни ленти са Narrowband - 300 до 3400 Hz - честотата на човешкия глас, Wideband - 50 до 7000 Hz - честотата на "HD" (High Definition) глас и Ultra-wideband - 50 - 20000 Hz. Диапазона на чuvане на човешкото ухо е от 20 Hz до 20 kHz, тоест при Opus могат да се предават звуци с честоти, които човек не би могъл да произведе с гласа си.

Глава 2. Основни характеристики на Session Border Controller

2.1. Функционални изисквания към дипломната работа

2.1.1. Основни изисквания

- Да се опишат основните протоколи, устройства и понятия, свързани с VoIP мрежите;
- Да се опишат основните характеристики на Oracle SBC;
- Да се опише механизъм за манипулиране на SIP header (SIP заглавия) с цел скриване на топологията на мрежата на оператора.

2.1.2. Изисквания към разработеното решение

Разработеното технологично решение трябва да:

- Установява връзката между VoIP оператори (доставчици на гласови услуги);
- Има функционалност за маршрутизация на обаждания чрез външен ENUM сървър;
- Има функционалност за маршрутизация на обаждания чрез локални политики (LP);
- Включва резервираност на устройствата;
- Реализира механизъм за манипулиране на SIP header (SIP заглавия) с цел скриване на топологията на мрежата на оператора;
- Бъде тествано, за да се докаже работоспособността на изградената мрежа, както и всички заложени функционалности.

2.2. Основни сведения за Session Border Controller

SBC (Session Border Controller - "Границен контролер на сесии") е устройство със специално предназначение, което защитава и регулира IP комуникационни потоци [\[18\]](#). Както подсказва името, тези устройства се разполагат на границите на мрежата, за да контролират IP сесии и да налагат политики върху тях. Първоначално замислени за защита и контрол на VoIP мрежи (именно това е и тяхната функция в настоящата дипломна работа), SBC вече се използват и за регулиране на всички форми на комуникации в реално време, включително VoIP, IP видео, текстови чатове и конферентни срещи. SBC устройство на производителя Eltex е показано на *Фиг.2.1.*



Фиг.2.1 - SBC-3000 - Session Border Controller на производителя Eltex

SBC устройствата защитава основната SIP мрежа и сървърите в рамките на SIP и осигуряват взаимодействие между клиент и сървър (UAC и UAS), като изпълняват ролята на B2BUA [\[19\]](#). Чрез вземане на абсолютен контрол върху сесията, тези устройства действат едновременно като сървър (UAS) и клиент (UAC) за всяко сигнално съобщение на всеки call leg на повикването. По този начин SBC може

да контролира детайлно комуникационната сесия и да налага политики и правила.

SBC устройства се използват както от доставчици на комуникационни услуги, така и от предприятия [18]. Доставчиците на услуги поставят SBC на границите на мрежите за достъп (access), опорните мрежи (backbone) и мрежите за взаимно свързване (peer). Предприятията обикновено разполагат SBC на границата на корпоративната мрежа, например като крайна точка за услугата SIP trunking (получаване и отправяне на обаждания в интернет, извън локалната мрежа). Много производители предлагат различни продуктови фамилии SBC за доставчици на услуги и SBC за предприятия, поради различните изисквания за функционалност и мащабируемост. Корпоративните SBC обикновено се наричат E-SBC или e-SBC (Enterprise SBC). Някои производители на SBC предлагат софтуерно базирани SBC, които работят на виртуализирани стандартни сървъри, за да позволят виртуализация на мрежовите функции. Подобни реализации се наричат e-vSBC или E-VSBC - Enterprise Virtual SBC.

2.3. Основни функционалности и приложения на Session Border Controller

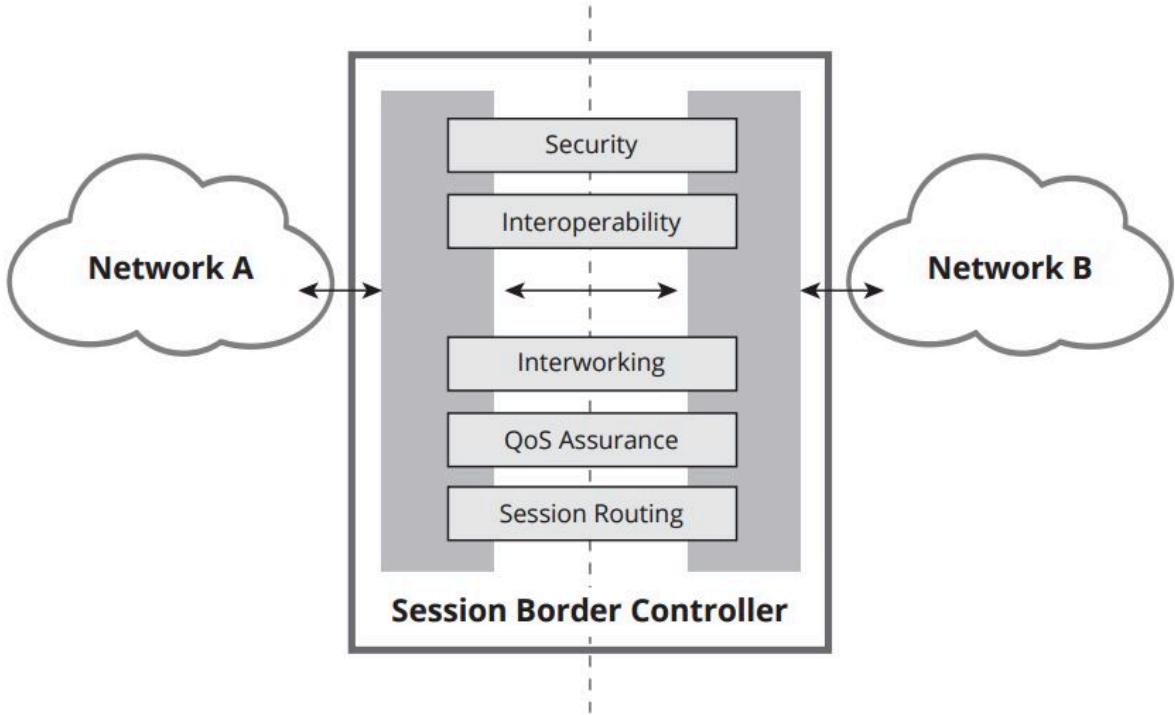
2.3.1. Основни функционалности на Session Border Controller

SBC устройствата предоставят множество функционалности, което ги прави гъвкави и предопределя все по-честото им използване им за много различни видове комуникации в наши дни.

Най-важните функции на SBC са следните [18]:

- Security (Сигурност) - SBC защитават мрежата от атаки от типа на DoS (Denial of Service) и DDoS (Distributed Denial of Service) и осигуряват криптиране на трафика и сигналния трафик. По този начин гарантират поверителност и защита от кражба на самоличност или друг вид интернет измами;
- Interoperability (Оперативна съвместимост с множество доставчици на устройства) - SBC стандартизират заглавията и съобщенията на сигналния поток от SIP, за да намалят несъвместимостта между различни доставчици на устройства;
- Interworking (Взаимодействие и транслация/транскодиране между протоколи и кодеци) - SBC позволяват взаимодействие и съответно транслация между различни протоколи (например SIP към H.323) или различни кодеци (например "транскодиране" от G.711 към G.729);
- Quality of Service (QoS) - SBC налагат политики за контрол на достъпа до повикване - най-често ограничаване на скоростта и контрол над ресурсите за осигуряване на качеството на услугата;
- Session routing (Маршрутизиране на сесии) - SBC маршрутизират сесиите през мрежови интерфейси, за да осигурят висока наличност за трафика и маршрутизиране по най-ниска цена за трафик - Least Cost Routing (LCR).

Осигуряването на съответните функции от SBC между две точки е илюстрирано на Фиг.2.2.



Фиг.2.2 - Позиция и функции на едно SBC устройство в мрежа

2.3.2. Основни приложения на Session Border Controller

Приложенията на тези устройства могат да варират, но често срещаните са описани по-долу [18][19]:

- SIP trunking - E-SBC обикновено се инсталират на границата на корпоративната мрежа за възползване от услугата SIP trunking. Някои доставчици на услуги за SIP trunking включват базираните в помещенията на клиента E-SBC в пакет с услугата, като запазват собствеността и управлението над устройството. Някои доставчици внедряват E-SBC в мрежата на доставчика на услуги като виртуално устройство. Други доставчици на услуги за SIP trunking препоръчват на клиентите да закупят и управляват свои собствени E-SBC.

- IP контактни центрове - E-SBC често се внедряват на границите на корпоративната IP мрежа, за да маршрутизират повикванията в разпределени среди на IP контактни центрове. Много организации мигрират старите мрежи на контактните си центрове към изцяло IP мрежи, за да премахнат скъпите такси за приемане и прехвърляне от PSTN (Public Switched Telephone Network).
- IP комуникационни услуги, базирани в облака - E-SBC се използват за защита и контрол на достъпа до отдалечно базирани в облака IP комуникационни услуги, като например аудио или видеоконферентни услуги.
- Мобилни работници и малки офиси (SO - Small Office) - E-SBC се инсталират на границите на мрежи, за да разширят по сигурен начин корпоративните IP комуникационни услуги до малки офиси, като по този начин се допускат и мобилни работници или хора, работещи от разстояние.
- Сигурност при границите на мрежите на доставчиците - Доставчиците на услуги използват SBC за защита и контрол на границите на основната мрежа, както и границите на мрежата за достъп и границите на "взаимовръзките" (връзки към други мрежи на доставчици на услуги или стари мрежи).
- Обединяване на различни комуникационни среди - E-SBC често се използват за трансформиране на фрагментирани комуникационни среди, съставени от различни реализации, системи, планове за набиране, функции и политики.

2.4. Oracle Session Border Controller

2.4.1. Основни сведения и ползи на Oracle Session Border Controller

Oracle Enterprise Session Border Controller е серия от SBC устройства на производителя Oracle. Устройствата на Oracle притежават множество функции и преимущества пред други производители на SBC. На *Фиг.2.3* са показани няколко примерни устройства от серията "Acme Packet" на Oracle. SBC Устройствата на Oracle Communications работят на Acme Packet OS [\[20\]](#).



Фиг.2.3 - SBC устройства на производителя Oracle от серията "Acme Packet"

Основните ползи на SBC на Oracle от гледна точка на компаниите и техния бизнес са показани по-долу [\[20\]](#):

- Защита на всички услуги, базирани на ИТ - независимо дали те са приложения или инфраструктури;

- Усъвършенствана защита от кибератаки - например IP Telephony spam, DoS, DDoS и много други;
- Ускорено въвеждане и разяване на всякакъв вид услуги в конкретната компания.

Oracle Enterprise Session Border Controller е доказано в практиката решение за свързване на широка гама системи за различни видове комуникации (най-вече VoIP) и контактни центрове, реализирани Unified Communications as a Service (UCaaS) или Contact Center as a Service (CCaaS). Заедно с това, Oracle Communications SBC се използват и за SIP trunking услуги, облачни приложения и миграция на данни към облачни услуги.

2.4.2. Преимущества на Oracle Session Border Controller пред устройства на други производители

Oracle SBC защитава IP комуникационните мрежи от кибератаки и измами, заедно с това смекчава последиците от мрежови смущения и прекъсвания. Устройствата на Oracle са оперативно съвместими с устройства на други производители, така че потребителите да могат да ползват надеждни гласови и видео комуникационни услуги от различни производители [\[20\]](#).

A) Защити от кибератаки и прекъсвания

Oracle Enterprise Session Border Controller е устройство, специално проектирано за справяне с все по-значимите проблеми със сигурността, надеждността и оперативната съвместимост, които могат да възникват, когато комуникационните сесии в реално време пресичат мрежовите граници. То защитава от

множество кибератаки и гарантира неприкосновеността на комуникациите, динамично маршрутизиране на самите комуникации, както и опции за манипулация на сесиите, което адресира проблемите с оперативната съвместимост.

С бързото преминаване към облачни технологии, някогашните централизирани "граници" на предприятията се разширяват, което създава трудности, свързани със сигурността, управлението и контрола за предприятията. Потребителите са все по-мобилни, което затруднява поддържането на инфраструктурите и приложениета, използвани от самите потребители. В резултат на това, корпоративните гласови, видео и обединени комуникационни услуги стават все по-уязвими към кибератаки (като например DoS/DDoS), прекъсвания и проблеми с оперативната съвместимост, които могат да възникват, когато комуникационните сесии преминават през границите на IP мрежите. Атаки и прекъсвания могат да нарушаат бизнес операциите, да застрашат приходите и да накърнят репутацията на предприятието. Проблемите с оперативната съвместимост могат да намалят гъвкавостта на бизнеса, да забавят проектите и излагат на риск инвестициите в ИТ сектора.

Б) High Availability (HA)

Повреди и сривове могат да възникнат навсякъде в комуникационната мрежа - включително в SBC. Те могат да бъдат трудни за изолиране и отстраняване и могат да доведат до поражения за съответната компания. Oracle SBC включва пълен набор от функции за осигуряване на непрекъснатост на бизнеса по време на мрежови повреди или сривове. Oracle SBC разполагат с 1:1 High Availability (HA) технология, която непрекъснато следи за състоянието на връзката и маршрутизира сесиите от активни

към резервни устройства без въздействие върху работата на потребителите и техните разговори. По този начин SBC динамично насочват сесиите и защитават от неизправности навсякъде в мрежата. Oracle SBC може да оптимизира производителността в множество SIP сесии, като маршрутизира сесиите въз основа на наблюдаваните QoS и Load Balancing (балансиране на натоварванията). За да помогне на системните администратори да наблюдават и отстраняват неточности в своите мрежи, Oracle SBC разполагат с инструменти за наблюдение и проследяване както на сесиите, така и на HA, които им позволяват бързо да визуализират и наблюдават важна информация за сесиите.

Глава 3. Маршрутизация чрез E.164 Number Mapping сървър и локални политики. Манипулация на Session Initiation Protocol заглавия

3.1. DNS - Name Authority Pointer записи

3.1.1. Основни сведения за Domain Name System

Системата за имена на домейни (Domain Name System, известна още с трибуквеното си съкращение "DNS") превръща имена на домейни (например "www.google.com") в IP адреси, които се използват от браузърите и устройствата в интернет, за да достъпят съответните ресурси в интернет. Всяко устройство, свързано към интернет, има свой IP адрес, който се използва от други устройства за неговото идентифициране [21].

DNS сървърите дават възможност на хората да въвеждат думи в браузърите си, защото да се помнят комбинации от букви (имена) е много по-лесно от това да се помнят комбинации от цифри, които нямат връзка помежду си. По този начин не се налага потребителите да следят IP адреса за всеки уебсайт, който посещават. DNS сървърът в общия случай е устройство с база данни, съдържаща публични IP адреси, свързани с имената на домейни, към които съответният IP адрес води.

DNS действа като телефонен указател в интернет - когато се въведе име на домейн в адресната лента на уеб браузърите, DNS системите намират правилния IP адрес. Този IP адрес на сайта насочва устройството да отиде на правилното място за достъп до данните на уебстраницата.

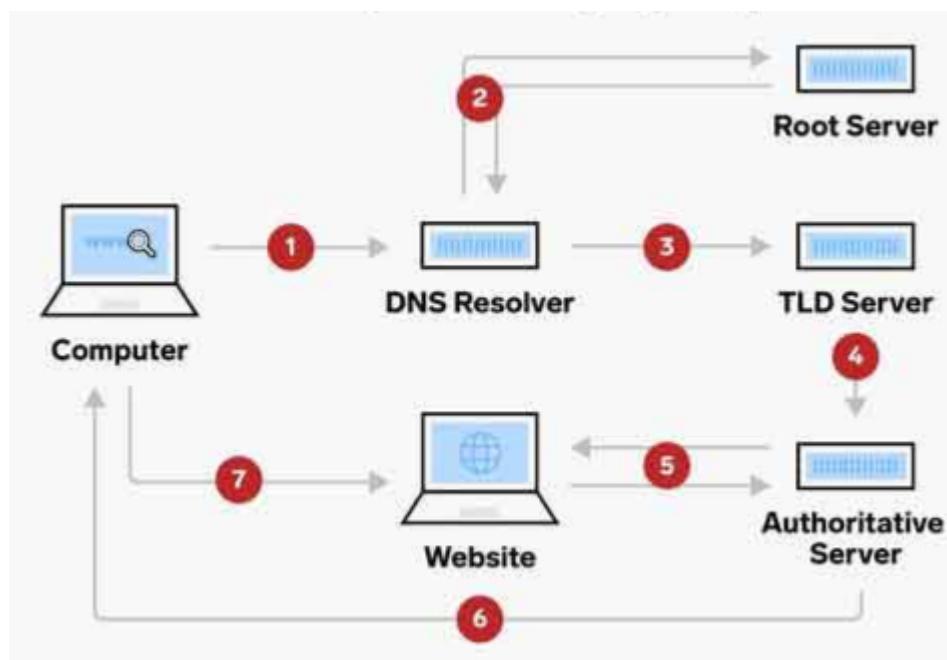
3.1.2. Видове сървъри в Domain Name System

При обикновена DNS заявка въведеният от потребителя URL адрес трябва да премине през четири сървъра, за да бъде предоставен IP адресът. Четирите сървъра работят съвместно, за да осигурят правилния IP адрес. Сървърите са следните [\[21\]](#)[\[22\]](#):

- DNS recursor. Нарича се още DNS resolver. Получава заявка от клиента, след което комуникира с останалите DNS сървъри, за да намери правилния IP адрес. След като получи заявката от клиента, самият той действа като клиент. Прави заявки, които се изпращат до другите три DNS сървъра: Root nameserver, TLD nameserver и authoritative nameserver. Може да се смята за него като за библиотекар, чиято работа е да намери дадена книга (заявка на потребителя) в цялата библиотека, от всичките стелажи с книги (IP адреси).
- Root nameservers. Root nameserver е предназначен за основната зона на DNS в интернет (Root). Неговата задача е да отговаря на изпратените до него заявки за записи в тази основна зона. Той отговаря на заявките, като изпраща обратно списък на авторитетните сървъри на имена, които са свързани с правилния TLD. Може да се разглежда като указател в библиотека, който сочи към различни стелажи с книги - обикновено служи като препратка към други по-специфични места.
- TLD nameservers. Съхранява IP адреса на домейна от второ ниво, който се съдържа в името на TLD. Той се грижи за последната част от името на хоста (в "example.com" TLD сървърът е "com"). Може да се разглежда като специфичен стелаж с книги от библиотека.

- Authoritative nameservers. Сървърът, който дава истинския отговор на DNS запитване. Съществуват два вида авторитетни сървъри за имена: главен/първичен сървър (primary) и подчинен/вторичен сървър (secondary). Може да се разглежда като речник за определен стелаж с книги, в който определено име на книга може да бъде посочено като номер на самата книга от стелажа.

На *Фиг.3.1* е показано взаимодействието между сървърите и клиента, за да се изпълни запитването от страна на клиента за съответен IP адрес на уебсайт:



Фиг.3.1 - Взаимодействие между различните DNS сървъри

3.1.3. Основни видове записи в Domain Name System

Съществуват много типове DNS записи. Най-често използваните са общо девет [\[23\]](#):

- A (Address) - запис, който съдържа IP адреса на даден домейн.
- AAAA - съдържа IPv6 адреса на даден домейн (за разлика от A записите, които съдържат IPv4 адреса).
- CNAME (Canonical NAME) - Пренасочва един домейн или поддомейн към друг домейн (име към име), но НЕ предоставя IP адрес.
- MX (Mail eXchange)- пренасочва към сървър за електронна поща.
- TXT (Text) запис - позволява на администратора да съхранява текстови съобщения и бележки в записа.
- NS (Name Server) - Съхранява сървъра за имена (name server) за даден DNS запис. С други думи, казва кой сървър да обслужва конкретна зона.
- SOA (Start of Authority record) - Съхранява информация за администратора на даден домейн.
- SRV (Service locator) - Посочва порт за определена услуга.
- PTR (Pointer) - Посочва име на домейн и го предоставя при reverse-lookups (обратни прегледи - прегледи в DNS, при които е нужно името, а се знае IP адреса).

3.1.4. NAPTR запис

NAPTR (Name Authority Pointer) е от по-рядко срещаните типове DNS записи [\[24\]](#). Записите NAPTR се използват най-често за приложения в интернет телефонията за съпоставяне на домейни към сървъри и потребителски адреси в SIP. Когато някой извърши проверка на NAPTR на домейн, той разбира какви протоколи работят и на кои

портове са реализирани, както и дали домейнът поддържа SIP или VoIP услуги.

Примерен NAPTR запис е показан на *Фиг.3.2* [25]:

```
$ORIGIN example.com.  
@ 10800 IN NAPTR 100 10 "U" "E2U+sip" "!.+.$!sip:service@example.com!" .
```

Фиг.3.2 - Примерен NAPTR запис в BIND формат

Полетата в записите от такъв тип са следните:

- Host Label (Етикет на хоста) - определя името на хоста на записа и дали името на хоста ще бъде добавено към етикета. В случая стойността е "example.com".
- TTL (Time to Live) - времето за "живот" на записа в секунди, времето за кеширане на записът в секунди. В случая стойността е "10800".
- Record Class (Клас на записа) - съществуват основно 3 класа DNS записи:
 - IN (Internet) - по подразбиране, използват се в интернет.
 - CH (Chaosnet) - използват се за запитване за версии на DNS сървърите.
 - HS (Hesiod) - използва функционалността на DNS за осигуряване на достъп до бази данни.
- Record Type (Тип на записа) - определя типа на записа - NAPTR.
- Order (Ред) е число (0 - 65535), определящо реда, в който трябва да се обработят множество NAPTR записи (от нисък към висок).
- Preference (Предпочтение) - число (0 - 6535), което определя реда (от нисък към висок), в който трябва да бъдат обработени

NAPTR записите с еднакви стойности на Order полето, ако има такива.

- Flags (Флагове) - съдържа флагове за контрол на презаписването и интерпретацията на полетата в записа. Флаговете са единични символи - A-Z или 0-9. Най-често използваните флагове са следните:
 - S (SRV) - следващото търсене/преглед (lookup) трябва да бъде за SRV записи;
 - A (A, AAAA) - следващото търсене трябва да бъде за записи A или AAAA;
 - U (URI) - следващата стъпка не е търсене в DNS, а изходът на полето "Regular Expression" е SIP URI.
- Services (Услуги). Полето посочва параметрите на услугата, приложими към този ресурс. В случая услугата е "E2U+sip" - от номер в E.164 (стандарт за телефонни номера) към URI в SIP.
- Regular Expression (Редовен/регулярен израз) - съдържа израз, който се прилага към оригиналния низ, съхраняван от клиента, за да се конструира следващото име на домейн за търсене. Регулярният израз има следната форма: "Delimit | ERE | Delimiter Substitution | Delimit" - "!^.*\$!sip:service@example.com!", където:
 - Delimiter = "!";
 - ERE (Extended Regular Expression) = В повечето случаи, както и в този = "^.*\$", което съвпада с произволен брой и произволен тип символи (".*") между началото ("^") и края ("\$") на клиентския низ - с други думи, съвпада с целия низ;
 - Delimiter = "!";
 - Substitution=sip:info@example.com;
 - Delimiter = "!".

- Replacement (Замяна) - това поле посочва следващото име на домейн (повечето пъти е FQDN), за което да се направи заявка за търсене. Това поле се използва, когато Regular Expression полето е празно - полетата Regular Expression и Replacement са взаимно изключващи се (по подразбиране само едното може да съдържа стойност по едно и също време). Ако това поле не присъства, трябва да бъде посочено с точка ("."), а ако присъства - полето Regular Expression трябва да съдържа празен низ ("").

3.2. E.164 Number Mapping

3.2.1. Основни сведения за E.164 Number Mapping

E.164 Number Mapping, широко известна като "ENUM", е технология (стандарт), разработена от IETF и описана в RFC 2916, която дефинира метод, по който телефонни номера да се обвържат с ресурси в интернет [\[26\]](#). ENUM указва как да се използва DNS за намиране на услуги, свързани с адреси от E.164 (телефонни номера).

A) E.164 телефонни номера

Уеб адресите в интернет са организирани по метода *Server.Domain.TLD* - в адресната лента на браузърите излгежда като "www.example.org". Телефонният номер е структуриран по сходен начин - единствената разлика е, че той е конструиран в обратната посока - започва се със своеобразно TLD. Той (според E.164 на ITU) има следния строеж: "+", код на страната (може да бъде разглеждано като TLD, за да се направи аналогия с уеб страниците), код на областта (по избор) и накрая абонатен номер.

Тъй като има една глобална система за телефонни номера, всеки номер е уникален и не може да има объркване, независимо

от коя държава или област в държава е съответния номер. Следователно е възможен прост превод на телефони в ресурси в интернет, защото всеки телефонен номер, следващ E.164, е уникален и посочва точно определен ресурс (в частност човек) [\[27\]](#).

Б) Процес на транслация между Е.164 номер и URI

Ако набран номер е +1 555 42 42 и се изисква той да бъде достъпен през интернет, тогава следва да бъде променен във формат за четене в интернет - бива преведен в URI.

Това се получава като [\[28\]](#):

1. Се премахнат всички символи, с изключение на цифрите - 15554242;
2. Се поставят точки (".") между всяка цифра - 1.5.5.5.4.2.4.2;
3. Се обърне реда на цифрите - 2.4.2.4.5.5.5.1;
4. Се добави домейнът ".e164.arpa" или частен ENUM домейн в края - 2.4.2.4.5.5.5.1.e164.arpa.

По този начин номерът +1 555 42 42 е представен като 2.4.2.4.5.5.5.1.e164.arpa. Така DNS сървърите на съответния доставчик на интернет услуги знаят как да попитат root server къде да намерят съответния ENUM сървър за региона "+1" (номерационен план на Северна Америка). Самият ENUM сървър, свързан с конкретния регион, може да отговори с различни отговори, състоящи се от NAPTR записи, които се използват за насочване към различни услуги. Те имат значение, подобно на: "за този домейн пощенският сървър е на адрес x.x.x.x, а уеб сървърът е на адрес x.x.x.y" или "за връзка към номер +1 234 56 78 е нужно да се установи VoIP/SIP сесия със sip:phoneme@example.net". NAPTR записите се използват, за да

разкрият начините, по които могат да се достъпят ресурси в интернет, които са свързани с подаденото от клиента име на домейн за търсене.

На *Фиг.3.3* е представен NAPTR запис, който насочва гореспоменатия телефонен номер към конкретен обект в интернет.

```
$ORIGIN 2.4.2.4.5.5.1.e164.arpa.  
  
IN NAPTR 100 10 "u" "E2U+sip" "!^.*$!sip:phoneme@example.net!".
```

Фиг.3.3 - NAPTR запис на преработения телефонен номер

Този NAPTR запис на практика указва, че за да се установи връзка с номера, трябва да бъде създадена VoIP сесия на адрес "phoneme@example.net". В този момент DNS сървърите започват да търсят example.net в интернет и след като намерят IP адреса, ще потърсят NAPTR записа, който посочва към конкретния SIP обект - SIP сървър. След това към този сървър ще бъде изпратена заявка, за да се инициализира сесия към phoneme@example.net и съответно да се извърши обаждане.

В) Домейн "e164.arpa" и значение на NAPTR записите за ENUM

Домейнът "e164.arpa" се попълва, за да се осигури инфраструктура в DNS за съхраняване на номера от вида E.164. С цел улеснение на разпределените операции, този домейн е разделен на поддомейни. Притежателите на номера по стандарта E.164, които искат номерата им да бъдат включени в DNS, трябва да се свържат със съответния администратор на зона. Политиките за такова вписване могат да се различават в различните части на

света. Може да се установи и частен ENUM домейн, например "e164.sip.abc.def.net", което е по-вероятно да направи едно частно предприятие. Съществуват голям брой подобни частни внедрявания на ENUM [\[28\]](#).

За дадено име в DNS записите от типа NAPTR се използват за идентифициране на наличните начини за връзка с конкретния възел, идентифициран с това име (този номер). По-конкретно, NAPTR записите се използват, за да се разбере какви услуги съществуват за даденото име, включително телефонни номера (които използват домейна e164.arpa), имайл адреси, уеб страници и други, като всички тези услуги могат да се степенуват по желание чрез полето "Preference".

Г) Видове изпълнения на ENUM

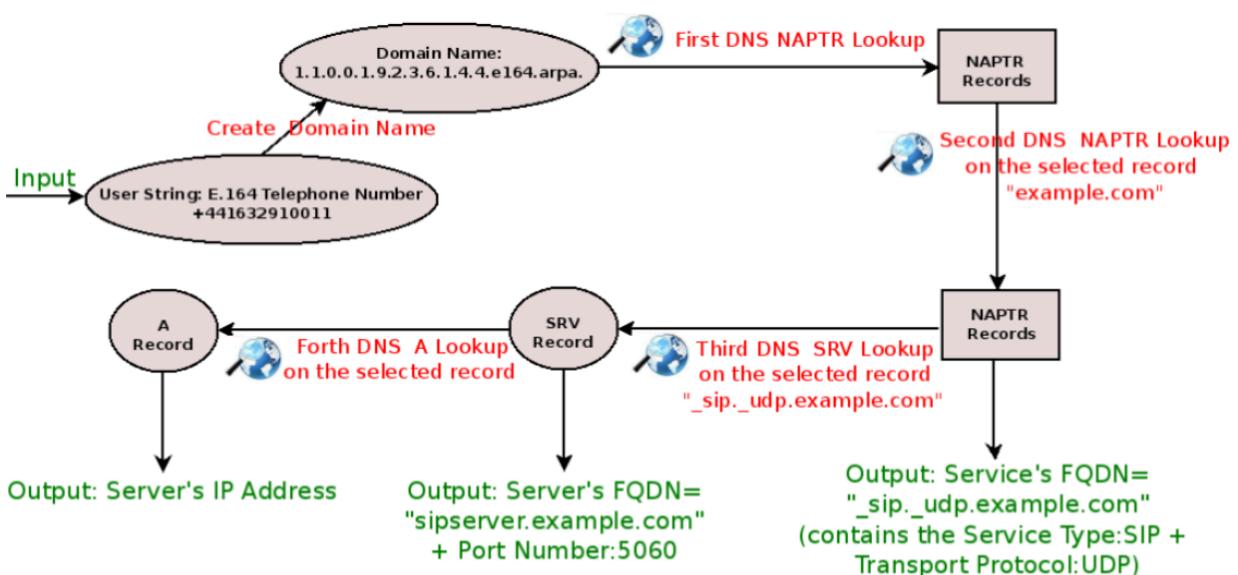
Съществуват три различни типа реализации на ENUM [\[29\]](#):

- Публичен ENUM за потребителите - Този тип имплементация на ENUM услугата позволява на крайните потребители да управляват и въвеждат собствени записи в регистъра на ENUM.
- ENUM в частна инфраструктура - използва се от определени групи, като те определят свой частен ENUM домейн.
- ENUM в публична инфраструктура - Този тип реализация на ENUM се управлява централно от национален администратор на съответната държава. Този орган делегира телефонен номер на операторите, които от своя страна присвояват телефонни номера на крайни потребители. Този тип ENUM е най-защитеният, тъй като

само доставчиците на услуги имат директен достъп до информацията.

3.2.2. Процес на търсене на услуги, свързани с домейн в DNS, образуван от E.164 номер

На Фиг.3.4 е илюстриран процесът на търсене на прилежащи към даден домейн (в случая номер по E.164) услуги и информация за тях, посредством преглед на NAPTR записи от DNS [30].



Фиг.3.4 - Проверяване на домейн, образуван от E.164 номер в DNS

Клиентския низ в случая е "+441632910011" - клиентът иска да разбере с какви услуги е обвързан този номер от формат E.164. Информацията, която касае клиента би могла да бъде типовете на тези услуги, транспортните протоколи за услугите, номерата на портове, IP адресите и друга полезна информация за съответните услуги. За да се започне търсенето в DNS трябва да се превърне номера в име на домейн, като се следват споменатите вече по-горе стъпки. Този домейн

изглежда по следния начин за "+441632910011":
"1.1.0.0.1.9.2.3.6.1.4.4.e164.arpa".

A) Първи преглед в DNS

След извършване на първото DNS търсене на името на домейна "1.1.0.0.1.9.2.3.6.1.4.4.e164.arpa" се разкриват два NAPTR записа, показани на *Фиг.3.5*:

\$ORIGIN 1.1.0.0.1.9.2.3.6.1.4.4.e164.arpa.

NAPTR 10 **100** "u" "E2U+sip" "!.+\$!sip:info@example.com!" . # Record Number 1

NAPTR 10 101 "u" "E2U+h323" "!.+\$!h323:info@example.com!" . # Record Number 2

*Фиг.3.5 - Двата NAPTR записи, свързани с домейн
"1.1.0.0.1.9.2.3.6.1.4.4.e164.arpa"*

И двата записа имат стойност на полето Order 10, но първият има по-ниска стойност на полето Preference, показана в червено (100, по-ниска от 101). Това означава, че DNS клиентът ще избере първия запис. Флагът "u" сигнализира, че изходът ще бъде URI. Името на услугата в избрания запис е "E2U+sip" и това означава, че записът се превежда в URI и след това се използва при заявки от телефонен номер към SIP. Следователно резултатът от първата стъпка е SIP URI, изглеждащо по следния начин: "sip:info@example.com" - използва се образецът, предоставен в подадения Regular Expression [\[30\]](#).

B) Втори преглед в DNS

След първото търсене низа на клиента вече е "sip:info@example.com". Клиентът извършва второ DNS търсене на името на домейна "example.com".

Резултатът на второто търсене отново са два NAPTR записи, представени на *Фиг.3.6*:

```
$ORIGIN example.com.  
IN NAPTR 100 10 "S" "SIP+D2U" "!^.*$!sip:info@example.com!" _sip._udp.example.com.  
IN NAPTR 102 10 "S" "SIP+D2T" "!^.*$!sip:info@example.com!" _sip._tcp.example.com.
```

Фиг.3.6 - Двата NAPTR записи, свързани с домейн "example.com"

При тези два NAPTR записи присъства полето за замяна Replacement (тоест по подразбиране Regular Expression полето трябва да бъде празно, но това не винаги е задължително). Флагът "S" означава, че полето за заместване съдържа FQDN, който сочи към запис от тип SRV. В случая се избира първия запис, защото той има по-ниска стойност на полето ред (Order). Замяната се прилага и резултатът изглежда по следния начин: "_sip._udp.example.com", а от него се разбира, че съответната услуга е SIP и транспортният протокол е UDP.

B) Трети преглед в DNS

След второто търсене низа на клиента извежда SRV запис с вида "_sip._udp.example.com". Този запис има следната форма: "*_Service._Protocol.Name TTL Class Type Priority Weight Port Target*" и е показан на *Фиг.3.7* [30].

_sip._udp.example.com 86400 IN SRV 0 5 5060 sipserver.example.com

Фиг.3.7 - SRV записа, изведен след третия преглед в DNS

Полетата в един SRV запис са обяснени по-долу [31]:

- Service - Услуга: Името на услугата, в случая е "sip";
- Protocol - Протокол: Транспортният протокол, в случая е UDP;
- Name - Име: Името на домейна, в случая е "example.com";
- TTL: Времето за живот (до изтриване) на записа в секунди, в случая е равно на 86400;
- Class - Клас: Клас на записа - IN (Internet);
- Priority - Приоритет (от 0 до 65535), по-ниските стойности са по-предпочитани;
- Weight - Тежест (от 0 до 65535): ако има еднакви стойности на полетата Priority на няколко записа това поле служи за разграничение - по-ниските стойности отново са по-предпочитани;
- Port - Порт: порт, на който услугата "слуша" - 5060 в случая;
- Target - Цел: "sipserver.example.com": Името на хоста на машината, която предоставя съответната услуга, към която посочва SRV записът.

Резултатът от тази трета стъпка е FQDN "sipserver.example.com", който представлява SIP сървър, "слушащ" на порт 5060 и предоставящ възможност за обаждане чрез SIP. Накрая, за да се разбере конкретния адрес в интернет на този домейн, се извършва последно DNS търсене на "sipserver.example.com", което извежда запис от тип A. Този запис съдържа IPv4 адреса на услугата.

3.3. Локални политики

3.3.1. Основни сведения за локални политики

Локалните политики позволяват да се посочи къде да се маршрутизират или препращат заявките за сесии (най-често SIP INVITES), както и да се зададе предпочтение за избор на даден маршрут пред друг. Една локалната политика съдържа информация, която влияе върху маршрутизирането на SIP сигналните съобщения. Най-важните три неща, дефинирани от локалните политики са показани по-долу [32]:

- Параметри "from-address" и "to-address"

Информацията в заглавията "From" и "To" на SIP съобщението се сравнява със записите в параметрите "from-address" "to-address" на локалната политика, за да се определи дали локалната политика се прилага за конкретния случай. С други думи, по този начин локалните политики позволяват прилагането на политики единствено върху трафик, генериран от определени адреси, или върху трафик, насочен към предварително описани в параметъра "to-address" адреси.

- Списък с конфигурирани пространства (Realms)

Този списък идентифицира от кое пространство (realm) идва даден трафик и се използва за маршрутизиране по входящи пространства ("source-realm"). Идентифицираните в списъка пространства на източника трябва да съответстват на валидните идентификатори на пространства, които са предварително конфигурирани.

- Атрибути на локалната политика

Атрибутите служат за изразяване на предпочтания - като средства за избор на един маршрут пред друг. Най-важните параметри съдържат информация за следващия желан "скок" (hop), който най-често е конфигуриран като "Session Agent" (SA) - крайно устройство. Посочва се и неговото конфигурирано пространство. Съществуват и други, не толкова често използвани параметри, например приложен протокол за изпращане на съобщението до следващия скок и времеви рамки за активиране на политиката. Атрибутите на локалните политики могат да бъдат използвани и за филтриране на определени видове трафик.

3.3.2. Маршрутизация чрез локални политики

Маршрутизацията на повикване в рамките на локалната политика може да се основава на съпоставяне на последователност от цифри с дефинирани такива в самата локална политика [\[32\]](#). Тази последователност се отнася до първите цифри в адреса. Сравнението се осъществява отляво надясно.

Ако номерът (адресът), за който трябва да се определи в коя политика да попадне, е "1234567" и стойността на параметъра "from/to-address" в едната локална политика е 123, а стойността на същия параметър в друга локална политика е 12, Oracle SBC препраща повикването към устройството, което е определено като следващ скок в първата локална политика, тоест ще приложи именно нея, защото действа на принципа "Longest prefix = Best match" - там има по-голямо съвпадение. Ако обаче номерът в този случай е "2134567", то тогава не би се намерило съвпадение и не би се приложила нито една от тези локални политики. Същите правила важат и за IP адреси - прилага се политиката, съдържаща в параметъра си "from/to-address" с най-дългото съвпадение.

3.3.3. Предпочтание за пътища, въведени чрез локални политики

Oracle SBC изгражда списък с възможни маршрути въз основа на пространството на източника и адресите From и To, от който след това се определя точния маршрут (от атрибут на локалната политика), накъдето да се изпрати съобщението и/или трафика. Не се използват маршрути на локалната политика, които в момента са извън конфигурираното време/ден, ако тези параметри изобщо са зададени. Пространството на източника се използва в процеса на търсене на локалната политика, но не се използва при изчисляване на предпочтанията на маршрутите [32].

Oracle SBC прилага предпочтение към конфигурираните маршрути в локалните политики в следния ред:

1. Cost. "Разходите" в атрибутите на локалната политика винаги са с най-голямо предимство в избора на маршрути. Ниските числа са по-предпочитани от високите;
2. Съответстващ кодек, определен от опциите за профили в атрибутите на локалната политика;
3. Най-дълъг съответстващ адрес (списък с адреси в локалната политика);
4. Най-кратък съвпадащ адрес (в списъка с адреси в локалната политика);
5. Най-дълго съвпадение на адрес "From" на съобщението с адрес от списъка с адреси в локалната политика;
6. Най-кратко съвпадение на адрес "From" на съобщението с адрес от списъка с адреси в локалната политика;
7. Най-строга спецификация за деня от седмицата (опция за дни от седмицата в атрибутите на локалната политика);

8. Най-строга спецификация за часа на деня (опции за начален и краен час в атрибутите на локалната политика);
9. Съответствия със заместващи символи (например използване на "*" като заместваща стойност за списъците с адреси From и To в локалната политика).

3.4. Манипулация на Session Initiation Protocol header

3.4.1. Основни сведения за Header Manipulation Rules

HMR (Header Manipulation Rules) представляват средства (правила) за преработване на SIP съобщения. Разлики между SIP мрежи, несъвместими внедрявания на доставчици или различни по вид и по конфигурации SIP услуги могат да влошат качеството на съответните SIP услуги и обаждания или да нарушат работата на протокола SIP като цяло. За да се разрешат подобни проблеми, Oracle имплементира правила за манипулиране на заглавията (HMR), които дават на мрежовите администратори възможност да контролират SIP трафика чрез манипулиране на SIP съобщенията. HMR са базирани на набори от правила - правила както за цялото SIP съобщение/заглавие, така и за елементи в него [\[33\]](#):

- Наборите от правила съдържат едно или повече заглавни правила, както и незадължителни правила за елементи, които действат върху определени части на SIP заглавието. Те се прилагат към входящия или изходящия трафик за даден Session Agent (крайно устройство в сесията), пространство или SIP интерфейс.
- Правилата за заглавието действат върху определени заглавия. Те могат да съдържат правила за елементи, всяко от които

определят действията, които да се извършат за даден елемент от заглавието (хедъра).

- Правилата за елементите изпълняват операции върху елементите на даден хедър. Елементите на заглавието включват всички под части на заглавието, с изключение на името на заглавието - например всякакви стойности и параметри на полета от заглавието, стойности под формата на URI и други.

SBC не може да извърши динамично валидиране, докато се въвеждат правила. Това означава, че трябва да се потвърди, че конфигурацията на HMR не съдържа невалидни или "кръгови" препратки/операции в самото правило.

- Невалидна препратка е препратка, която сочи към несъществуващо правило;
- "Кръгова" препратка е препратка, която създава безкраен цикъл от манипулационни действия.

3.4.2. Приложения и функции на Header Manipulation Rules

Най-често HMR се употребяват с цел скриване на чувствителна информация за идентичността на съответния участник в обаждането, както и информация за топологията на неговата вътрешна мрежа, IP адреси и други. Пълен списък с възможностите на HMR по отношение модифициране на SIP хедърите и техните полета и параметри е показан по-долу [\[33\]](#):

- Вмъкване, изтриване или промяна на SIP заглавия или параметри;
- Скриване на данни от заглавия и параметри;

- Копиране или преместване на стойностите на заглавията или параметрите;
- Преименуване имена на параметрите;
- Модифициране на тела на заявките (header body), включително SDP, XML (Extensible Markup Language) и други;
- Категоризиране на специфични потоци от съобщения за специална обработка;
- Улавяне на информация от съобщение и вмъкването ѝ в друго съобщение.

Едно SBC може да извършва действия според HMR въз основа на следното:

- Тип на SIP съобщението (заявка или отговор);
- Тип на заявката (INVITE, REGISTER и др.);
- Успех или неуспех на израз за съвпадение на заглавие или параметър - използва се като филтър дали даден израз, упоменат в съответния HMR, съвпада с израз от SIP заглавието.

3.4.3. Видове Header Manipulation Rules според начина на приложение

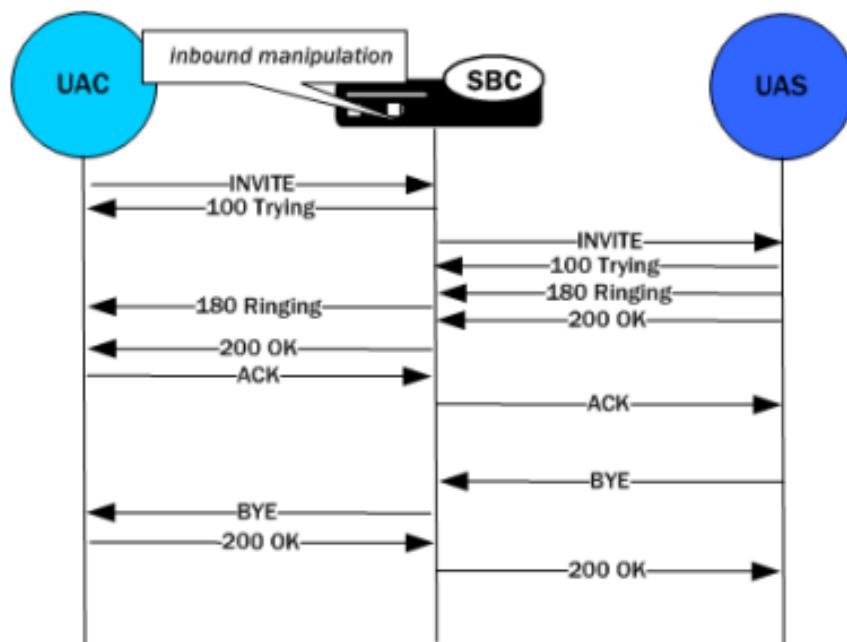
HMR се разделят на два вида според начина и времето на тяхната обработка - "входящи" и "изходящи" [\[33\]](#).

A) Входящи

Входящите (Inbound) правила на HMR се прилагат преди повечето обработки, извършвани от SBC и преди препращането на самото съобщение. Източникът на съобщението се определя, за да се реши към кой Session agent, пространство или SIP интерфейс то принадлежи. По подразбиране правилата за

заглавието се прилагат след обработката на съобщението; по този начин се проверява дали съобщението е добре оформено и следва спецификациите на SIP. Това е необходимо, за да се извърши сигурно всяка последваща обработка на съобщението. Изключение от това правило може да бъде чрез задаване на опцията "inmanip-before-validate".

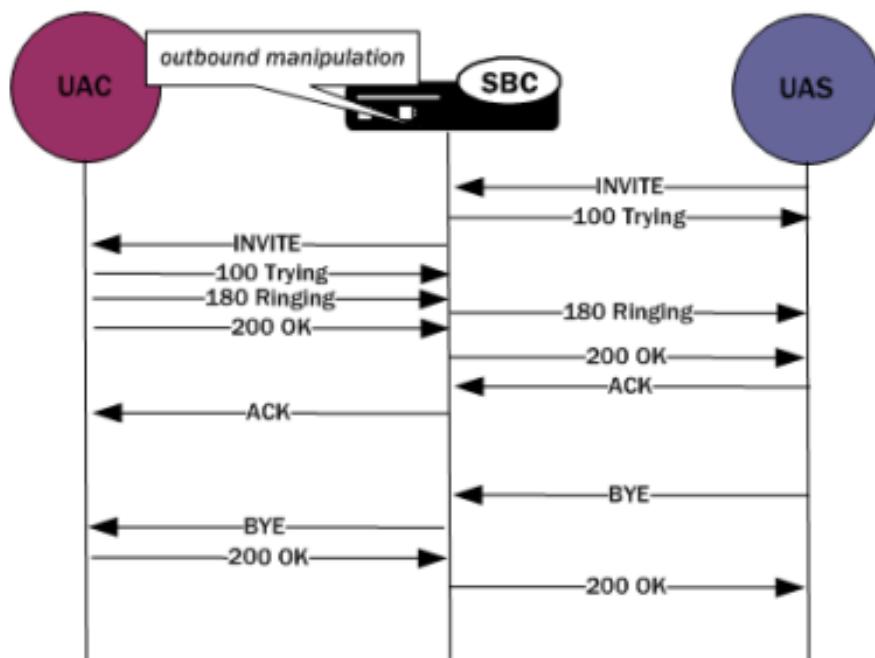
Тъй като входящите правила се прилагат преди съобщението да бъде напълно обработено от SBC, е възможно те да бъдат използвани, за да се извършат специфични действия извън обичайната обработка от SBC устройството. Входящите правила по подразбиране са "stateless" - не пазят/съхраняват стойности от хедърите. Въпреки това, ако SBC е в режим B2BUA (най-често използваният режим) той съхранява и запомня определени стойности на заглавието за по-късна употреба в диалога. Изпълнението на входящите HMR, преди повечето обработки на съобщението, е показано на Фиг.3.8.



Фиг.3.8 - Приложение на входящ HMR

Б) Изходящи

Изходящите HMR правила се прилагат точно преди SIP съобщението да бъде изпратено от SBC, след почти всички видове обработки и решения за маршрутизация. Всички промени, направени от Outbound HMR, се отразяват на съобщението. Правилата се изпълняват отново по "stateless" начин. Те не съхраняват стойности в съобщенията и не помнят какво са направили в предишни съобщения. Възможно е да се заобиколи това поведение, като се конфигурира входящ HMR, който да копира необходимата информация в частен хедър, който след това преминава през SBC и по този начин прави тази информация достъпна. Изходящото правило след това може да потърси в запазената информация и да действа според нея. Изпълнението на изходящите HMR, след повечето обработки от OCSBC, е показано на Фиг.3.9.



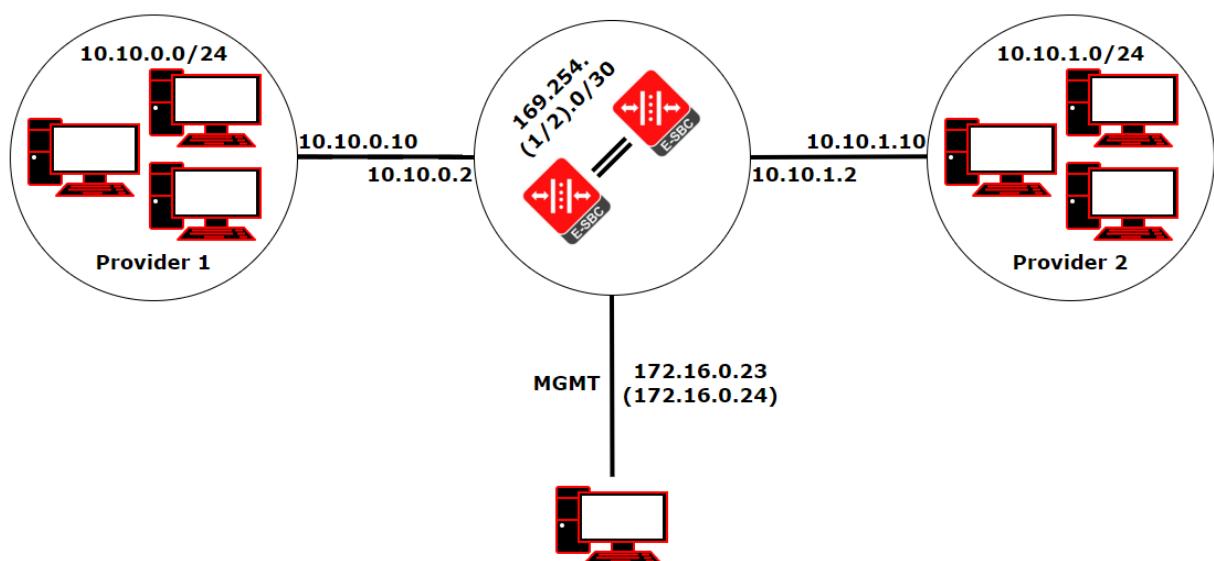
Фиг.3.9 - Приложение на изходящ HMR

Глава 4. Конфигурация и реализиране на свързаността и основната функционалност

4.1. Топологии на мрежата

4.1.1. Логическа топология

На Фиг.4.1 е показана логическата топология на технологичното решение.

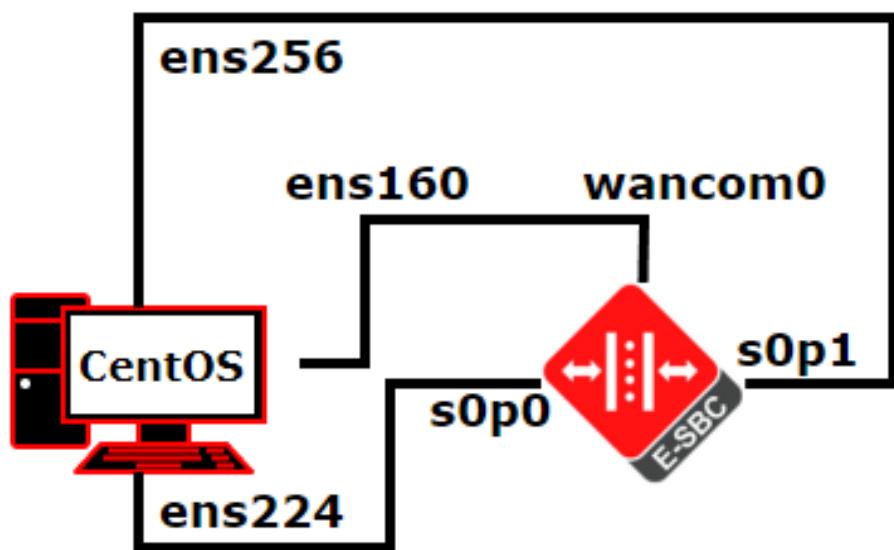


Двете мрежи Provider 1 и Provider 2 въобще са представени от една и съща виртуална машина CentOS, но едната мрежа е репрезентирана от интерфейса "ens224", а другата - от "ens256". Както се вижда на фигурата, мрежата на Provider 1 е 10.10.0.0/24 и има адрес на ens224 интерфейса 10.10.0.10, а мрежата на Provider 2 е 10.10.1.0/24, с адрес на ens256 съответно 10.10.1.10. През Secure Shell (SSH) отдалечена връзка на адреса 172.16.0.23 (172.16.0.24) CentOS машината се свързва с двете SBC устройства за конфигуриране - това е Management (MGMT) интерфейса на Oracle SBC. Самите SBC са

свързани в High Availability възел за резервираност и изглеждат като едно SBC за всички потребители. Помежду си те комуникират през първите два адреса от двете мрежи 169.254.1.0/30 и 169.254.2.0/30.

4.1.2. Физическа топология

На *Фиг.4.2* е представена физическата топология на технологичното решение.



Фиг.4.2 - Физическа топология на мрежата

Физически свързаността на решението се осъществява от една виртуална CentOS машина и едно SBC. Тя има три активни интерфейса - **ens160**, **ens224** и **ens256**. Интерфейсът **ens160** се използва за достъп до самата виртуална машина. **ens224** интерфейсът играе роля на едното крайно устройство в решението, а **ens256** е съответно другото крайно устройство. Те са свързани към **s0p0** и **s0p1** интерфейсите на SBC устройството.

4.2. ACLI Интерфейс

4.2.1. ACLI

ACLI (Acme Command Line Interface) е административният интерфейс, който комуникира с други компоненти на Oracle Communications Session Border Controller (OCSBC - отнася се до всички видове SBC на производителя Oracle Communications). ACLI поддържа интерфейс с модел на въвеждане командите "ред по ред". ACLI е създаден по модела на стандартните CLI в индустрията [\[34\]](#).

A) Нива на достъп

В ACLI има две нива на привилегии: "User" и "Superuser". И двете са защитени с пароли.

- User - на ниво "User" достъпът е ограничен и е до определен набор от инструменти за наблюдение и конфигуриране на Oracle SBC. В това ниво на достъп е възможно:
 - Да се преглеждат версията на конфигурацията, както и статистически данни за системата и нейната производителност;
 - Да се обработва информация за сертификати за функциите на протоколи за сигурност;
 - Да се тестват правила, локални политики и транслации на сесии (session translations);
 - Да се показват системни аларми;
 - Да се задава таймер за наблюдение на системата;
 - Да се задават размерите на дисплея за терминала.

Потребителите се намират в "User" режим, когато системната индикация завършва с триъгълна скоба (">").

- Superuser - На ниво "Superuser" потребителите имат достъп до всички системни команди и права за конфигуриране. Mogat да се използват всички команди, описани и дефинирани в ръководствата на Oracle, както и да се изпълняват и достъпват елементи за конфигуриране. Оттук е достърен първичният елемент "configure terminal", който дава достъп до всички останали конфигурационни елементи в ACLI.

Потребител се намира в "Superuser" режим, когато системната индикация завършва със знака "#", и го достъпва чрез команда "enable" в User режим. Изписането на команда "exit" при който и да е елемент от конфигурацията в ACLI премества потребителя на предишното ниво в ACLI. След излизане от "User" режим потребителя излеза от системата.

Б) Интерфейс и клавишни комбинации в ACLI

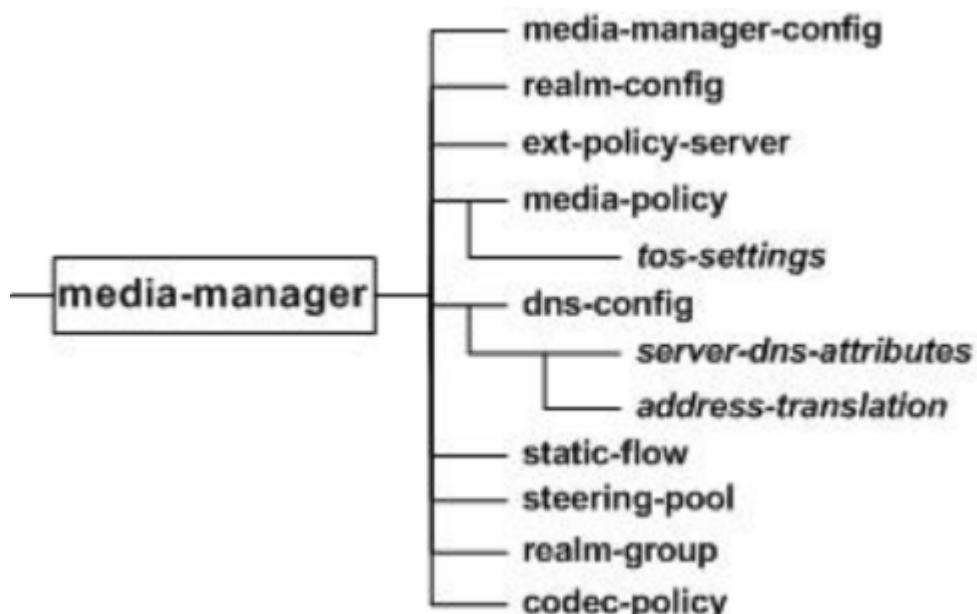
Клавишните комбинации позволяват на потребителите да боравят по-бързо и ефективно с ACLI. Тези бързи клавиши от ACLI на Oracle SBC са подобни на тези които се намират в много други CLI. В ACLI се допуска и частично въвеждане на команди, ако те са уникални в конкретния контекст (например въвеждане на "conf t" вместо "configure terminal") Най-важните бързи клавишни комбинации са представени по-долу [\[34\]](#):

- <стрелка нагоре> - Превърта напред през предишни команди;

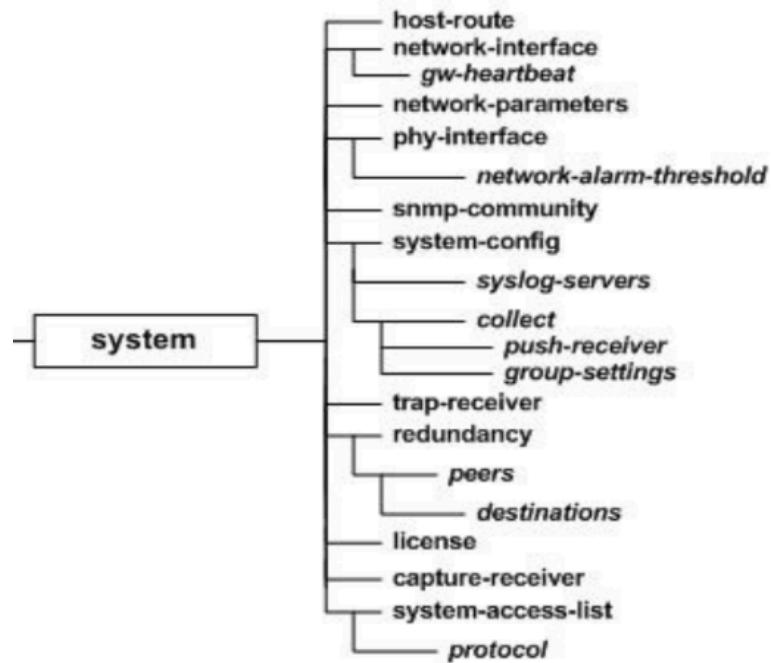
- <стрелка надолу> - Превърта назад през предишни команди;
- <tab> - Завършва частична команда или изброява всички налични опции, ако въведените вече символи съвпадат с няколко команди. Изпълнена в началото на командния ред, този клавиш изброява наличните конфигурируеми команди, елементи и/или параметри;
- <?> - Предоставя контекстна помощ. Функционира както за ACLI командите, така и за елементи на конфигурацията и показва резултатите в азбучен ред;

4.2.2. Конфигурационна йерархия в Oracle SBC

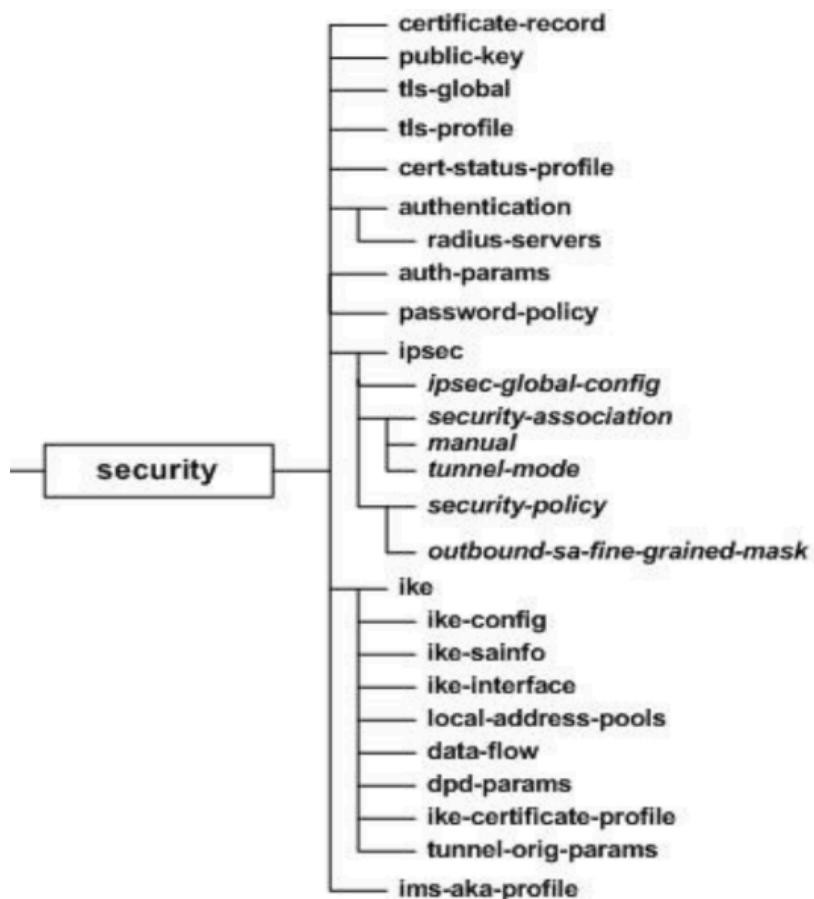
Главните елементи и техните съответни поделементи/модули в конфигурацията на OCSBC са показани на Фиг.4.3 а), б), в) и г). Четирите главни елемента "media-manager", "system", "session-router" и "security" се достъпват от "configure terminal" [\[32\]](#).



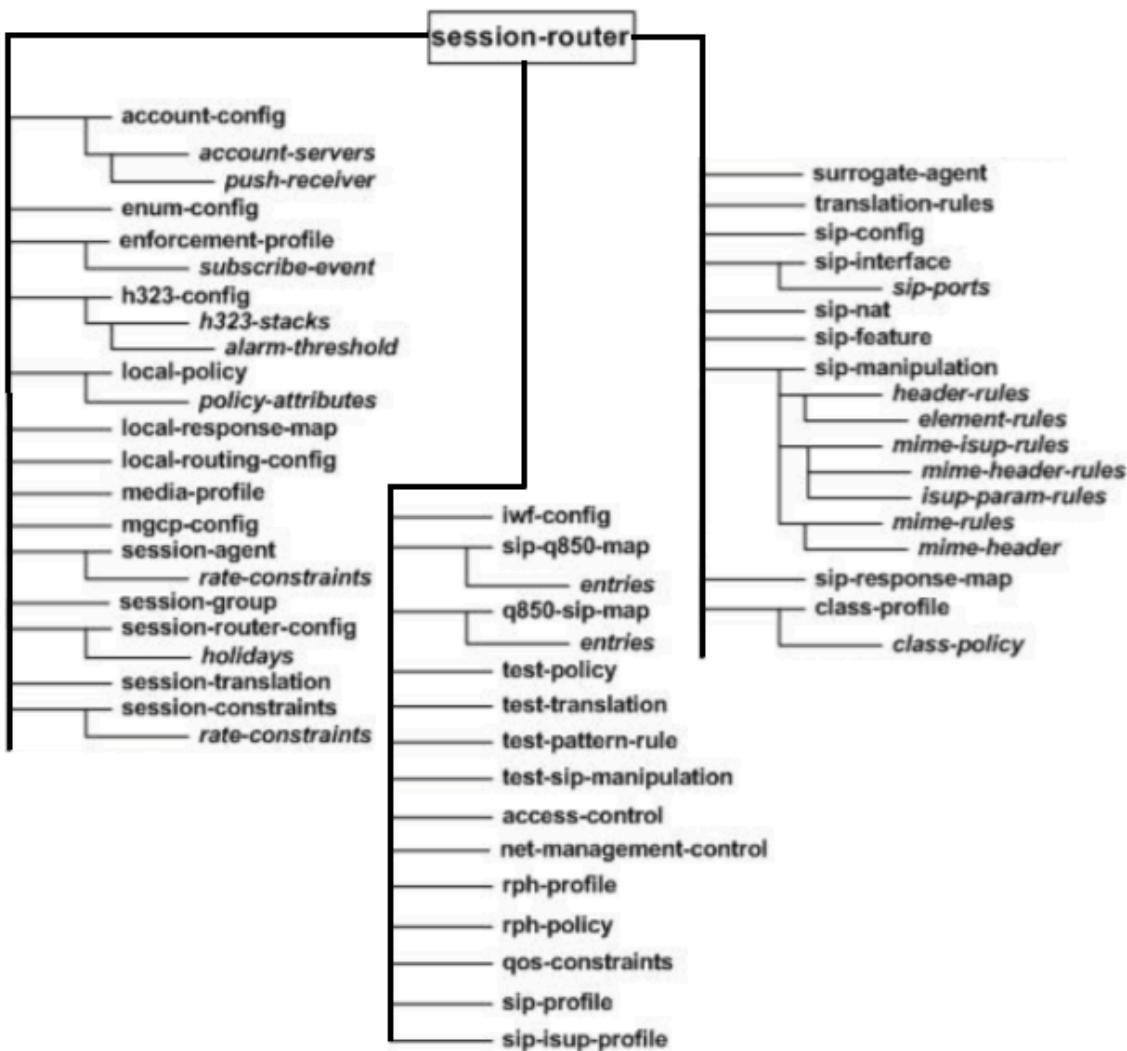
Фиг.4.3 а) - Елемент "media-manager"



Фиг.4.3 б) - Елемент "system"



Фиг.4.3 в) - Елемент "security"



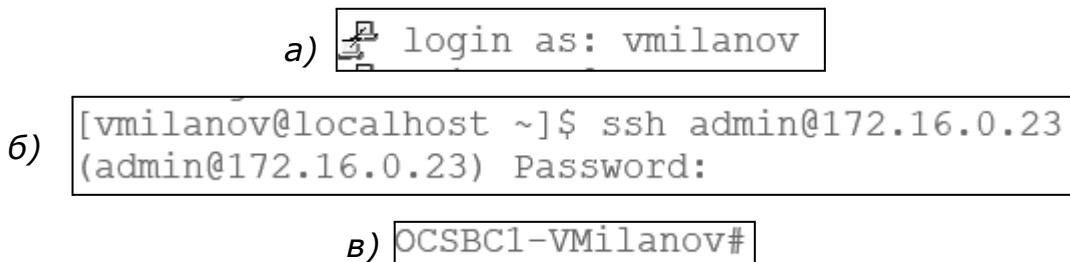
Фиг.4.3 г) - Елемент "session-router"

4.3. Установяване на достъп до SBC и базова конфигурация

4.3.1. Установяване на достъп до SBC

За достъпване на ACLI на двете Oracle SBC устройства се използва SSH през вече достъпена, отново през SSH виртуална CentOS машина. Достъпът до CentOS машината се осъществява чрез потребителското име "vmilanov" и парола. След установяване на достъп до виртуалната машина се стартира SSH сесия с "admin@172.16.0.23" или "admin@172.16.0.24" съответно за двете SBC устройства. За целите на настоящата дипломна работа второто SBC играе ролята на резервно устройство и съответно изпълнява функционалността за резервираност на трафика.

Стъпките за установяване на достъп до първото SBC устройство са показани на *Фиг.4.4 а), б) и в)*. Аналогично се постига достъпът и до второто устройство.



Фиг.4.4 а), б) и в) - Установяване на достъп до SBC

4.3.2. Базова конфигурация

За конфигуриране на всякакъв вид параметри, включително и базовите такива на Oracle SBC устройствата, първо е нужно да се въведе команда "conf t" или "configure terminal", която дава достъп до първичния елемент в дървото на конфигурацията, откъдето след това са достъпни всички останали елементи. Начините за достъпването са

два - чрез пълно изписване или чрез частично изписване. Те, както и индикацията за успешното достъпване до първичния елемент, са показани на *Фиг.4.5 а), б) и в)*.

a) OCSBC1-VMilanov# conf t

б) OCSBC1-VMilanov# configure terminal

в) OCSBC1-VMilanov (configure) #

Фиг.4.5 а), б) и в) - Достъп и индикация за достъп до configure terminal

За проследяване на промените в конфигурацията се въвежда команда "prompt-enabled enable". По този начин преди индикаторите за елементите в началото на командния ред се изписва [\[35\]](#):

- "##", когато конфигурацията е променена, но все още не е записана (done команда е въведена и е имало промени);
 - "*" - конфигурацията е запазена, но все още не е активирана (save-config команда е въведена, но не и activate-config) ;
- Конфигурирането на командата е показано на *Фиг.4.6.*

```
OCSBC1-VMilanov# prompt-enabled enabled  
Current Setting for configuration prompt: enabled
```

Фиг.4.6 - Въвеждане и изход от командата prompt-enabled enable

Конфигурирането на глобалните параметри се извършва от поделемент в елемента system, наречен "system-config". На *Фиг.4.7* е показана последователността от команди за конфигуриране на име и описание на устройството.

```
OCSBC1-VMilanov(configure)# system  
OCSBC1-VMilanov(system)# system-config  
OCSBC1-VMilanov(system-config)# hostname Viktorio-OCSBC1  
OCSBC1-VMilanov(system-config)# description "First SBC, Viktorio"
```

Фиг.4.7 - Конфигуриране на глобални параметри в system-config

Чрез конфигуриране на модула "bootparam" се задават важни конфигурационни променливи за SBC, като например файл с image на операционна система, адрес на Management интерфейс (wancom0), име на устройството и други (Фиг.4.8 а) и б)). Важно е да се отбележи, че "Target Name" взема превес в определянето на името на устройството пред "hostname" от system-config, тоест стойността на "Target Name", ако има въведена такава (както в случая), е реалното име на устройството.

```
OCSBC1-VMilanov# conf t
OCSBC1-VMilanov(configure)# bootparam

'.' = clear field; '-' = go to previous field; q = quit

Boot File          : /boot/bzImage
IP Address        : 172.16.0.23
VLAN              :
Netmask           : 255.255.255.0
Gateway           : 172.16.0.1
IPv6 Address      :
IPv6 Gateway      :
Host IP           :
FTP username      :
FTP password      :
Flags              :
Target Name       : OCSBC1-VMilanov
Console Device    : VGA
Console Baudrate  : 115200
Other              :

NOTE: These changed parameters will not go into effect until reboot.
Also, be aware that some boot parameters may also be changed through
PHY and Network Interface Configurations.
```

Фиг.4.8 а) - Глобални конфигурационни променливи на първото SBC

```

OCSBC2-VMilanov(configure) # bootparam

'.' = clear field; '-' = go to previous field; q = quit

Boot File          : /boot/bzImage
IP Address         : 172.16.0.24
VLAN               :
Netmask            : 255.255.255.0
Gateway            : 172.16.0.1
IPv6 Address       :
IPv6 Gateway       :
Host IP           :
FTP username       :
FTP password       :
Flags              :
Target Name        : OCSBC2-VMilanov
Console Device     : VGA
Console Baudrate   : 115200
Other              :

NOTE: These changed parameters will not go into effect until reboot.
Also, be aware that some boot parameters may also be changed through
PHY and Network Interface Configurations.

```

Фиг.4.8 б) - Глобални конфигурационни променливи на второто SBC

Непосредствено след конфигуриране на нови параметри или редактиране на вече конфигурирани такива се изисква изписването на командата "done", която поставя промените в запазано, но все още непотвърдено състояние. Това състояние е индикирано чрез "##" в началото на командния ред, *Фиг.4.9 а) и б)*. Командата done извежда направените промени върху конфигурацията след въвеждане [\[35\]](#).

a) 

```

OCSBC1-VMilanov(system-config) # done
system-config
    hostname          Viktorio-OCSBC1
    description       Viktorio First OCSBC.

```


b) 

```

**OCSBC1-VMilanov(system-config) # [REDACTED]

```

Фиг.4.9 а) и б) - Команда done и състояние след въвеждането ѝ

На практика, done добавя промените в сегашната конфигурация (running-config), но не и в началната конфигурация (startup-config), тоест при reboot на SBC те не биха влезли в сила. Промените биха влезли в сила само след последователното въвеждане на командите "save-config" и "activate-config", които съответно запазват сегашната конфигурация в паметта и я "активират", тоест я копират в началната. Добра практика е преди тях да се провери конфигурацията за грешки с "verify-config", Фиг.4.10 а), б) и в).

	<pre>**OCSBC1-VMilanov# verify-config</pre>
a)	<pre>----- Verification successful! No errors nor warnings in the configuration</pre>
	<pre>**OCSBC1-VMilanov# save-config checking configuration Save-Config received, processing. save-config waiting 120000 ms for request to finish Request to 'SAVE-CONFIG' has Finished, Save complete</pre>
б)	<pre>Currently active and saved configurations do not match! To sync & activate, run 'activate-config' or 'reboot activate'.</pre>
	<pre>*OCSBC1-VMilanov# activate-config Activate-Config received, processing. activate-config waiting 120000 ms for request to finish</pre>
в)	<pre>Request to 'ACTIVATE-CONFIG' has Finished, Activate Complete</pre>

Фиг.4.10 а), б) и в) - Трите команди и техните изходи

4.4. Конфигуриране на интерфейси

Преди да се конфигурират параметрите за маршрутизация, както и всякакви други функции на SBC, е нужно да се конфигурират интерфейсите на устройството - първо физическите, а след това и мрежовите.

4.4.1. Физически интерфейси - Physical Interfaces

Физическите интерфейси на OCSBC са в основата на всички маршрутизиращи функции. Те се свързват чрез своите имена към мрежовите интерфейси и е от изключителна важност да са конфигурирани правилно и да са успешно свързани, в противен случай устройството не би могло да извършва своите заложени функционалности.

Достъпването на модула за конфигуриране на физическите интерфейси е показано на *Фиг.4.11.*

```
OCSBC1-VMilanov# conf t
OCSBC1-VMilanov(configure)# system
OCSBC1-VMilanov(system)# phy-interface
OCSBC1-VMilanov(phy-interface)# █
```

Фиг.4.11 - Команди за достъпване на модул "phy-interface"

Конфигурирането на първите два физически интерфейса е показано на *Фиг.4.12.* Това са интерфейсите, върху които протича маршрутизацията. Точно поради тази тяхна особеност е нужно параметърът "operation-type" да бъде "Media" - това означава, че те ще бъдат използвани за гласов трафик. Техните имена не са случаини - "s0p0" означава, че това е интерфейсът на първия слот - s0 и на първия port - p0 (броенето започва от 0), а пък "s0p1" означава втория port в същия слот. За превключване между два обекта от 1 модул се използва командалата "select".

```
OCSBC1-VMilanov(phy-interface)# name s0p0
OCSBC1-VMilanov(phy-interface)# operation-type Media
OCSBC1-VMilanov(phy-interface)# name s0p1
OCSBC1-VMilanov(phy-interface)# operation-type Media
OCSBC1-VMilanov(phy-interface)# port 1
```

Фиг.4.12 - Конфигуриране на двата физически интерфейса

Конфигурирането на двата контролни физически интерфейса е показано на *Фиг.4.13.*

```
OCSBC1-VMilanov(phy-interface) # name wancom1
OCSBC1-VMilanov(phy-interface) # operation-type Control
OCSBC1-VMilanov(phy-interface) # port 1
OCSBC1-VMilanov(phy-interface) # wancom-health-score 8
OCSBC1-VMilanov(phy-interface) # name wancom2
OCSBC1-VMilanov(phy-interface) # port 2
OCSBC1-VMilanov(phy-interface) # wancom-health-score 9
OCSBC1-VMilanov(phy-interface) # operation-type Control
```

Фиг.4.13 - Конфигуриране на двата служебни физически интерфейса

Имената на тези интерфейси са конвенционални - "wancom1" и "wancom2" и при тях "operation-type" е зададен на "Control", тоест те са използвани за служебна информация. Тази информация въщност се изразява в съобщенията, препращани между двете устройства, конфигурирани в режим на резервираност. Параметрите "port" отговарят на номера на съответния интерфейс (физически на шасито на устройството). Счита се за добра практика да се конфигурират параметрите "wancom-health-score" за wancom1 и wancom2 по различен начин, което позволява намаляването на точките да бъде лесно разпознаваемо в случай на повреда на някой от тези интерфейси. Когато тези стойности бъдат зададени на "8" за wancom1 и "9" за wancom2 например, може да се гарантира, че повредата на някой от тях няма да повлияе на работата и да смени ролите на устройствата, а заедно с това би се улеснило бързото определяне на това кой интерфейс е изключен само въз основа на загубата на точки [32].

4.4.2. Мрежови интерфейси - Network Interfaces

Модулът за конфигурация на мрежовите интерфейси, наречен "network-interface" се достъпва отново през елемента "system" (Фиг.4.14).

```
OCSBC1-VMilanov# conf terminal  
OCSBC1-VMilanov(configure)# system  
OCSBC1-VMilanov(system)# network-interface  
OCSBC1-VMilanov(network-interface) #
```

Фиг.4.14 - Команди за достъпване на модул "network-interface"

Конфигурирането на първия network interface е представено на Фиг.4.15. Вторият е конфигуриран аналогично, с единствените разлики, че неговото име е "s0p1", а адресите му са от мрежата 10.10.1.0/24, а не от 10.10.0.0/24.

```
OCSBC1-VMilanov(network-interface) # name s0p0  
OCSBC1-VMilanov(network-interface) # description "Network Interface  
OCSBC1-VMilanov(network-interface) # ip-address 10.10.0.2  
OCSBC1-VMilanov(network-interface) # pri-utility-addr 10.10.0.51  
OCSBC1-VMilanov(network-interface) # sec-utility-addr 10.10.0.52  
OCSBC1-VMilanov(network-interface) # netmask 255.255.255.0  
OCSBC1-VMilanov(network-interface) # gateway 10.10.0.1  
OCSBC1-VMilanov(network-interface) # add-icmp-ip 10.10.0.2  
OCSBC1-VMilanov(network-interface) # add-hip-ip 10.10.0.2
```

Фиг.4.15 - Първият конфигуриран network interface

Важно е да се отбележи, че имената на мрежовите интерфейси трябва да съответстват на вече създадените физически интерфейси. Описанието е незначително за работата на устройството - то е за улеснение на администраторите. Адресите в "pri-utility" и "sec-utility" са такива, които се използват за обмен на информация в клъстери с резервираност. В случая няма gateway устройство, но е добра практика за него да се задели първият свободен адрес от мрежата. Адресът на

интерфейса се пише и в "icmr-ip" и "hip-ip", за да не се блокира ping (по подразбиране OCSBC блокира ping към мрежовите интерфейси).

На *Фиг.4.16* е показано конфигурирането на двета служебни мрежови интерфейса - wancom1 и wancom2.

```
OCSBC1-VMilanov(network-interface) # name wancom1
OCSBC1-VMilanov(network-interface) # description "HA Network Interface 1."
OCSBC1-VMilanov(network-interface) # pri-utility-addr 169.254.1.1
OCSBC1-VMilanov(network-interface) # sec-
sec-utility-addr      sec-gateway

OCSBC1-VMilanov(network-interface) # sec-utility-addr 169.254.1.2
OCSBC1-VMilanov(network-interface) # netmask 255.255.255.252
OCSBC1-VMilanov(network-interface) # name wancom2
OCSBC1-VMilanov(network-interface) # description "HA Network Intercace 2."
OCSBC1-VMilanov(network-interface) # pri-utility-addr 169.254.2.1
OCSBC1-VMilanov(network-interface) # sec-utility-addr 169.254.2.2
OCSBC1-VMilanov(network-interface) # netmask 255.255.255.252
```

Фиг.4.16 - wancom1 и wancom2 мрежови интерфейси

Тук "pri-utility-addr" (адресите за служебна комуникация на първичното SBC във връзката) и "sec-utility-addr" (тези на вторичното SBC във връзката) са конфигурирани да бъдат първите адреси от мрежите 169.254.1.0/30 и 169.254.2.0/30. Това на практика означава, че служебния трафик за състоянието на резервираността ще се ипраща точно през тези интерфейси.

След конфигуриране на всяка една от промените се използва команда "done" за запазване и след това "exit", която връща потребителя едно ниво назад в йерархията, което е демонстрирано на *Фиг.4.17.*

```
OCSBC1-VMilanov(network-interface) # exit
OCSBC1-VMilanov(system) # █
```

Фиг.4.17 - Използване на команда "exit" на ниво "network-interface"

4.5. Реализиране на резервираност между SBC устройствата

Моделът за High Availability на Oracle SBC представлява кълстеп от два възела (устройства), при който активният възел осигурява нормално обслужване според конфигурацията си, а резервният комуникира с активния, така че да може незабавно ролите да бъдат прехвърлени при непредвидена аварийна ситуация. Когато резервният възел стане активен (поради отказ на предишния активен или поради ръчно превключване), той репликира оригиналния активен възел във всички аспекти, включително и получава рамки на същите MAC адреси, които той е използвал. Това на практика означава, че промяната на възлите в кълстера остава напълно незабележима за другите устройства в мрежата [\[32\]](#).

Посредством съобщения между интерфейсите на SBC wancom1 и/или wancom2 се проверява състоянието на връзката и то е известно и на двета възела - всеки от тях знае за другия и за неговото състояние при нормални условия. Доколкото е възможно, в зависимост от естеството на повредата, двете устройства ще запазят своята комуникация и ще се обменят "heartbeats" - механизъм за keep-alive. В случай на прекъсване на връзката, както е показано, heartbeats все още ще се обменят. Резервен възел обикновено става активен поради индикация, че "здравето" (health) на активния възел е спаднало под определен праг - например може резервният възел да стане активен, след като не е получил heartbeat съобщения от активния възел дълго време. Максималната стойност на параметъра health е 100 точки. Едно действие на SBC може да намали, увеличи или да остави състоянието непроменено. Например, по подразбиране, "link-down" (падане на връзката) намалява здравето с 50 точки, а "link-up" (обратно възвръщане на връзката) - го увеличава с 50. По този начин, когато

результатът се понижи под предварително зададен праг, настъпва "switchover" - смяна на активното устройство.

Веднъж конфигурирани в режим на HA, Oracle SBC устройствата притежават функция за придобиване на конфигурация от основното SBC във връзката. Само едно SBC - основното трябва да бъде конфигурирано, защото [32]:

- Конфигурацията определя кое SBC е първично и кое е вторично;
- Конфигурацията съдържа параметри, които да се използват от първичното и, както и такива, които да бъдат използвани от вторичното SBC;
- Когато първичното устройство се стартира, то знае, че е първично и използва предвидените за него параметри;
- Когато вторичното устройство е придобило конфигурацията (чрез acquire-config) и след това е рестартирано, то знае, че е вторично и използва своите предвидени в конфигурацията параметри.

Последователността от команди, нужни за достъпването на модула за резервираност и включването на самата резервираност, са показани на *Фиг.4.18*.

```
OCSBC1-VMilanov# configure terminal  
OCSBC1-VMilanov(configure)# system  
OCSBC1-VMilanov(system)# redundancy  
OCSBC1-VMilanov(redundancy)# select  
OCSBC1-VMilanov(redundancy)# state enabled
```

Фиг.4.18 - Команди за достъпване на модул "redundancy" и включване на резервираността

Стойностите за health не са променяни и са оставени по подразбиране. Конфигурирането на двета възела (peers) в ACLI е показано на *Фиг.4.19 а) и б)*.

```
OCSBC1-VMilanov(redundancy)# peers
OCSBC1-VMilanov(rdncy-peer)# name OCSBC1-VMilanov
OCSBC1-VMilanov(rdncy-peer)# state enabled
OCSBC1-VMilanov(rdncy-peer)# type Primary
OCSBC1-VMilanov(rdncy-peer)# destinations
OCSBC1-VMilanov(rdncy-peer-dest)# address 169.254.1.1:9090
OCSBC1-VMilanov(rdncy-peer-dest)# network-interface wancom1:0
OCSBC1-VMilanov(rdncy-peer-dest)# address 169.254.2.1:9090
OCSBC1-VMilanov(rdncy-peer-dest)# network-interface wancom2:0
```

Фиг.4.19 а) - Конфигуриране на резервираност за първично SBC

```
OCSBC1-VMilanov(rdncy-peer)# name OCSBC2-VMilanov
OCSBC1-VMilanov(rdncy-peer)# state enabled
OCSBC1-VMilanov(rdncy-peer)# type Secondary
OCSBC1-VMilanov(rdncy-peer)# destinations
OCSBC1-VMilanov(rdncy-peer-dest)# address 169.254.1.2:9090
OCSBC1-VMilanov(rdncy-peer-dest)# network-interface wancom1:0
OCSBC1-VMilanov(rdncy-peer-dest)# address 169.254.2.2:9090
OCSBC1-VMilanov(rdncy-peer-dest)# network-interface wancom2:0
```

Фиг.4.19 б) - Конфигуриране на резервираност за вторично SBC

В модула "destinations" се задават адресите на всеки от възлите на wancom1 и/или wancom2 интерфейсите, които ще бъдат ангажирани със служебния трафик между двете устройства, който от своя страна е на порт 9090.

След конфигурация на гореспоменатите параметри и запазването им без грешки, от вторичното SBC се изпълняват командите "delete-config", "reboot", "acquire-config" и накрая отново "reboot", в тази последователност. Командата "acquire-config" изисква изтриване на конфигурацията преди своето изпълнение. Тази поредица от команди се използва, за да се изтрие сегашната конфигурация и

придобие конфигурацията от първичното устройство, така впоследствие се задейства резервираността - *Фиг.4.20 а) и б).*

```
OCSBC2-VMilanov# acquire-config 172.16.0.23
Must delete configuration and reboot to acquire-config
OCSBC2-VMilanov# delete-config
*****
Do you really want to ERASE the saved config?: [y/n]?: 
```

Фиг.4.20 а) - Командите acquire-config и delete-config

```
OCSBC2-VMilanov# reboot
-----
WARNING: you are about to reboot this SBC!
-----
Reboot this SBC [y/n]?: y
OCSBC2-VMilanov# Connection to 172.16.0.24 closed.
```

Фиг.4.20 б) - команда reboot

Проверката на синхронизацията на конфигурациите се случва чрез командите "display-current-cfg-version" или "display-running-cfg-version", които съответно показват версията на запазената или на текущата конфигурация. Изход от команда "display-current-cfg-version" е показан на *Фиг.4.21 а) и б).*

```
OCSBC2-VMilanov# display-current-cfg-version
Current configuration version is 41
```

Фиг.4.21 а) - Командата във вторичното SBC

```
OCSBC1-VMilanov# display-current-cfg-version
Current configuration version is 41
```

Фиг.4.21 б) - Командата в първичното SBC

Проверката за правилното функциониране на резервираността се реализира чрез команда "show health" Фиг.4.22 а) и б).

OCSBC2-VMilanov# show health

Media Synchronized	true
SIP Synchronized	true
REC Synchronized	disabled
XSERV Synchronized	disabled
Config Synchronized	true
Collect Synchronized	disabled
RADIUS CDR Synchronized	disabled
Rotated CDRs Synchronized	disabled
IPSEC Synchronized	disabled
Iked Synchronized	disabled
Lbpd Synchronized	disabled
tCCD Synchronized	disabled
Service Health Synchronized	true
Active Peer Address	169.254.2.1

Redundancy Protocol Process (v3):

State	Standby
Health	100
Lowest Local Address	169.254.1.2:9090

1 peer(s) on 2 socket(s):

OCSBC1-VMilanov: v3, Active, health=100, max silence=1050
last received from 169.254.2.1 on wancom2:0

Switchover log:

a)

OCSBC1-VMilanov# show health

Media Synchronized	true
SIP Synchronized	true
REC Synchronized	disabled
XSERV Synchronized	disabled
Config Synchronized	true
Collect Synchronized	disabled
RADIUS CDR Synchronized	disabled
Rotated CDRs Synchronized	disabled
IPSEC Synchronized	disabled
Iked Synchronized	disabled
Lbpd Synchronized	disabled
tCCD Synchronized	disabled
Service Health Synchronized	true
Active Peer Address	

Redundancy Protocol Process (v3):

State	Active
Health	100
Lowest Local Address	169.254.1.1:9090

1 peer(s) on 2 socket(s):

OCSBC2-VMilanov: v3, Standby, health=100, max silence=1050
last received from 169.254.1.2 on wancom1:0

Switchover log:

б)

Фиг.4.22 б) - "show health" на вторично и първично SBC

4.6. Реализиране на маршрутизацията

4.6.1. Пространства

Следващата стъпка за реализиране на маршрутизация след конфигурирането на интерфейси и на резервираност е конфигурирането на пространства (realms). Един физически интерфейс може да има няколко мрежови интерфейса, които от своя страна могат да имат много пространства - тази йерархия наподобява "дърво", защото на практика на един физически интерфейс могат да съществуват много пространства, но не и обратно.

На Фиг.4.23 е показано достъпването на конфигурирането пространствата ("realm-config"), както и двете пространства. Те са кръстени съответно "Provider 1" и "Provider2" и се свързват с вече съществуващи мрежови интерфейси (s0p0 и s0p1). Символите ":0" в края на интерфейсите указват, че се използва "sub-port-id=0" от параметрите на мрежовите интерфейси, което е стойността по подразбиране. Всяка различна от "0" стойност създава VLAN (Virtual Local Area Network) със съответната стойност, което не е нужно за целите на настоящата дипломна работа [\[32\]](#).

```
OCSBC1-VMilanov(configure)# media-manager
OCSBC1-VMilanov(media-manager)# realm-config
OCSBC1-VMilanov(realm-config)# identifier Provider1
OCSBC1-VMilanov(realm-config)# description "Provider1 realm."
OCSBC1-VMilanov(realm-config)# net
network-interfaces      net-management-control

OCSBC1-VMilanov(realm-config)# network-interfaces s0p0:0
OCSBC1-VMilanov(realm-config)# identifier Provider2
OCSBC1-VMilanov(realm-config)# description "Provider2 realm."
OCSBC1-VMilanov(realm-config)# network-interfaces s0p1:0
```

Фиг.4.23 - Достъпване и конфигуриране на пространства

4.6.2. Модул "sip-config"

Следващият важен модул е "sip-config". Той е запазен със стойностите по подразбиране, които позволяват SIP трафик и дават на SBC ролята на B2BUA (това се контролира от опцията "operation-mode", която е оставена на "dialog"). Параметрите "home-realm-id" и "egress-realm-id" са конфигурирани да са със стойност Provider1, защото в този realm се намира DNS сървърът, на който се изпращат заявки от страна на OCSBC, а "registrar-domain" и "registrar-host" са със стойност "*", защото регистрации не са нужни за реализацията на заданието [\[32\]](#).

Конфигурирането е илюстрирано на *Фиг.4.24*.

```
OCSBC1-VMilanov(sip-config)# state enabled
OCSBC1-VMilanov(sip-config)# operation-mode dialog
OCSBC1-VMilanov(sip-config)# home-realm-id Provider1
OCSBC1-VMilanov(sip-config)# egress-realm-id Provider1
OCSBC1-VMilanov(sip-config)# registrar-domain *
OCSBC1-VMilanov(sip-config)# registrar-host *
```

Фиг.4.24 - Конфигуриране на "sip-config"

4.6.3. SIP интерфейси

SIP интерфейси всъщност са интерфейсите, през които се изпраща SIP комуникацията между крайните устройства и SBC устройствата. Те са свързани с вече съществуващ realm чрез параметъра "realm-id".

Параметрите в подмодула "sip-ports" определят кой би могъл да се свърже за съответния SIP интерфейс и на кой адрес. Точно заради това полето "address" се попълва с адреса на някой от съществуващите мрежови интерфейси (в случая е адресът на s0p0 за първия SIP интерфейс и този на s0p1 за втория). Параметърът "allow-anonymous" е зададен на "agents-only", за да са позволени обаждания само чрез съществуващи и дефинирани крайни устройства.

Конфигурирането на SIP интерфейсите е представено на *Фиг.4.25 а) и б).*

```
OCSBC1-VMilanov(sip-interface) # realm-id Provider1  
OCSBC1-VMilanov(sip-interface) # description "SIP-Interface  
OCSBC1-VMilanov(sip-interface) # sip-ports  
OCSBC1-VMilanov(sip-port) # address 10.10.0.2  
OCSBC1-VMilanov(sip-port) # allow-anonymous agents-only
```

Фиг.4.25 а) - Конфигуриране на първия SIP интерфейс

```
OCSBC1-VMilanov(sip-interface) # realm-id Provider2  
OCSBC1-VMilanov(sip-interface) # description "SIP-Interface  
OCSBC1-VMilanov(sip-interface) # sip-ports  
OCSBC1-VMilanov(sip-port) # address 10.10.1.2  
OCSBC1-VMilanov(sip-port) # allow-anonymous agents-only
```

Фиг.4.25 б) - Конфигуриране на втория SIP интерфейс

4.6.4. Модул "steering-pool"

Модулът "steering-pool" представлява средство за делегиране на порт от предварително конфигуриран набор от такива. Тези портове са за RTP трафик от OCSBC - на практика указват максималния брой на едновременни сесии със съответния негов интерфейс. По този начин се установява комуникация между крайно устройство (обикновено на порт 5060) и SBC (на порт от списъка в "steering-pool").

Конфигурирането на двета модула е показано на *Фиг.4.26.*

```
OCSBC1-VMilanov(media-manager) # steering-pool  
OCSBC1-VMilanov(steering-pool) # ip-address 10.10.0.2  
OCSBC1-VMilanov(steering-pool) # start-port 20000  
OCSBC1-VMilanov(steering-pool) # end-port 20999  
OCSBC1-VMilanov(steering-pool) # realm-id Provider1  
OCSBC1-VMilanov(steering-pool) # ip-address 10.10.1.2  
OCSBC1-VMilanov(steering-pool) # start-port 21000  
OCSBC1-VMilanov(steering-pool) # end-port 21999  
OCSBC1-VMilanov(steering-pool) # realm-id Provider2
```

Фиг.4.26 - Достъпване до и конфигуриране на "steering-pool"

Портовете за SBC са конфигурирани в диапазона 20000-20999 за първото устройство и 21000-21999 за второто - това означава, че в момента са предвидени максимално по 1000 комуникационни сесии на интерфейс.

4.6.5. Session Agents

Session Agents (SA) представляват крайните устройства в мрежата. Те са всъщност клиентите в едно обаждане и генерираят заявки за комуникация, било то чрез инсталирани софтфони или VoIP телефони.

Достъпването на модула и конфигурирането на двета агента е показано на *Фиг.4.27.*

```
OCSBC1-VMilanov(session-router) # session-agent
OCSBC1-VMilanov(session-agent) # hostname PhoneProvider1
OCSBC1-VMilanov(session-agent) # ip-address 10.10.0.10
OCSBC1-VMilanov(session-agent) # realm-id Provider1
OCSBC1-VMilanov(session-agent) # description "Provider1 agent phone."
OCSBC1-VMilanov(session-agent) # hostname PhoneProvider2
OCSBC1-VMilanov(session-agent) # ip-address 10.10.1.10
OCSBC1-VMilanov(session-agent) # port 5061
OCSBC1-VMilanov(session-agent) # realm-id Provider2
OCSBC1-VMilanov(session-agent) # description "Provider2 agent phone."
```

Фиг.4.27 - Достъпване и конфигуриране на SA

Важни особености за клиентите са, че те са в две отделни пространства и "слушат" на различни портове - единият отговаря на порт 5060 (порът по подразбиране), а другият е зададен на 5061.

4.6.6. DNS сървър и база данни MariaDB

За реализирането на ENUM функционалността на дипломната работа е нужен DNS сървър, който поддържа NAPTR записи и съответно заявки. Чрез подобни заявки е възможно SBC устройствата да получават отговори, които съответно успяват или не да "валидират"

конкретен SA. SA е "валидиран", ако получен контактен низ от DNS NAPTR заявка съвпада с вече конфигуриран адрес на SA в SBC.

A) DNS сървър

За DNS сървър е използван сървърът PowerDNS (PDNS), защото той е способен да работи с Linux машини и поддържа NAPTR записи. PowerDNS е Authoritative DNS сървър с отворен код, който предлага добра производителност и има минимални изисквания към паметта, което е важно за машини с ограничен ресурс (каквите са и виртуалните). PowerDNS работи в комбинация с много технологии, вариращи от прости файлове със зони (.zone файлове) до бази данни на различни SQL платформи, най-често MySQL [\[36\]](#). За целта на дипломната работа pdns е инсталиран като услуга и е конфигуриран на ens224 интерфейса, като използва порт 53.

Конфигурационният файл на PowerDNS, който се намира в "/etc/pdns" и се нарича "pdns.conf", е показан на *Фиг.4.28*.

```
GNU nano 5.6.1          /etc/pdns/pdns.conf
# Autogenerated configuration file template

launch=gmysql
gmysql-host=localhost
gmysql-user=viktorio
gmysql-password=*****
gmysql dbname=powerdns
logging-facility=0
local-address=10.10.0.10
local-port=53
```

Фиг.4.28 - Конфигурационен файл на PowerDNS - /etc/pdns/pdns.conf

Важно е да се отбележи, че след каквito и да било промени по конфигурацията на PowerDNS е нужно услугата на име "pdns" да се рестартира, например с команда, подобна на: "*sudo systemctl restart pdns*". PowerDNS използва отделна програма, наречена PowerDNS Recursor (*pdns_recursor*), за свой DNS Recursor. Той функционира между крайния потребител и authoritative DNS сървър. Заяvkите, подадени от крайния потребител, пристигат първо в рекурсивния DNS сървър, който след това търси записите в своя кеш. Ако заяvеният запис не може да бъде намерен в кеша (не е бил кеширан/използван скоро), той изпраща заявката до authoritative сървър и го кешира за употреба напред във времето.

Б) База данни MariaDB

MariaDB е технология за бази данни с отворен код. Използва се за съхранение на данни в табличен формат - с редове и колони. MariaDB е модифицирана версия на MySQL - една от най-широко разпространените бази данни. Всички SQL (Structured Query Language) базирани решения са наричани "релационни" бази данни [\[37\]](#). Релационните бази данни са най-разпространените и използвани в момента. Информацията в релационни бази данни е организирана в таблици с редове и колони, в които всеки ред има уникален идентификатор - ID. Между данните съществуват връзки (оттук релационни, "relation" - връзка), което прави достъпа и работата с тях много по-гъвкави и ефективни, особено когато става въпрос за ясно структурирана информация [\[38\]](#). MariaDB е адаптиран MySQL клиент в много различни системи, и е използван заради интеграцията му с PowerDNS. Преди да бъде използван този клиент, той трябва да бъде инсталиран като услуга.

Създаването на таблици и бази данни чрез MariaDB се извършва след влизане чрез конзола в MariaDB с командата "mysql -u root -p", което позволява достъп като "root" - потребител с неограничени права, от който е възможно да се създават други потребители и бази данни (*Фиг.4.29*) [39].

```
[vmilanov@localhost ~]$ mysql -u root -p
Enter password:
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MariaDB connection id is 4885
Server version: 10.5.22-MariaDB MariaDB Server

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and
its subsidiaries. All rights reserved.

Type 'help;' or '\h' for help. Type '\c' to clear the current
query.

MariaDB [(none)]> 
```

Фиг.4.29 - Установяване на достъп до MariaDB терминал като root

Последователността за създаване на нужните таблици за функцията на DNS сървърът е показана на *Фиг.4.30 а), б), в) и г)*. За влизане в създадена базата данни се използва команда "use powerdns".

```
MariaDB [(none)]> create database powerdns;
Query OK, 1 row affected (0.000 sec)

MariaDB [(none)]> create user 'pdns' identified by 'mypassword' ;
Query OK, 0 rows affected (0.000 sec)

MariaDB [(none)]> grant all privileges on powerdns.* to
'pdns'@'localhost' identified by 'mypassword';
Query OK, 0 rows affected (0.000 sec)

MariaDB [(none)]> flush privileges;
Query OK, 0 rows affected (0.001 sec)
```

Фиг.4.30 а) - Създаване на база данни и привилегии на потребител

```

CREATE TABLE domains (
    id                  INT AUTO_INCREMENT,
    name                VARCHAR(255) NOT NULL,
    master              VARCHAR(128) DEFAULT NULL,
    last_check          INT DEFAULT NULL,
    type                VARCHAR(6) NOT NULL,
    notified_serial     INT DEFAULT NULL,
    account             VARCHAR(40) DEFAULT NULL,
    PRIMARY KEY (id)
) Engine=InnoDB;

CREATE UNIQUE INDEX name_index ON domains(name);

```

Фиг.4.30 б) - Създаване на таблица "domains" в "powerdns"

```

CREATE TABLE records (
    id                  BIGINT AUTO_INCREMENT,
    domain_id          INT DEFAULT NULL,
    name                VARCHAR(255) DEFAULT NULL,
    type                VARCHAR(10) DEFAULT NULL,
    content             VARCHAR(64000) DEFAULT NULL,
    ttl                 INT DEFAULT NULL,
    prio               INT DEFAULT NULL,
    change_date         INT DEFAULT NULL,
    disabled            TINYINT(1) DEFAULT 0,
    ordername           VARCHAR(255) BINARY DEFAULT NULL,
    auth                TINYINT(1) DEFAULT 1,
    PRIMARY KEY (id)
) Engine=InnoDB;

```

Фиг.4.30 в) - Създаване на таблица "records"

```

CREATE INDEX nametype_index ON records(name,type);
CREATE INDEX domain_id ON records(domain_id);
CREATE INDEX recordorder ON records (domain_id, ordernumber);

```

Фиг.4.30 в) - Индексиране на таблица "records"

Таблицата records трябва да бъде синхронизирана с domains и съответни записи (в случая на настоящата дипломната работа NAPTR) да бъдат свързани с конкретен домейн от domains (в случая той е "e164.arpa"). Въвеждането на записи и след това визуализация на текущите записи в таблица records чрез командата "*SELECT * FROM records;*" са показани на *Фиг.4.31 а) и б)*.

```

INSERT INTO records (domain_id, name, type, content, ttl)
VALUES (
    (SELECT id FROM domains WHERE name = 'e164.arpa'),
    '4.3.2.1.e164.arpa',
    'NAPTR',
    '100 10 "u" "E2U+sip" "!^.*$!sip:4321@10.10.0.10!".',
    86400
);

```

Фиг.4.31 а) - Команда за въвеждане на произволен NAPTR запис в records

name	type	content
5.5.5.e164.arpa	NAPTR	10 100 "u" "E2U+sip" "!^.*\$!sip:5555@192.168.0.3!".
e164.arpa	SOA	ns1.example.com hostmaster.example.com 1 86400 7200
1.1.1.1.e164.arpa	NAPTR	100 10 "u" "E2U+sip" "!^.*\$!sip:1111@10.10.0.10!".
2.2.2.2.e164.arpa	NAPTR	100 10 "u" "E2U+sip" "!^.*\$!sip:2222@10.10.1.10!".
8.8.8.8.e164.arpa	NAPTR	100 10 "u" "E2U+sip" "!^.*\$!sip:8888@10.10.0.2!".
9.9.9.9.e164.arpa	NAPTR	100 10 "u" "E2U+sip" "!^.*\$!sip:9999@10.10.1.2!".
3.3.3.3.e164.arpa	NAPTR	100 10 "u" "E2U+sip" "!^.*\$!sip:123@10.10.0.10!".
4.4.4.4.e164.arpa	NAPTR	100 10 "u" "E2U+sip" "!^.*\$!sip:123@10.10.1.10!".
6.6.6.6.e164.arpa	NAPTR	100 10 "u" "E2U+sip" "!^.*\$!sip:6666@10.10.1.10!".
7.7.7.7.e164.arpa	NAPTR	100 10 "u" "E2U+sip" "!^.*\$!sip:7777@10.10.0.10!".

Фиг.4.31 б) - Визуализация на записи от таблица records

За правилното функциониране на DNS услугата е нужен и запис в records от тип SOA [40]. Записи от този тип записи съхраняват важна информация за домейна, като например имейл адреса на администратора, информация за последната актуализация на домейна, колко време трябва да изчака сървърът между обновяванията и други. Всички DNS зони, включително и "e164.agra" се нуждаят от SOA запис, за да отговарят на стандартите на IETF и да са възможни отговорите на заявки за елементи от тях.

4.6.7. ENUM функционалност на SBC

Модулът от конфигурацията на OCSBC, който се използва за конфигуриране на ENUM функционалността за конкретното SBC се нарича "enum-config". Чрез конфигуриране на параметри в този поделемент, устройството има възможността да прави NAPTR запитвания към DNS сървър за домейни, образувани от E.164 номера и съответно да получава от този сървър NAPTR записи. Те биха разкрили на OCSBC услугите зад съответния домейн (в случая тези услуги са изразени чрез IP адрес за контакт със съответен дефиниран Session Agent) [41].

Достъпването на и конфигурирането на модула "enum-config" са показани съответно на *Фиг.4.32 а) и б)*.

```
OCSBC1-VMilanov# configure terminal  
OCSBC1-VMilanov(configure)# session-router  
OCSBC1-VMilanov(session-router)# enum-config  
OCSBC1-VMilanov(enum-config)#[ ]
```

Фиг.4.32 а) - Достъпване на "enum-config"

```
OCSBC1-VMilanov(enum-config) # name enumconf
OCSBC1-VMilanov(enum-config) # top-level-domain e164.arpa
OCSBC1-VMilanov(enum-config) # realm-id Provider1
OCSBC1-VMilanov(enum-config) # enum-servers 10.10.0.10:53
OCSBC1-VMilanov(enum-config) # health-query-number +5555
OCSBC1-VMilanov(enum-config) # health-query-interval 60
```

Фиг.4.32 б) - Конфигуриране на ENUM в OCSBC

Важни параметри за изследването на NAPTR заявките от SBC към DNS и като цяло на свързаността между двете системи са "health-query-number" и "health-query-interval", в които се задава номер и интервал за периодични заявки към DNS с цел keep-alive (проверка на връзката).

4.6.8. Локални политики

Достъпването и конфигурирането на двете локални политики, включващи "enum:enumconf" вместо SA в "policy-attributes" за валидиране на SA чрез ENUM, са показани съответно на на Фиг.4.33 а), б) и в).

```
OCSBC1-VMilanov# conf t
OCSBC1-VMilanov(configure) # session-router
OCSBC1-VMilanov(session-router) # local-policy
OCSBC1-VMilanov(local-policy) # █
```

Фиг.4.33 а) - Достъпване на модул "local-policy" в OCSBC

```
OCSBC1-VMilanov(local-policy) # from-address *
OCSBC1-VMilanov(local-policy) # to-address *
OCSBC1-VMilanov(local-policy) # source-realm Provider1
OCSBC1-VMilanov(local-policy) # policy-attributes
OCSBC1-VMilanov(local-policy-attributes) # next-hop enum:enumconf
OCSBC1-VMilanov(local-policy-attributes) # action replace-uri
OCSBC1-VMilanov(local-policy-attributes) # app-protocol SIP
```

Фиг.4.33 б) - Първа локална политика

```

OCSBC1-VMilanov(local-policy)# from-address *
OCSBC1-VMilanov(local-policy)# to-address *
OCSBC1-VMilanov(local-policy)# source-realm Provider2
OCSBC1-VMilanov(local-policy)# policy-attributes
OCSBC1-VMilanov(local-policy-attributes)# next-hop enum:enumconf
OCSBC1-VMilanov(local-policy-attributes)# action replace-uri
OCSBC1-VMilanov(local-policy-attributes)# app-protocol SIP
OCSBC1-VMilanov(local-policy-attributes)# exit

```

Фиг.4.33 в) - Втора локална политика

Единствената разлика между двете локални политики е в параметъра "source-realm", който в единия случай е със стойност "Provider1", а в другия - "Provider2". Това означава, че на практика двете се прилагат във всяка получена заявка от съответното пространство и, ако не се намери съответствие между отговор от NAPTR запитване към DNS и предварително конфигуриран SA, OCSBC ще блокира обаждането.

4.7. Конфигурация на софтфони

За осъществяване на обаждания са нужни два SIP клиента, закачени за двета интерфейса на CentOS машината. По този начин е възможно установяването на разговор между тях в рамките на една виртуална машина. За целта са използвани софтфони "1" (за ens224:5060) и "2" (за ens256:5061). След инсталация и стартиране, конфигурационните им параметри са представени съответно на *Фиг.4.34 а), б) и в)*.

Main SIP account settings

SIP address*	<input type="text" value="sip:viktorio2@10.10.0.2"/>
SIP Server address*	<input type="text" value="<sip:10.10.0.2;transport=udp>"/>

Фиг.4.34 а) - Конфигурационни параметри в клиент "1"

viktorio1@10.10.1.2

SIP Credentials

Domain	10.10.1.2
Username	viktorio1

Фиг.4.34 б) - Конфигурационни параметри в клиент "2"

Network

SIP options:

Port: 5061 Open random available port

Фиг.4.34 в) - Конфигурация на порт в клиент "2"

Важно е да се отбележи, че портът по подразбиране на двете програми е 5060, но използването му и на двете места би довело до проблеми в разговорите - винаги само единият софтфон би отговарял на обаждания, затова в конфигурацията на клиент "2" (както и на SA с адрес 10.10.1.10) е зададен порт 5061. Заедно с това, и на двета софтфона е изключена опцията за регистрация на клиенти, защото тя не е нужна за реализацията на решението. Имената на клиентите са символични и нямат отношение към обажданията, защото самите обаждания се осъществяват чрез номера във формат E.164 чрез ENUM.

4.8. Реализиране на манипулация на Session Initiation Protocol хедър

HMR представляват мощен инструмент, чрез който е възможно цялата информация от един SIP/SDP хедър да бъде манипулирана [42]. Правилата за манипулация на хедъри се намират в модула "sip-manipulation" и често се наричат "набор от правила" (rulesets).

Наборите от правила съдържат "правила за заглавия" (header rules), като едно правило за заглавие може да съдържа "правила за елементи" (element rules):

- Правилата за заглавието работят върху цялото заглавие/хедър.
- Правилата за елемент работят върху избрани елементи в рамките на определен хедър.

Достъпването и конфигурирането на модула "sip-manipulation" е показано на *Фиг.4.35 а), б) и в)*.

```
OCSBC1-VMilanov# conf t
OCSBC1-VMilanov(configure)# session-router
OCSBC1-VMilanov(session-router)# sip-manipulation
OCSBC1-VMilanov(sip-manipulation) # █
```

Фиг.4.35 а) - Модула "sip-manipulation"

```
OCSBC1-VMilanov(sip-manipulation) # name topology_hiding
OCSBC1-VMilanov(sip-manipulation) # description "Hiding From and To."
OCSBC1-VMilanov(sip-manipulation) # header-rules
OCSBC1-VMilanov(sip-header-rules) # name NewTo
OCSBC1-VMilanov(sip-header-rules) # header-name To
OCSBC1-VMilanov(sip-header-rules) # action manipulate
OCSBC1-VMilanov(sip-header-rules) # msg-type request
OCSBC1-VMilanov(sip-header-rules) # element-rules
OCSBC1-VMilanov(sip-element-rules) # name NewToValue
OCSBC1-VMilanov(sip-element-rules) # type uri-host
OCSBC1-VMilanov(sip-element-rules) # action replace
OCSBC1-VMilanov(sip-element-rules) # match-val-type ip
OCSBC1-VMilanov(sip-element-rules) # new-value $REMOTE_IP
OCSBC1-VMilanov(sip-element-rules) # done
```

Фиг.4.35 б) - Конфигуриране на "sip-manipulation" и на първи ruleset

```

OCSBC1-VMilanov(sip-header-rules)# name NewFrom
OCSBC1-VMilanov(sip-header-rules)# header-name From
OCSBC1-VMilanov(sip-header-rules)# action manipulate
OCSBC1-VMilanov(sip-header-rules)# msg-type request
OCSBC1-VMilanov(sip-element-rules)# name NewFromValue
OCSBC1-VMilanov(sip-element-rules)# type uri-h
uri-header      uri-header-name uri-host

OCSBC1-VMilanov(sip-element-rules)# type uri-host
OCSBC1-VMilanov(sip-element-rules)# action replace
OCSBC1-VMilanov(sip-element-rules)# match-val-type ip
OCSBC1-VMilanov(sip-element-rules)# new-value $LOC
OCSBC1-VMilanov(sip-element-rules)# new-value $LOCAL_IP

```

Фиг.4.35 в) - Конфигуриране на втори ruleset

Двата конфигурирани набора от правила са съответно за To и From полетата в SIP хедъра и по-точно техните стойности на "uri-host" елемента (тоест частта след "@" - IP адреса). Данните се заменят по начин, по който съответният SA да не може да разбере за другата страна в обаждането - за него разговорът се провежда единствено със SBC.

За да може да се приложи HMR е нужно в пространство, SIP интерфейс, или SA да се дефинира това правило съответно като входящо или изходящо. В случая, то е дефинирано като изходящо. Това действие е направено и за двете пространства "Provider1" и "Provider2" и е показано на *Фиг.4.36*.

```
OCSBC1-VMilanov(realms-config)# out-manipulationid topology_hiding
```

Фиг.4.36 - Задаване на HMR като изходящ в пространство

По подобен начин е създаден и втори HMR, който е наречен "no_hiding" и е с празни параметри, за да бъде възможно превключването между състоянията на манипулация на SIP заглавията.

Глава 5. Тестване на свързаността и доказване на работоспособността

5.1. Елементи и параметри на конфигурацията

За доказване на работоспособността на конфигурацията е използвана командата "show", както и нейни разновидности (в частност "show configuration <име на модул> [short]").

5.1.1. Интерфейси

A) Физически интерфейси

Конфигурацията на физическите интерфейси е показана на *Фиг.5.1* чрез въвеждане на командата "show configuration phy-interface short".

```
OCSBC1-VMilanov# show configuration phy-interface short
phy-interface
    name                                s0p0
    operation-type                      Media
    duplex-mode
    speed
phy-interface
    name                                s0p1
    operation-type                      Media
    port                                 1
phy-interface
    name                                wancom1
    port                                 1
    duplex-mode
    speed
    wancom-health-score                 8
phy-interface
    name                                wancom2
    port                                 2
    duplex-mode
    speed
    wancom-health-score                 9
```

Фиг.5.1 - Конфигурация на физическите интерфейси

Чрез добавяне на "short" в командата се показват само направените промени, а параметрите, които се извеждат без стойност, са оставени със стойностите по подразбиране.

Б) Мрежови интерфейси

Конфигурацията на мрежовите интерфейси е представена на *Фиг.5.2* чрез въвеждане на команда "show configuration network-interface short".

```
OCSBC1-VMilanov# show configuration network-interface short
network-interface
  name                      s0p0
  description               Network Interface associated with s0p0 port.
  ip-address                10.10.0.2
  pri-utility-addr          10.10.0.51
  sec-utility-addr          10.10.0.52
  netmask                   255.255.255.0
  gateway                   10.10.0.1
  hip-ip-list               10.10.0.2
  icmp-address              10.10.0.2
network-interface
  name                      s0p1
  description               Network Interface associated with s0p1 port.
  ip-address                10.10.1.2
  pri-utility-addr          10.10.1.51
  sec-utility-addr          10.10.1.52
  netmask                   255.255.255.0
  gateway                   10.10.1.1
  hip-ip-list               10.10.1.2
  icmp-address              10.10.1.2
network-interface
  name                      wancom1
  description               HA Network Interface 1.
  pri-utility-addr          169.254.1.1
  sec-utility-addr          169.254.1.2
  netmask                   255.255.255.252
network-interface
  name                      wancom2
  description               HA Network Interface 2.
  pri-utility-addr          169.254.2.1
  sec-utility-addr          169.254.2.2
  netmask                   255.255.255.252
```

Фиг.5.2 - Конфигурация на мрежовите интерфейси

5.1.2. Конфигурация за резервираност

Конфигурацията за резервираността на двете OCSBC е показана на **Фиг.5.3** чрез въвеждане на команда "show configuration redundancy-config short".

```
OCSBC1-VMilanov# show configuration redundancy-config short
redundancy-config
    peer
        name          OCSBC1-VMilanov
        type          Primary
        destination
            address      169.254.1.1:9090
            network-interface wancom1:0
        destination
            address      169.254.2.1:9090
            network-interface wancom2:0
    peer
        name          OCSBC2-VMilanov
        type          Secondary
        destination
            address     169.254.1.2:9090
            network-interface wancom1:0
        destination
            address     169.254.2.2:9090
            network-interface wancom2:0
```

Фиг.5.3 - Конфигурация за резервираност

5.1.3. Пространства

Конфигурацията на пространствата е представена на **Фиг.5.4** чрез въвеждане на команда "show configuration realm-config short".

```
OCSBC1-VMilanov# show configuration realm-config short
realm-config
    identifier      Provider1
    description     Provider1 realm.
    network-interfaces s0p0:0
    out-manipulationid topology_hiding
realm-config
    identifier      Provider2
    description     Provider2 realm.
    network-interfaces s0p1:0
    out-manipulationid topology_hiding
```

Фиг.5.4 - Конфигурация на пространствата

5.1.4. Модул "sip-config" и SIP интерфейси

A) Модул "sip-config"

Конфигурацията на модула "sip-config" е представена на

Фиг.5.5 чрез въвеждане на "show configuration sip-config short".

```
OCSBC1-VMilanov# show configuration sip-config short
sip-config
    home-realm-id                                Provider1
    egress-realm-id                               Provider1
    registrar-domain                            *
    registrar-host                                *
```

Фиг.5.5 - Конфигурация на модула "sip-config"

B) SIP интерфейси

Конфигурацията на SIP интерфейсите е показана на **Фиг.5.6**

чрез въвеждане на "show configuration sip-interface short".

```
OCSBC1-VMilanov# show configuration sip-interface short
sip-interface
    realm-id                                     Provider1
    description                                  SIP-Interface for Provider1 Realm.
    sip-port
        address                                 10.10.0.2
        allow-anonymous                         agents-only
sip-interface
    realm-id                                     Provider2
    description                                  SIP-Interface for Provider2 Realm.
    sip-port
        address                                 10.10.1.2
        allow-anonymous                         agents-only
```

Фиг.5.6 - Конфигурация на SIP интерфейси

5.1.5. Session Agents

Конфигурацията на двата сесийни агента е показана на **Фиг.5.7**

чрез въвеждане на команда "show configuration session-agent short".

```
OCSBC1-VMilanov# show configuration session-agent short
session-agent
    hostname                                    PhoneProvider1
    ip-address                                 10.10.0.10
    realm-id                                   Provider1
    description                                Provider1 agent phone.
session-agent
    hostname                                    PhoneProvider2
    ip-address                                 10.10.1.10
    port                                       5061
    realm-id                                   Provider2
    description                                Provider2 agent phone.
```

Фиг.5.7 - конфигурация на SA

5.1.6. ENUM функционалност

Конфигурацията на ENUM в OCSBC е представена на *Фиг.5.8* чрез въвеждане на команда "show configuration enum-config short".

```
OCSBC1-VMilanov# show configuration enum-config short
enum-config
    name                                enumconf
    top-level-domain                     e164.arpa
    realm-id                            Provider1
    enum-servers                        10.10.0.10:53
    health-query-number                 +5555
    health-query-interval               60
```

Фиг.5.8 - ENUM конфигурация в OCSBC

Примерно запитване към DNS сървъра, с цел проверка на работоспособността на самия сървър, се извършва чрез команда "dig" с модификатор "-t NAPTR", за да се специфицира вида на запис, който се търси. След това е въведен примерен запис от таблицата "7.7.7.7.e164.arpa" и е указан адреса на сървъра, към който се насочва заявката. По този начин цялата команда изглежда така: "dig -t NAPTR 7.7.7.7.e164.arpa @10.10.0.10" и нейният резултатът е показан на *Фиг.5.9*.

```
[vmilanov@localhost ~]$ dig -t NAPTR 7.7.7.7.e164.arpa @10.10.0.10
; <>> DiG 9.16.23-RH <>> -t NAPTR 7.7.7.7.e164.arpa @10.10.0.10
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 43884
;; flags: qr aa rd; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1
;; WARNING: recursion requested but not available

;; OPT PSEUDOSECTION:
;; EDNS: version: 0, flags:; udp: 1232
;; QUESTION SECTION:
;7.7.7.7.e164.arpa.           IN      NAPTR

;; ANSWER SECTION:
7.7.7.7.e164.arpa.     86400   IN      NAPTR    100 10 "u" "E2U+sip" "!^.^$!sip:7777@10.10.0.10!" .

;; Query time: 0 msec
;; SERVER: 10.10.0.10#53(10.10.0.10)
;; WHEN: Sun Feb 25 19:23:33 EET 2024
;; MSG SIZE  rcvd: 100
```

Фиг.5.9 - Резултат от команда "dig"

Примерно запитване към DNS сървъра от страна на SBC, както и проверка на кешираните записи, са осъществени съответно с командите "show enum lookup enumconf +7777" и "show enum cache-entry enumconf 7.7.7.7.e164.arpa" и са показани на *Фиг.5.10*.

```
OCSBC1-VMilanov# show enum lookup enumconf +7777
Enum Lookup Result:
Query Name -->
+7777
Answers -->
sip:7777@10.10.0.10 ttl= 86400
OCSBC1-VMilanov# show enum cache-entry enumconf 7.7.7.7.e164.arpa
DNS Result:
Query Name -->
NAPTR:7.7.7.7.e164.arpa
Answers -->
order=100 pref=10 "u" "E2U+sip" "!^.^$!sip:7777@10.10.0.10!"
```

Фиг.5.10 - Запитване в DNS от SBC и кеширан запис от SBC

Периодичните запитвания, които се случват през 60 секунди за запис "5.5.5.5.e164.arpa", както и информацията от отговорите на DNS сървъра, са илюстрирани съответно на *Фиг.5.11 а) и б)*. Използвана е програмата Wireshark.

dns						
No.	Time	Source	Destination	Proto	Ler	Info
35	32.15...	10.10.0.2	10.10.0.10	DNS	77	Standard query 0x255f NAPTR 5.5.5.5.e164.arpa
36	32.16...	10.10.0.10	10.10.0.2	DNS	1...	Standard query response 0x255f NAPTR 5.5.5.5.
95	92.15...	10.10.0.2	10.10.0.10	DNS	77	Standard query 0x2560 NAPTR 5.5.5.5.e164.arpa
96	92.16...	10.10.0.10	10.10.0.2	DNS	1...	Standard query response 0x2560 NAPTR 5.5.5.5.

Фиг.5.11 а) - Периодични запитвания през 60 секунди към DNS

```
.....5.5.5.5.e
164.arpa .#.....
#....Q.. +...d.u.
E2U+sip. !^.^$!si
p:5555@1 92.168.0
.3!.
```

Фиг.5.11 б) - Информация от отговори на запитванията

5.1.7. Модул "steering-pool"

Конфигурацията на модула "steering-pool" в OCSBC е илюстрирана на *Фиг.5.12* чрез въвеждане на команда "show configuration steering-pool short".

```
OCSBC1-VMilanov# show configuration steering-pool short
steering-pool
    ip-address          10.10.0.2
    start-port          20000
    end-port            20999
    realm-id            Provider1
steering-pool
    ip-address          10.10.1.2
    start-port          21000
    end-port            21999
    realm-id            Provider2
```

Фиг.5.12 - Конфигурация на модула "steering-pool"

5.1.8. Локални политики

Конфигурацията на локалните политики е показана на *Фиг.5.13* чрез въвеждане на команда "show configuration local-policy short".

```
OCSBC1-VMilanov# show configuration local-policy short
local-policy
    from-address          *
    to-address             *
    source-realm           Provider1
    description            Local-policy from Provider1.
    policy-attribute
        next-hop           enum:enumconf
        action              replace-uri
        app-protocol        SIP
local-policy
    from-address          *
    to-address             *
    source-realm           Provider2
    description            Local-policy from Provider2.
    policy-attribute
        next-hop           enum:enumconf
        action              replace-uri
        app-protocol        SIP
```

Фиг.5.13 - Конфигурация на локалните политики

5.1.9. Правила за манипулация на SIP заглавия

Конфигурацията на HMR е показана на Фиг.5.14 чрез въвеждане на "show configuration sip-manipulation short".

```
OCSBC1-VMilanov# show configuration sip-manipulation short
sip-manipulation
    name                                     no_hiding
sip-manipulation
    name                                     topology_hiding
    description                               Hiding From and To.
    header-rule
        name                                     NewFrom
        header-name                             From
        action                                    manipulate
        element-rule
            name                                     NewFromValue
            type                                      uri-host
            action                                   replace
            match-val-type                         ip
            new-value                                $LOCAL_IP
    header-rule
        name                                     NewTo
        header-name                             To
        action                                    manipulate
        element-rule
            name                                     NewToValue
            type                                      uri-host
            action                                   replace
            match-val-type                         ip
            new-value                                $REMOTE_IP
```

Фиг.5.14 - Конфигурация на правила за манипулация на хедъри

5.2. Резервираност на устройствата

Симулиране на switchover (размяна на ролите на двете OCSBC) се извършва чрез командата "notify berpd force". Преди нейното изпълнение е нужно администратора да се увери, че двете устройства са успешно синхронизирани с команда "show health" (Фиг.5.15).

```
Redundancy Protocol Process (v3):
State                           Active
Health                          100
Lowest Local Address           169.254.1.1:9090
1 peer(s) on 2 socket(s):
OCSBC2-VMilanov: v3, Standby, health=100, max silence=1050
                                last received from 169.254.1.2 on wancom1:0
```

Фиг.5.15 - Част от команда, показваща успешната синхронизацията

Резултатът след командите "notify berpd force" и след това "show health" е показан на *Фиг.5.16*.

```
OCSBC1-VMilanov# notify berpd force
OCSBC1-VMilanov# Feb 25 18:40:19.874: Active to RelinquishingActive, forced by command

OCSBC1-VMilanov# show health

  Media Synchronized          true
  SIP Synchronized           true
  REC Synchronized           disabled
  XSERV Synchronized          disabled
  Config Synchronized          true
  Collect Synchronized          disabled
  RADIUS CDR Synchronized          disabled
  Rotated CDRs Synchronized          disabled
  IPSEC Synchronized           disabled
  Iked Synchronized           disabled
  Lbpd Synchronized           disabled
  tCCD Synchronized           disabled
  Service Health Synchronized          true
  Active Peer Address        169.254.1.2

Redundancy Protocol Process (v3):
  State                      Standby
  Health                     100
  Lowest Local Address      169.254.1.1:9090
  1 peer(s) on 2 socket(s):
    OCSBC2-VMilanov: v3, Active, health=100, max silence=1050
                           last received from 169.254.1.2 on wancom1:0

  Switchover log:
  Feb 25 18:39:28.394: Active to RelinquishingActive, forced by command
  Feb 25 18:39:43.243: Standby to BecomingActive, forced by command
  Feb 25 18:40:19.874: Active to RelinquishingActive, forced by command
```

Фиг.5.16 - Резултат след "notify berpd force" и "show health"

Вижда се, че след ръчния switchover OCSBC1-VMilanov е в State "Standby", а OCSBC2-VMilanov е "Active". С команда "show health" се визуализират и журнални съобщения (logs) за състоянията на двете OCSBC. В моментите на switchover активният възел се прехвърля в състояние "RelinquishingActive", а възелът, който е бил "Standby" - в състояние "BecomingActive".

5.3. Реализиране на обаждане

5.3.1. Осъществяване на Layer 3 свързаност

Проверката за свързаност на ниво 3 от OSI модела (Network Layer) се осъществява чрез команда ping. Чрез въвеждане на команда "ifconfig" в Linux се получава информация за IP адресите на съответните портове на CentOS машината (Фиг.5.17 а) и б)).

a) ens224: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 10.10.0.10 netmask 255.255.255.0 broadcast 10.10.0.255

б) ens256: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 10.10.1.10 netmask 255.255.255.0 broadcast 10.10.1.255

Фиг.5.17 а) и б) - Информация за ens224 ens256 чрез въвеждане на ifconfig

След достъп до OCSBC е тествана свързаността посредством команда "ping" към двата интерфейса на виртуалната машина. Резултатите са показани на Фиг.5.18.

```
OCSBC1-VMilanov# ping 10.10.0.10
PING 10.10.0.10 from s0p0:1

44 bytes from 10.10.0.10: icmp_seq=1 ttl=64 time=3.819 ms
44 bytes from 10.10.0.10: icmp_seq=2 ttl=64 time=0.975 ms
44 bytes from 10.10.0.10: icmp_seq=3 ttl=64 time=0.915 ms
44 bytes from 10.10.0.10: icmp_seq=4 ttl=64 time=0.888 ms

4 packets transmitted, 4 received, 0% packet loss
OCSBC1-VMilanov# ping 10.10.1.10
PING 10.10.1.10 from s0p1:1

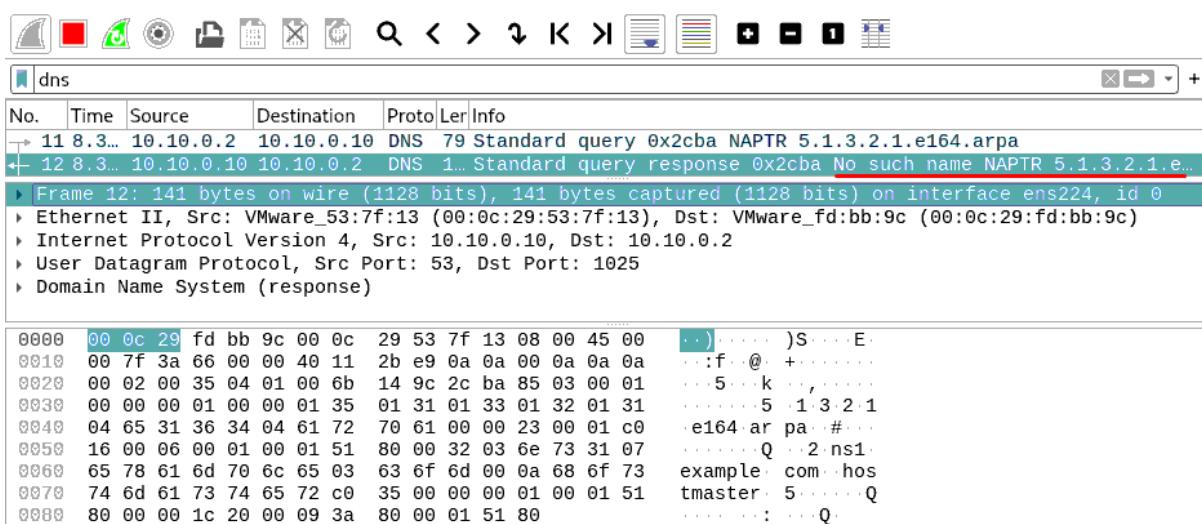
44 bytes from 10.10.1.10: icmp_seq=1 ttl=64 time=2.131 ms
44 bytes from 10.10.1.10: icmp_seq=2 ttl=64 time=1.913 ms
44 bytes from 10.10.1.10: icmp_seq=3 ttl=64 time=0.946 ms
44 bytes from 10.10.1.10: icmp_seq=4 ttl=64 time=0.939 ms

4 packets transmitted, 4 received, 0% packet loss
```

Фиг.5.18 - "ping" към 10.10.0.10 и към 10.10.1.10 от SBC

5.3.2. Последователност от стъпки за реализиране на обаждане

Първата стъпка за реализиране на обаждане е набирането на номер във формат E.164 в един от SIP клиентите. Набиране на какъвто и да е номер (дори и несъществуващ в базата данни) би довело до DNS NAPTR заявка от страна на OCSBC за установяване на домейна. Получената информация за несъществуващ в базата данни номер (+12315) е показана на Фиг.5.19.



Фиг.5.19 - Получена информация за несъществуващ запис

Заявката за съществуващ номер в базата данни (+6666) и получената информация за него е показана на Фиг.5.20 а) и б).

No.	Time	Source	Destination	Protocol	Info
	... 331.8...	10.10.0.2	10.10.0.10	DNS	77 Standard query 0x2564 NAPTR 6.6.6.6.e164.arpa
	... 331.8...	10.10.0.10	10.10.0.2	DNS	1.. Standard query response 0x2564 NAPTR 6.6.6.6.

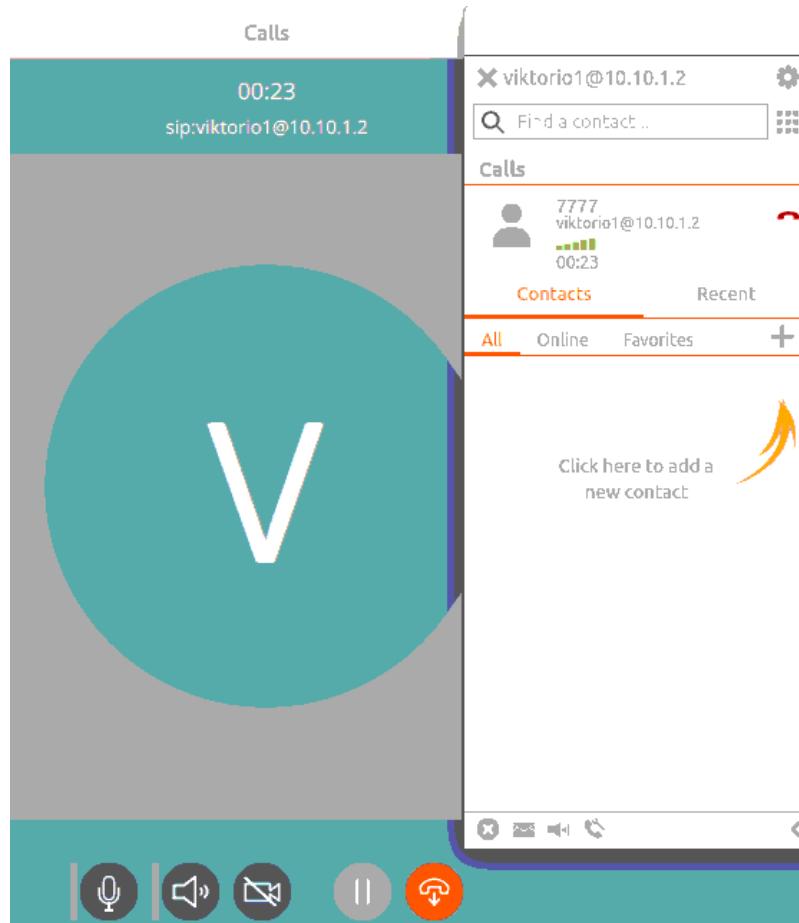
Фиг.5.20 а) - Запитване и отговор от DNS за номер +6666

```

.....6.6.6.6.e
164.arpa. #....
#....Q.. *..d...u.
E2U+sip. !^.*$!si
p:6666@1 0.10.1.1
0!.
  
```

Фиг.5.20 б) - Получената информация за съществуващия запис

Примерно обаждане между клиент "1" (отляво) и клиент "2" (отдясно), инициирано от "2" към номер +7777 (или просто 7777), е показано на *Фиг.5.21*.



Фиг.5.21 - Примерно обаждане към +7777 от страна на клиент "2"

Направено е проследяване на интерфейсите "ens224" и "ens256" едновременно, за да се прихванат всички пакети в разговор с +7777. Важно е да се следи и да се сортират резултатите по полето "Time", защото често пакетите са ресинхронизирани от Wireshark. Възможно е да се използва команда в терминала "reordercap", която преподрежда пакетите от даден файл и го записва в нов файл (*Фиг.5.22*).

```
[vmilanov@localhost Documents]$ reordercap capture1.pcapng capture2.pcapng  
1119 frames, 58 out of order
```

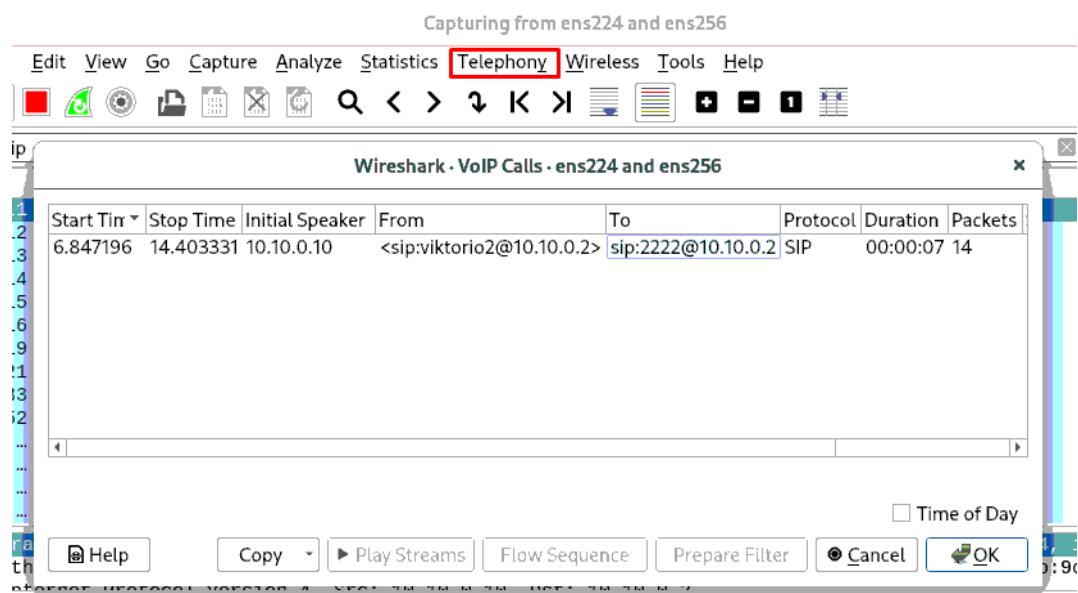
Фиг.5.22 - Команда за преподреждане на пакети от pcap/pcapng файл

Резултатите от преподреденото проследяване са илюстрирани на **Фиг.5.23.**

No.	Time	Source	Destination	Proto	Ler	Info
3	-0...	10.10.1.10	10.10.1.2	SIP	8...	Request: INVITE sip:7777@10.10.1.2;transport=UDP
4	-0...	10.10.1.2	10.10.1.10	SIP	3...	Status: 100 Trying
10.0.	10.10.0.2	10.10.0.10	SIP	8...	Request: INVITE sip:7777@10.10.0.10:5060	
20.0.	10.10.0.10	10.10.0.2	SIP	2...	Status: 100 Trying	
50.3.	10.10.0.10	10.10.0.2	SIP	4...	Status: 180 Ringing	
60.3.	10.10.1.2	10.10.1.10	SIP	4...	Status: 180 Ringing	
71.4.	10.10.0.10	10.10.0.2	SIP	8...	Status: 200 Ok	
121.4.	10.10.1.2	10.10.1.10	SIP	8...	Status: 200 Ok	
181.5.	10.10.1.10	10.10.1.2	SIP	4...	Request: ACK sip:10.10.1.2:5060;transport=udp	
111.5.	10.10.0.2	10.10.0.10	SIP	4...	Request: ACK sip:10.10.0.10;transport=udp	
...5.2.	10.10.0.10	10.10.0.2	SIP	4...	Request: BYE sip:viktorio1@10.10.0.2:5060;transport=udp	
...5.2.	10.10.1.2	10.10.1.10	SIP	4...	Request: BYE sip:viktorio1@10.10.1.10:5061;transport=UDP	
...5.2.	10.10.1.10	10.10.1.2	SIP	3...	Status: 200 OK	
...5.2.	10.10.0.2	10.10.0.10	SIP	3...	Status: 200 OK	

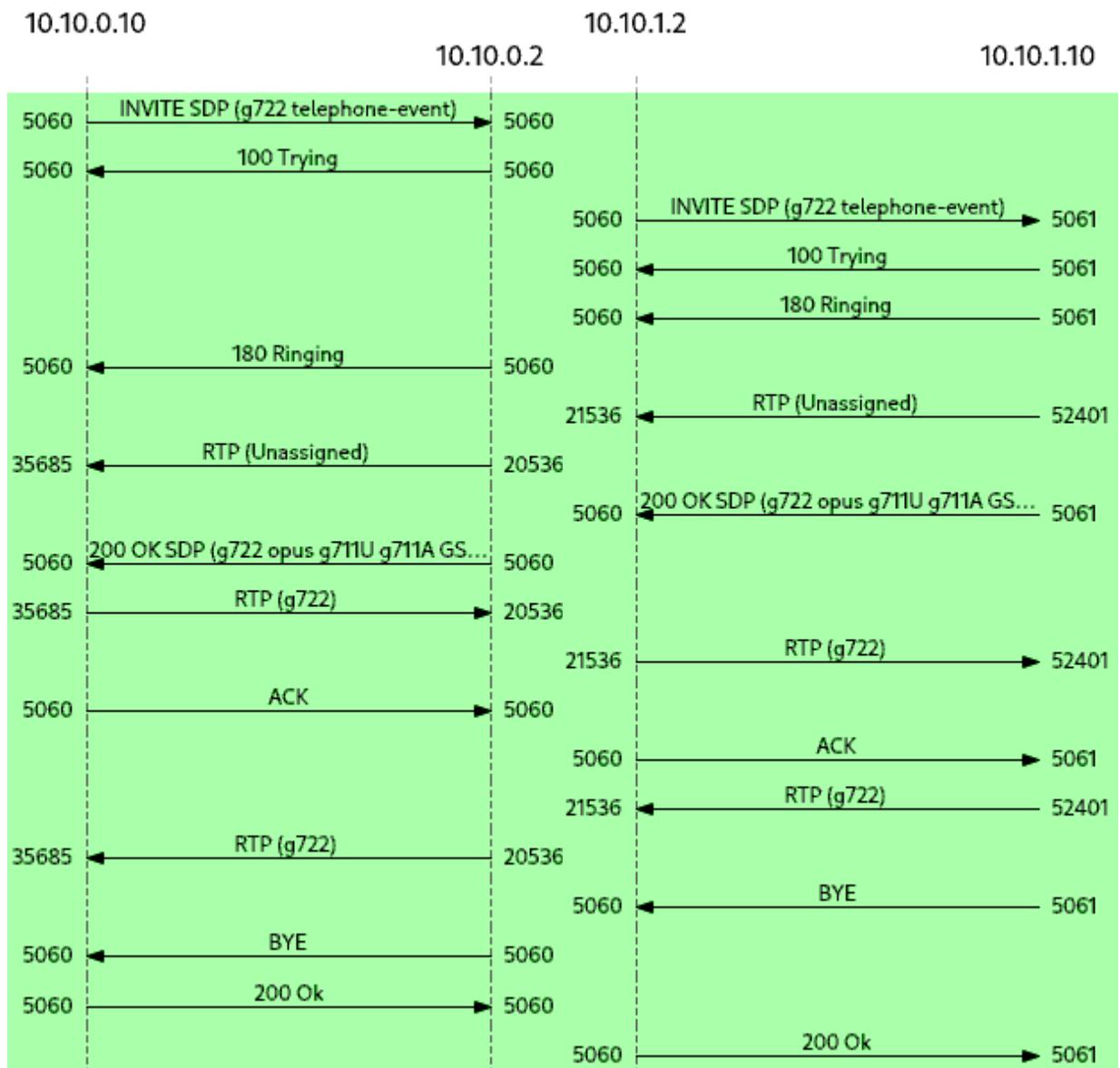
Фиг.5.23 - Проследяване на сигналния трафик в обаждане с филтър "sip" в Wireshark

Прозорецът за визуализиране на VoIP обаждания е показан на **Фиг.5.24** и се отваря от падащото меню "Telephony" и след това "VoIP Calls".



Фиг.5.24 - Прозорец за визуализация на обаждания

Визуализация на примерно обаждане към номер +2222, осъществено след преподреждане на пакетите чрез "reordercap", е показана на *Фиг.5.25*.



Фиг.5.25 - Визуализация на трафик от обаждане към 2222 от клиент "1"

В двета края на обаждането стоят двета SA, с адреси съответно 10.10.0.10 и 10.10.1.10. Между тях се намира OCSBC, което действа като B2BUA и е представено чрез своите интерфейси в съответните две мрежи - 10.10.0.2 за мрежата 10.10.0.0/24 и 10.10.1.2 за мрежата

10.10.1.0/24. За да не бъдат съобщенията с много голям обем е избран само един кодек - "G.722" и обаждането протича нормално. Показани са и портовете, на които всяка точка от обаждането комуникира за различните типове трафик в разговора.

5.3.3. Обаждане преди и след манипулация на Session Initiation Protocol хедър

За да се демонстрират разликите в поведението на OCSBC и съответно в данните от комуникацията преди и след манипулация на Session Initiation Protocol хедър, е използвано едно и също тестово обаждане от страна на клиент "1" към номер +6666. Проследен е UDP Stream чрез "Follow UDP Stream" от интерфейс ens256 и се наблюдават съобщения, изпратени от отдалечената мрежа (в случая 10.10.0.0/24) към SA. На практика това означава такива съобщения, които SBC проксира към SA.

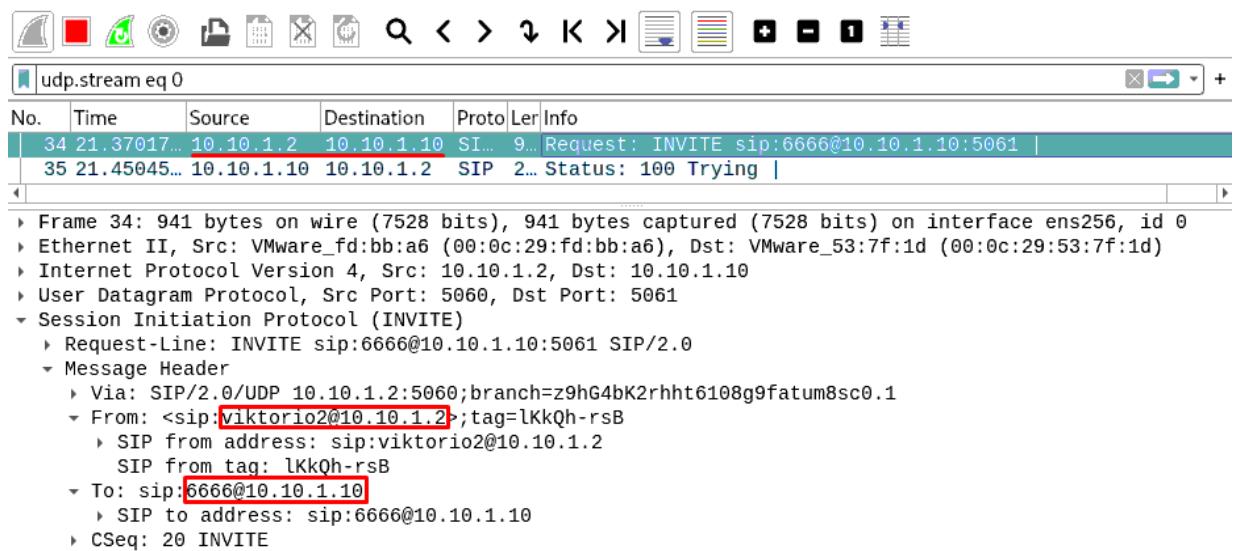
В първият случай, показан на Фиг.5.26 се вижда, че елементът "uri-host" и в двете полета от хедъра (и във From и в To) е със стойност IP адреса на интерфейса на SBC от отдалечената мрежа (10.10.0.0/24).

No.	Time	Source	Destination	Protocol	Length	Info
5	5.728307...	10.10.1.2	10.10.1.10	SIP	9...	Request: INVITE sip:6666@10.10.1.10:5061
6	5.808835...	10.10.1.10	10.10.1.2	SIP	2...	Status: 100 Trying

```
Frame 5: 940 bytes on wire (7520 bits), 940 bytes captured (7520 bits) on interface ens256, id 0
Ethernet II, Src: VMware_fd:bb:a6 (00:0c:29:fd:bb:a6), Dst: VMware_53:7f:1d (00:0c:29:53:7f:1d)
Internet Protocol Version 4, Src: 10.10.1.2, Dst: 10.10.1.10
User Datagram Protocol, Src Port: 5060, Dst Port: 5061
Session Initiation Protocol (INVITE)
  Request-Line: INVITE sip:6666@10.10.1.10:5061 SIP/2.0
  Message Header
    Via: SIP/2.0/UDP 10.10.1.2:5060;branch=z9hG4bK0bu97400d8o9fkkkdjio.1
    From: <sip:viktorio2@10.10.0.2>;tag=2WDnD-Y98
      SIP from address: sip:viktorio2@10.10.0.2
      SIP from tag: 2WDnD-Y98
    To: sip:6666@10.10.0.2
      SIP to address: sip:6666@10.10.0.2
    CSeq: 20 INVITE
```

Фиг.5.26 - Обаждане към +6666 без HMR - адресите не са променени

На Фиг.5.27 е илюстриран трафика след HMR и се вижда, че елементът "uri-host" от полето "From" заема стойност "\$LOCAL_IP", тоест IP адреса на SBC от мрежата 10.10.1.0/24, а същият елемент в полето "To" заема стойност "\$REMOTE_IP" - IP адреса на съответния SA (в примера това е адреса 10.10.1.10). По този начин е приложено условието за скриване на топологията на отсъщната страна - при всеки един отговор от страна на OCSBC (защото параметърът в HMR "msg-type" е със стойност "any") се модифицират адресите от отдалечената мрежа. Така всеки SA знае единствено за SBC в рамките на едно обаждане и за него разговорът се осъществява единствено със SBC.



Фиг.5.27 - Обаждане към 6666, този път след HMR - променени от

OCSBC адреси

Заключение

В настоящата дипломна работа е описано и реализирано технологично решение, свързващо два VoIP доставчика чрез локални политики и външен ENUM сървър, позволяващо гласова връзка между доставчици на услуги.

Основните постижения на дипломната работа са описани по-долу. В първа глава от разработката са разгледани и описани основните протоколи, устройства и понятия, свързани с VoIP мрежите, както и техния начин на функциониране. Втора глава е свързана с преглед на Session Border Controller устройства, като специално внимание е отделено на Oracle Communications SBC. Трета глава е свързана с изследване на важните технологични особености на решението - записи от тип NAPTR в DNS, ENUM, заедно с локални политики и правила за манипулация на SIP хедъри. Четвърта и пета глава са насочени към практическата част на дипломната работа. В четвърта глава са описани основните концепции при реализирането на конфигурацията при OCSBC и PowerDNS. Пета глава представя резултатите от множество проведени тестове и проверки на разработеното решение, проведени с цел доказване на неговата работоспособност.

Въз основа на всичко, описано по-горе, може да се направи извод, че заданието и целта на дипломната работа са изцяло изпълнени. Подробно са описани технологиите и протоколите, върху които се изгражда комуникацията в дипломната работа. Направени са необходимите тестове за доказателство на постигнатата свързаност чрез външен ENUM сървър между доставчици на услуги.

Бъдещото развитие на дипломната работа би могло да включва следното:

- Добавяне на допълнително оборудване към мрежата, като например Registrar или Location сървър;

- Поектиране на трето пространство за вече създадения DNS сървър и за допълнителното оборудване, което би позволило на тези устройства да бъдат установени отделно от пространствата на доставчиците;
- Имплементиране на графичен интерфейс за визуализиране, добавяне и редактиране на записи и таблици в DNS. Пример за това би могъл да е PowerAdmin.

Използвани съкращения

A - Address

ACLI - Acme Command Line Interface

AOR - Address Of Record

B2BUA - Back-to-Back User Agent

BIND - Berkeley Internet Name Daemon

CCaaS - Contact Center as a Service

CH - Chaosnet

CLI - Command Line Interface

CNAME - Canonical NAME

CSRC - Contributing Source

DDoS - Distributed Denial of Service

DNS - Domain Name System

DOS - Disk Operating System

DoS - Denial of Service

E-SBC, e-SBC - Enterprise Session Border Controller

E-VSBC, e-vSBC - Enterprise Virtual Session Border Controller

ENUM - E.164 Number Mapping

ERE - Extended Regular Expression

FQDN - Fully Qualified Domain Name

HA - High Availability

HD - High Definition

HMR - Header Manipulation Rules

HS - Hesiod

HTTP - HyperText Transfer Protocol

HTTPS - HyperText Transfer Protocol Secure

IANA - Internet Assigned Numbers Authority

ID - Identifier

IETF - Internet Engineering Task Force

IN - Internet
IP - Internet protocol
IPv4 - Internet protocol version 4
IPv6 - Internet protocol version 6
ISO - International Organization for Standardization
ITU - International Telecommunications Union
LCR - Least Cost Routing
LP - Local Policy
MX - Mail Exchange
NAPTR - Naming Authority Pointer
NS - Name Server
NTP - Network Time Protocol
OCSBC - Oracle Communications Session Border Controller
OSI - Open Systems Interconnection
PDNS, pdns - PowerDNS
PSTN - Public Switched Telephone Network
PTR - Pointer
QoS - Quality of Service
RR - Receiver Report
RTCP - Real-time Transport Control Protocol
RTP - Real-time Transport Protocol
RTP/AVP - Real-time Transport Protocol Audio/Video Profile
RTP/SAVP - Real-time Transport Protocol Secure Audio/Video Profile
RTSP - Real-Time Streaming Protocol
SA - Session Agent
SBC - Session Border Controller
SDES - Source Description
SDP - Session Description Protocol
SIP - Session Initiation Protocol
SO - Small Office

SOA - Start of Authority
SQL - Structured Query Language
SR - Sender Report
SRTP - Secure Real-time Transport Protocol
SRV - Service Locator
SSH - Secure Shell
SSRC - Synchronization Source
TCP - Transmission Control Protocol
TLD - Top-Level Domain
TLS - Transport Layer Security
TTL - Time To Live
TXT - Text
UA - User Agent
UAC - User Agent Client
UAS - User Agent Server
UCaaS - Unified Communications as a Service
UDP - User Datagram Protocol
URI - Uniform Resource Identifier
UTC - Coordinated Universal Time
VLAN - Virtual Local Area Network
VoIP - Voice over IP
XML - Extensible Markup Language

Използвана литература

- [1]. <https://www.wpbeginner.com/beginners-guide/beginners-guide-what-is-voip-and-how-does-it-work-explained/>, последно посетен 29.02.2024
- [2]. <https://www.nextiva.com/blog/what-is-a-voip-phone.html>, последно посетен 29.02.2024
- [3]. <https://www.metaswitch.com/knowledge-center/reference/what-is-session-initiation-protocol-sip>, последно посетен 29.02.2024
- [4]. <https://trueconf.com/blog/reviews-comparisons/why-sip-better-than-h323.html>, последно посетен 29.02.2024
- [5]. <https://www.nextiva.com/blog/sip-protocol.html>, последно посетен 29.02.2024
- [6]. https://www.tutorialspoint.com/session_initiation_protocol/index.htm, последно посетен 29.02.2024
- [7]. https://andrewjprokop.wordpress.com/2014/03/24/understanding_sip-addresses/, последно посетен 29.02.2024
- [8]. <https://datatracker.ietf.org/doc/html/rfc3261#section-7.1>, последно посетен 29.02.2024
- [9]. <https://kb.iu.edu/d/aiuv>, последно посетен 29.02.2024
- [10]. https://docs.oracle.com/cd/E13209_01/wlcp/wlss30/programming_compactform.html, последно посетен 29.02.2024
- [11]. <https://www.3cx.com/pbx/sip-responses/>, последно посетен 29.02.2024
- [12]. https://andrewjprokop.wordpress.com/2013/09/30/understanding_session-description-protocol-sdp/, последно посетен 29.02.2024
- [13]. <https://datatracker.ietf.org/doc/html/rfc8866>, последно посетен 29.02.2024
- [14]. https://www.tutorialspoint.com/session_initiation_protocol/session_initiation_protocol_sdp.htm, последно посетен 29.02.2024
- [15]. <https://www.tutorialspoint.com/real-time-transport-protocol-rtp>, последно посетен 29.02.2024

- [16]. <https://datatracker.ietf.org/doc/html/rfc3550>, последно посетен 29.02.2024
- [17]. <https://getvoip.com/blog/voip-codecs/>, последно посетен 29.02.2024
- [18]. <https://ribboncommunications.com/company/get-help/glossary/session-border-controller-sbc>, последно посетен 29.02.2024
- [19]. <https://www.metaswitch.com/knowledge-center/reference/what-is-a-session-border-controller-sbc>, последно посетен 29.02.2024
- [20]. <https://www.oracle.com/a/ocom/docs/industries/communications/enterprise-session-border-controller-ds.pdf>, последно посетен 29.02.2024
- [21]. <https://www.fortinet.com/resources/cyberglossary/what-is-dns>, последно посетен 29.02.2024
- [22]. <https://www.cloudflare.com/learning/dns/what-is-dns/>, последно посетен 29.02.2024
- [23]. <https://www.cloudflare.com/learning/dns/dns-records/>, последно посетен 29.02.2024
- [24]. <https://www.cloudns.net/wiki/article/189/>, последно посетен 29.02.2024
- [25]. <https://www.dynu.com/Resources/DNS-Records/NAPTR-Record>, последно посетен 29.02.2024
- [26]. <https://arstechnica.com/information-technology/2010/01/enum-dragging-telephone-numbers-into-the-internet-age/>, последно посетен 29.02.2024
- [27]. <https://www.networkworld.com/article/883692/lan-wan-what-is-enum.html>, последно посетен 29.02.2024
- [28]. <https://datatracker.ietf.org/doc/html/rfc2916>, последно посетен 29.02.2024
- [29]. <https://www.3cx.com/pbx/what-does-enum-mean/>, последно посетен 29.02.2024
- [30]. <https://voipmagazine.wordpress.com/tag/e164-arpa/>, последно посетен 29.02.2024

- [31]. <https://www.cloudns.net/wiki/article/21/>, последно посетен 29.02.2024
- [32]. <https://docs.oracle.com/en/industries/communications/session-border-controller/9.1.0/configuration/sbc-configuration-guide.pdf>, последно посетен 29.02.2024
- [33]. <https://docs.oracle.com/en/industries/communications/session-border-controller/9.2.0/hmr/hmr-guide.pdf>, последно посетен 29.02.2024
- [34]. https://docs.oracle.com/cd/E95618_01/doc/sbc_scz810_aclreference.pdf, последно посетен 29.02.2024
- [35]. <https://docs.oracle.com/en/industries/communications/session-border-controller/9.0.0/aclireference/index.html#Oracle%C2%AE-Communications-Session-Border-Controller>, последно посетен 29.02.2024
- [36]. <https://www.liquidweb.com/kb/what-is-power-dns/>, последно посетен 29.02.2024
- [37]. <https://aws.amazon.com/compare/the-difference-between-mariadb-vs-mysql/>, последно посетен 29.02.2024
- [38]. <https://softuni.bg/blog/relational-databases-applications>, последно посетен 29.02.2024
- [39]. <https://computingforgeeks.com/install-powerdns-on-centos-with-powerdns-admin/>, последно посетен 29.02.2024
- [40]. <https://www.cloudflare.com/learning/dns/dns-records/dns-soa-records/>, последно посетен 29.02.2024
- [41]. <https://docs.oracle.com/en/industries/communications/session-border-controller/9.0.0/aclireference/enum-config.html#GUID-8BDFF8E4-FB8D-4C89-93ED-EE7353A0250D>, последно посетен 29.02.2024