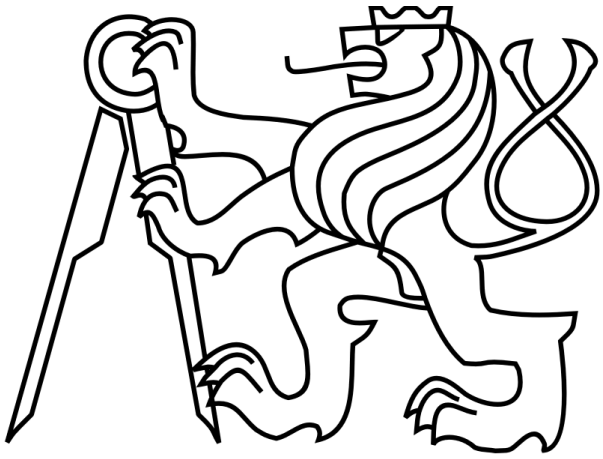
České vysoké učení technické v Praze

fakulta elektrotechnická

katedra telekomunikační techniky



Diplomová práce

Pokročilé komponenty laboratorní sítě IMS

Advanced components of the laboratory IMS network

Autor: **Evgeny Khalafyan**

Vedoucí práce: Ing. Pavel Troller, CSc.

2017

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

ZADÁNÍ DIPLOMOVÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Khalafyan** Jméno: **Evgeny** Osobní číslo: **399374**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra telekomunikační techniky**

Studijní program: **Komunikace, multimédia a elektronika**

Studijní obor: **Komunikační systémy**

II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

Pokročilé komponenty laboratorní IMS sítě

Název diplomové práce anglicky:

Advanced compoents of the laboratory IMS network

Pokyny pro vypracování:

Proveďte průzkum dostupnosti otevřených řešení pokročilých komponent laboratorní IMS sítě (TAS, MRF, SBC, DIAMETER server atd.). Jednu z komponent vyberte a integrujte do laboratorní IMS sítě na bázi projektu Kamailio/IMS.

Seznam doporučené literatury:

[1] Camarillo, G.; García-Martín, M.A.: The 3G IP multimedia subsystem (IMS) : Merging the Internet and the Cellular Worlds (2 ed.). Chichester [u.a.]: Wiley 2007. ISBN 0-470-01818-6.
[2] Syed A. Ahson, M. I.: IP multimedia subsystem (IMS) handbook. Boca Raton: CRC Press 2009. ISBN 1-4200-6459-2.

Jméno a pracoviště vedoucí(ho) diplomové práce:

Ing. Pavel Troller CSc., katedra telekomunikační techniky FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **16.02.2017** Termín odevzdání diplomové práce: **26.05.2017**

Platnost zadání diplomové práce: **30.09.2018**

Podpis vedoucí(ho) práce | Podpis vedoucí(ho) ústavu/katedry | Podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

10. 5. 17

Datum převzetí zadání | Podpis studenta

Čestné prohlášení

Prohlašuji, že jsem zadanou diplomovou práci „Pokročilé komponenty laboratorní sítě IMS“ zpracoval sám s přispěním vedoucího práce a používal jsem pouze literaturu uvedenou na konci práce. Souhlasím se zapůjčováním práce a jejím zveřejňováním.

podpis

V Praze dne Evgeny Khalafyan

Poděkování

Především bych chtěl poděkovat svému vedoucímu Ing. Pavlu Trollerovi, CSc. za příležitost pracovat na instalaci a konfiguraci zakladu IMS sítí a SBC, pomoc při realizace, cenné rady a možnosti se naučit hodně nových věci. Dále bych chtěl poděkovat svým rodičům a příbuzným za jejich podporu.

Abstrakt

V dnešní době přenášení hlasu prostřednictvím paketů již začalo pronikat do světa mobilních technologií. Platformou, která to umožňuje, je IP Multimedia Subsystem.

Náplní této diplomové práce je problematika IMS sítí. Bude tady stručně uvedeno, co představuje IMS síť, její funkce, protokoly, podle kterých funguje, základní a některé z doplňujících prvků. Cílem další části je zprovoznění virtuálních serveru, správné propojeni je mezi sebou, následné instalovaní Kamailio/IMS a SBC, jejich konfigurace a další testovaní.

Klíčová slova

IMS, SBC, Volte, SIP.

Abstract

Nowadays, the voice transmission through the packages has already begun to penetrate the world of mobile technology. The platform that makes this possible is the IP Multimedia Subsystem.

This diploma thesis deals with IMS networks. It will briefly outline what the IMS network is, its functions, protocols according to which it works, basic and some of the complementary elements. The aim of the next part is to put the virtual servers into operation, then their correct connection, and installation of Kamailio/IMS and SBC, to configure them and to test them.

Key words

IMS, SBC, Volte, SIP.

Obsah

1. Úvod

S každým rokem se víc a víc zvětšuje důležitost rozvoje a podpory IMS sítí. Je to způsobeno tím, že sítí starého formátu už v blízkém období nebudou schopné splnit požadavky rychle rozvíjejícího telekomunikačního trhu. Abychom pochopili, proč se to děje právě teď, ale ne například před 8 lety, a to navzdory tomu, že popis IMS už existoval v tu dobu je potřeba označit klíčová kritéria, podle kterých paketový přenos hlasu v tu dob nebyl možný. Jedná se o rychlost a latenci v paketovém přenosu hlasu. V sítích generaci 3G a starších, vysoké přenosové zpoždění bránilo zavedení nových komunikačních standardů. Například pokud účastník odesílá "Ahoj", komunikační systém rozdělí zprávu na pakety a pošle je odděleně. Některé pakety se mohou zpožďovat a některé se mohou ztratit. Pak přijímací strana se pokusí dat to, co dostane dohromady. Při ztrátě a zpoždění, vše, co účastník na druhém konci uslyší bude například "A" a "oj". Nicméně, všechno se prudce změnilo s příchodem LTE. Kromě toho ze rychlosti přijmu a odesíláni paketů se výrazně zvětšili, zpoždění jejich přenosu prudce kleslo. Na tolik mala latence už umožnovala zavedení přenosu hlasových paketů. Vzhledem k tomu, že nebylo možné za okamžik převést 100 procent účastníků na novy komunikační standard přenosu, bylo nutné najit takovou platformu, která by podporovala jak standardní sítě, tak i sítí předchozích generací. Jedním z takových řešeni je IMS, které se dnes aktivně zavadí.

Tato diplomová práce musí být věnovaná pokročilým komponentům IMS sítě. Nicméně s ohledem na to že v rámci jedné diplomové práci množství komponent sítí můžou být prozkoumané jenom povrchně námi byl odůvodněně vybrán pouze jeden komponent sítí, a to je SBC (Session Border Controller). V této práci budou uvedené definice sítě IMS, většina jejich bodů, jakož i protokolům podle kterých body mezi sebou komunikuji. Dále se zaměříme na implementaci IMS a SBC na základě

open source zdrojů, během prací námi budou taky předvedeny procesy instalaci, nastaveni a troubleshootingu problémů se kterými se setkáme. Poté, jak budou IMS sít‘ a SBC dostatečně vyladěné a nakonfigurované, budou provedené nezbytné testy, aby byly určený možnosti a funkčnost SBC. Nakonec uděláme závěr provedené prací.

2. IMS síť

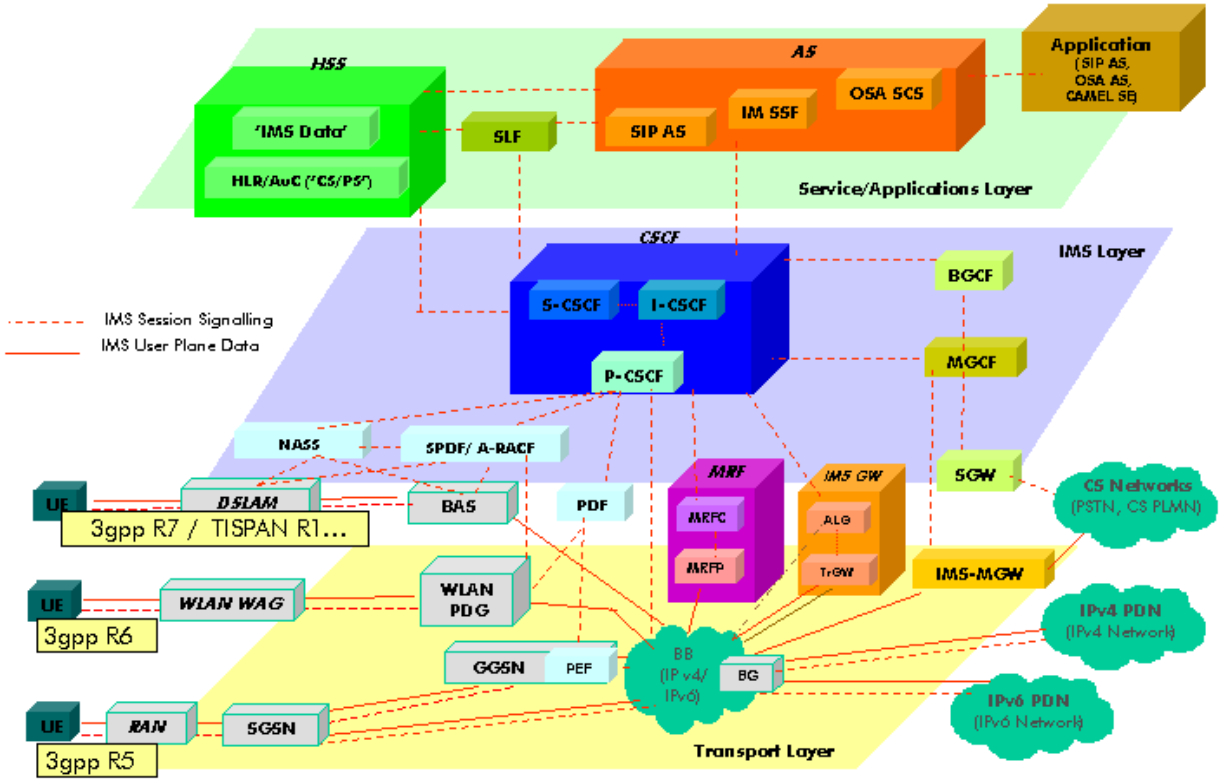
Poněvadž celé povídaní je věnováno IMS sítí a její komponentům povazuji za nutnost seznámeni s tím co je IMS platforma, k čemu slouží a jaké má základní komponenty.

2.1 Historie

Zpočátku myšlenka vybudování mobilní sítě založené na IP technologiích byla obsazena skupinou 3G.IP. Právě ona vyvinula technologii GPRS, která následně zahájila vývoj architektury IP sítí. Následně byla sestavena pracovní skupina 3GPP, která v roce 2001 představila verzi 4 (původně nazvanou Release 2000), ve které se objevily prvky architektury ALL-IP. Páté vydání představilo počáteční verzi architektury s názvem IMS a byla přidaná technologie vysokorychlostního přenosu paketových dat (HSDPA). V šestém vydání byly provedeny změny architektury IMS a byla zavedena podpora pro bezdrátové sítě LAN. Sedmé vydání 3GPP přidalo podporu pevných sítí (PSTN). V roce 2013 jednou z nejdůležitějších aplikací IMS byla podpora plnohodnotné hlasové komunikační technologie v sítích LTE (VoLTE). V dnešní době technologie VoLTE už se začala aktivně rozvíjet na telekomunikačním trhu.

Jako hlavni protokol pro zahájeni relace byl vybrán SIP. Důležitou vlastností SIP je rozšiřitelnost, což je možnost přidat nové funkce do protokolu přidáním nových záhlaví a zpráv, což umožnuje přidáváni do sítě nových funkcí bez změny protokolu.

2.2 Architektura sítě



Obr.1: Architektura sítě [1]

2.3 Základní prvky

V IMS vrstvě, která da se říct je hlavní vrstvou sítí leží jeden z klíčových bodu a to je CSCF(Call Session Control Function), který se pak děli na tři části a to jsou:

- S-CSCF – Session-Call Session Control Function
- P-CSCF – Proxy Call-Session Control Function
- I-CSCF – Interrogating Call Session Control Function

Jedna nebo více z těchto funkcí může být hostované fyzickým síťovým uzlem v doméně IMS sítě.

Jak je vidět CSCF je sbírka funkcí, která hraje zásadní roli v jádře síti IMS. CSCF je zodpovědný za kontrolu signalizace komunikace uživatelského vybavení IMS (UE – user equipment) s rozšířenými službami IMS v různých síťových přístupech a doménách. CSCF řídí zřízení relace a její ukončeni, stejně jako autentizaci uživatelů, zabezpečení sítě a QoS (Quality of Service).

Dále se podrobněji podíváme na části CSCF.

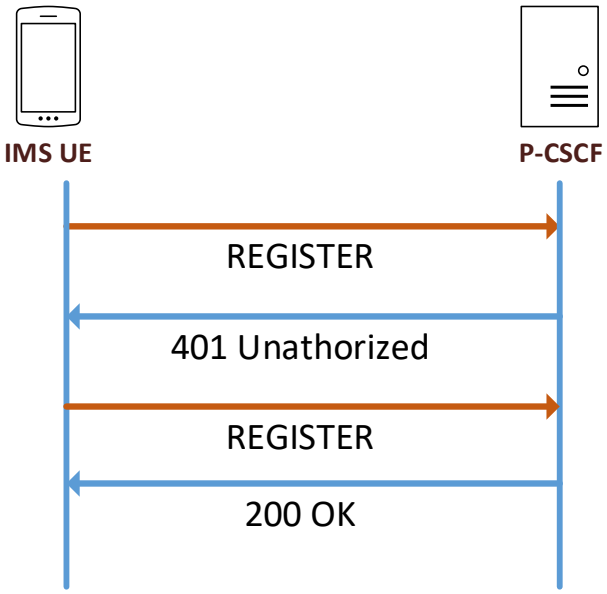
2.3.1 P-CSCF

Proxy-CSCF (P-CSCF) je SIP proxy, který je prvním kontaktním bodem v IMS sítí. Může se nacházet buď v síti (v plné síti IMS) nebo v domácí síti (pokud síť dosud ta sít není kompatibilní s IMS). Připojeni k P-CSCF je pro terminál nezbytné pro zahájeni registrací v IMS sítí. Aby došlo ke spojeni účastník provádí proceduru pro vyhledávaní P-CSCF pomoci DNS dotazu. Po zjištění P-CSCF může UE odeslat požadavek SIP REGISTER, aby se zaregistroval do jádrové sítě IMS.

Funkce P-CSCF:

- Před registrací je přiřazen k terminálu IMS a po celou dobu registrace se nezmění.
- Sedí na cestě veškeré signalizace a může zkontrolovat každý signál; terminál IMS musí ignorovat jakoukoli jinou nešifrovanou signalizaci. IMS musí ignorovat jakoukoli jinou nešifrovanou signalizaci.

Poskytuje ověření odběratele a může vytvořit asociaci zabezpečení IPsec nebo TLS s IMS terminálem. Tím se zabrání útokům spoofingu a útokům typu replay a chrání se soukromí účastníka.). Logické kanály SA jsou vždy jednosměrné a vytváří se jak na straně jak účastníka tak i na straně P-CSCF, tudíž celkem se vytváří 4 kanály. Nastavovaní bezpečnostní se uskutečni během two-way handshake:



Obr.2: two-way handshake

- Může komprimovat a dekomprimovat SIP zprávy pomocí SigComp (Signaling Compression), čímž usnadňuje cestu přes pomalé linky.

2.3.2 I-CSCF

I-CSCF je zprostředkovatel pro interakci s externími sítěmi. Jeho adresa IP je publikována v Domain Name System (DNS) domény, aby jej mohly najít vzdálené servery a použít je jako bod pro předávání paketů SIP do této domény.

Funkce I-CSCF:

- Jednou z hlavních funkcí Interrogating-CSCF je zjišťovaní adres dalších entit nutných pro uskutečnění spoje, vzdálené body I-CSCF taky vidí a můžou požívat jako bod pro předávaní SIP paketů. Dotazy na nutná data uživatele I-CSCF dostava od P-CSCF.
- Dotazuje HSS na získání adresy S-CSCF a přiřazení k uživateli provádějícímu SIP registraci.
- Až do vydání verze 6 se mohl také používat k ukrytí interní sítě z vnějšího světa, avšak tyto funkce teď může mít na sebe SBC (Session Border Controller).

2.3.3 S-CSCF

Serving CSCF je centrálním uzlem sítě IMS, zpracovává všechny SIP zprávy mezi koncovými body. Při první registraci se obrati na HSS, stáhne profil uživatele a ověří identitu UE. Není jen SIP serverem, ale má za sebou i kontrolu relace. Vždy se nachází v domácí síti.

Funkce S-CSCF:

- Sedí na cestě všech signalizačních zpráv místně registrovaných uživatelů a může zkontrolovat každou zprávu.
- Rozhoduje, na který aplikační servery bude SIP zpráva předána, aby poskytli své služby.
- Poskytuje routovaci služby, obvykle využívající při vyhledávání elektronického číslování (ENUM).
- Zajišťuje dodržování politiky provozovatele sítě (operátoru).

2.4 Doplňující prvky

2.4.1 HSS

Home subscriber server sice neleží v IMS vrstvě, ale je jedním ze základních bodu sítí. Jedna se o hlavni databázi, na které jsou uložené veškera data tykající konkrétního uživatele. V případě, že je v síti IMS použito více serverů HSS, je nutné přidat funkce SLF (Subscriber Locator Function), která vyhledává HSS s údaji konkrétního uživatele.

HSS podporuje sítové entity cele sítí IMS zprostředkováním nutných pro uskutečnění hovoru data. Provádí autentifikace a autorizace uživatele, obsahuje taky informace o poloze UE a obsahuje taky všechny prvky pro každou předchozí technologii. Třeba pro GSM slouží jako HLR (Home Location Register) a AuC (Authentication Center). S IMS mohou byt spojeny různé identity uživatele.

2.4.2 MRF

Podobné funkce jako manipulace s médii (např. Mixování hlasových přenosů) a přehrávání tónů a hlasových oznámení schopni dělat většina B2B agentů jako třeba Asterisk, a nepotřebuje k tomu jednotlivý prvek, protože veškera komunikace mezi účastnici jde přes PBX. IMS ale na to má speciální prvek, a to je MRF (Media Resource Function)

Každý MRF je dále rozdělen na media resource function controller (MRFC) a media resource function processor (MRFP).

MRFC (Media Resource Function Controller) je signalizační prvek, který přenáší informace z AS (Application Server) a S-CSCF pro řízení MRFP

MRFP (Media Resource Function Processor) je mediální rovina používaná pro mixování nebo zpracování mediálních toků. Může také spravovat přístup ke sdíleným zdrojům.

2.4.3 SBC (Session Border Controller)

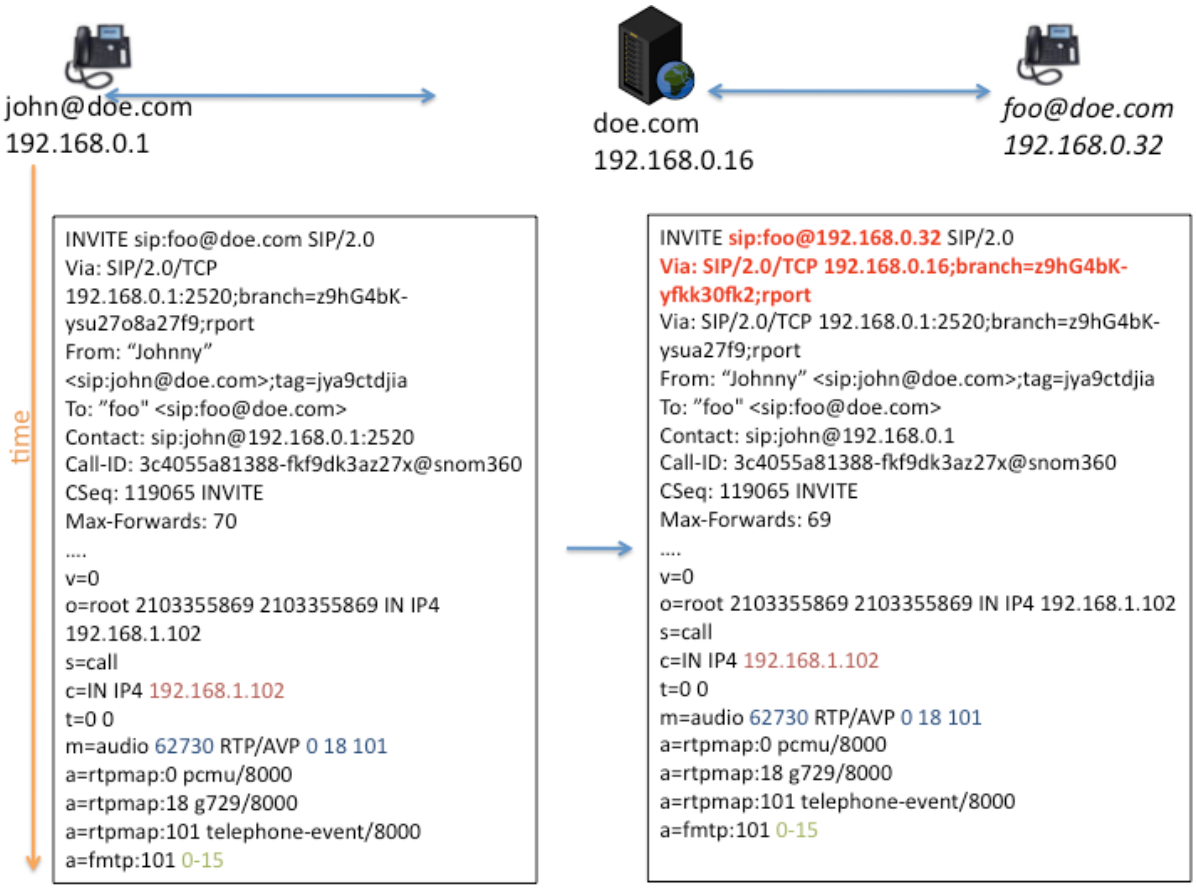
SBC se nachází na hranici sítě poskytovatele služeb a provádí následující funkci: translace protokolů signalizaci, analýza kvality mediálních kanálů, na kterých je směrován hlasový přenos (parametry, jako je zpoždění, jitter, procent ztráty paketů a tak dále), zajištění kvality služby, je stanovená v SLA (dohoda o úrovni služeb English. - service level Agreement), sběr statistických informací, kontrolu RTP-provozu a další. SBC je jediný vstupní a výstupní bod v síti operátora to znamená že se nachází ještě před P-CSCF, čímž se skrývá topologie sítě, zvyšuje její spolehlivost a odolnost proti DoS útokům, zjednoduší konfigurace a správu administrací.

Aby byla zajištěna vysoká úroveň spolehlivosti, Session Border Controller musí podporovat šifrování signálních dat (například SIPS a SIP přes TLS) a přenos media (SRTP, ZRTP).

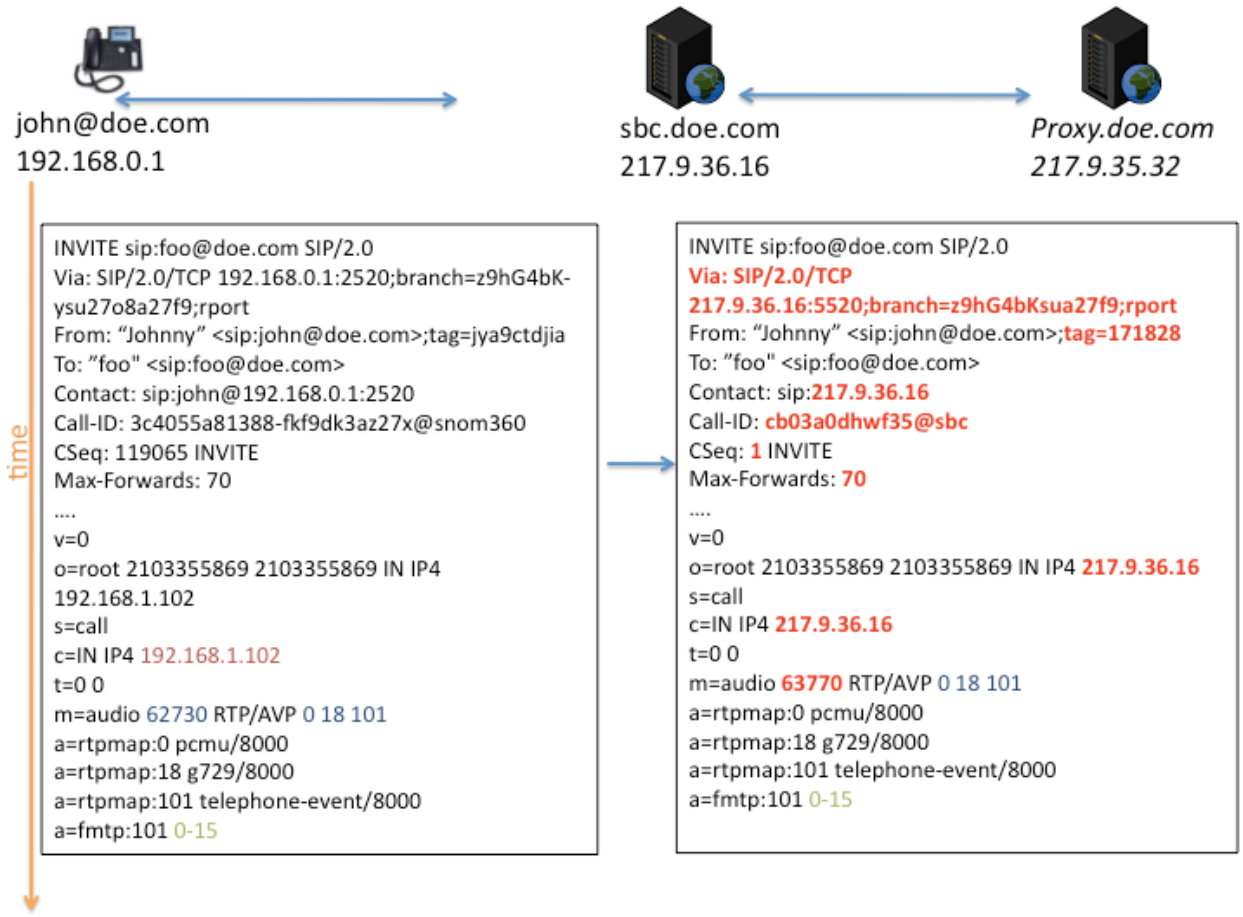
Dnes podle doporučeni pracovní skupiny TISPAN (Telecommunications and Internet converged Services and Protocols for Advanced Networking) SBC by mel byt integrován do P-CSCF,

nicméně tyto návrhy dosud platné nejsou, a ve všech reálných sítích SBC existuje jako jednotlivý prvek, přes který do jdou všechny zprávy směrem tam a ven.

Na dalších obrazcích je vidět rozdíl mezi normálním voláním SIP a voláním přes SBC.



Obr.3: SIP volání bez SBC [6]



Obr.4: SIP volání s SBC [7]

Jak vidíme v případě použiti SBC, protilehlá strana už je skryta, a Max-Forwards (podoba ttl v SIPu) je znova 70, ale ne 69, to znamená že SBC jako hraniční bod sítí se chápe jako prvotní bod, ze kterého se posila správa.

2.5 Identity uživatele v IMS

2.5.1 IP Multimedia Private Identity (IMPI)

IP Multimediální soukromá identita (IMPI) je jedinečná trvale přidělená globální identita přiřazená provozovatelem domácí sítě, má následující tvar- user.name@domain a používá se například pro registraci, autorizace, řízeni a účetní účely. Každý uživatel služby IMS má jeden IMPI.

2.5.2 IP Multimedia Public Identity (IMPU)

Veřejná identita IP multimédií (IMPU) používá jakýkoli uživatel pro vyžádání komunikace s ostatními uživateli. Pro jeden IMPI může být více IMPU. IMPU lze také sdílet s jiným telefonem, takže lze obojí dosáhnout pomoci stejné identity.

2.5.3 Globally Routable User Agent URI (GRUU)

Je identita, která identifikuje unikátní kombinaci IMPU a UE. Existují dva typy GRUU: Public-GRUU (P-GRUU) a Temprorary-GRUU (T-GRUU).

2.5.4 Wildcarded Public User Identity

Zástupná veřejná uživatelská identita vyjadřuje soubor IMPU seskupených dohromady.

Kromě popsaných identit do HSS se taky ukládají IMSI a MSISDN, tyto identity se vztahuji k mobilním sítím.

2.6 Rozhraní (Referenční body)

Když se mluvo o rozhraních IMS sítí, není míněno o fyzických zařízeních. Slouží vsak pro popis komunikaci a procesu probíhajících mezi jednotlivými body sítě. Nebo taky pro popis toho, kam vede signalizace.

2.6.1 Gm

Tento bod se používá k výměně zpráv mezi uživatelským zařízením SIP (UE) nebo VoIP a P-CSCF. Kromě signalizace přes inervace Gm se prochází i media.

2.6.2 Cx

Používá se pro odesílání dat účastníka z HSS do S-CSCF včetně kritérií filtrů a jejich priority. Pro přenos se používá protokol Diameter.

2.6.3 ISC

Bod mezi S-CSCF a aplikačními servery. Zaprvé, oznamuje AS (Application server) o registrovaném IMPU (IP multimedia public identity), možnostech UE a registračním stavu. Poskytuje AS data nutné pro poskytování vice služeb (multiple services).

2.6.4 Ma

Tento bod se používá pro směrovaní některých procesů, které nepotřebuji spolupracovat s S-CSCF, ale můžou být poslány přímo na AS server.

2.6.5 Cr

Požívá se MRFC pro dat (například skriptů, upozorněni a dalších zdrojů) z AS. Používá se taky pro příkazy tykající řízení mediu.

2.6.6 Dx

V případě sítí, ve kterých jsou dva nebo vice HSS tento bod je využít I-CSCF nebo C-CSCF pro nalezeni správní databázi.

2.6.7 Mg

Používá se prý převedeni ISUP signalizace na signalizace SIP a předávaní SIP zprav do I-CSCF.

2.6.8 Mm

Využívá se při výměně zprav mezi IMS a externími IP sítí.

2.7 Protokoly

2.7.1 Protokol SIP (Session Initiation Protocol)

Jelikož velká část komunikace mezi body v IMS sítí probíhá podle SIP protokolu myslím se ze bude nezbytné popsat jeho principy aspoň stručně.

Jak plyne z nazvu je to protokol, který je zodpovědný za inicializaci relaci. Ten protokol popisuje, jak se navazuje, probíhá a ukončuje relace mezi uživateli, nebo jinak řečeno je zodpovědný za signalizace a žádný hlas či jiná media sám o sobě nepřenáší. Pro přenos medii se spolu se SIPem využívá protokol RTP (Real-time Transport Protokol). Dá se říct, že má za úkol podobné věci jako soubor protokolu SS7 v obyčejné PSTN sítí. Pro přenos signalizace se obvykle používá UDP na portu 5060, můžou být použité taky TCP a ATM protokoly. Pro zabezpečeni lze využít TLS (Transport Layer Security) protokol. Na ten samy ucel ale pro přenášená media se používá SRTP (Secure Real-time Transport Protokol).

Důležitý bude taky říct ze SIP je textový protokol, který funguje v aplikační vrstvě podle modelu OSI. Odpovědi na dotazy SIP zdědil od jiného textového protokolu – http. Je definováno jenom 6 možných typu odpovědi na dotazy. Typ odpovědi je zakódován třímístným číslem, nejdůležitějším ze třech čísel je vždycky první.

2.7.1.1 Odpovědi SIP

1. 1XX – Informativní odpovědi. Nejznámější jsou **100 Trying**, **180 Ringing**, **183 Sesion Proggres**.
2. 2XX – Finální odezva, která znamená že dotaz byl úspěšně zpracován. Nejznámější je **202OK**.
3. 3XX – Finální odezva, která informuje o změně polohy zařízeni volaného uživatele. Nejznámější **302 Moved Temprorary**.
4. 4XX – Finální odezva, která informuje o odchylce nebo o chybě na straně klienta. Nejznámější **404 Not Found**.

5. 5XX – Finální odezva, která informuje o chybě na straně serveru. Nejznámější **500 Server Internal Error**
6. 6XX – Finální odezva, která informuje že spojení s volaným uživatelem není možné uskutečnit. Nejznámější je **603 Decline**.

2.7.1.2 Možné dotazy

REGISTER – Registrace uživatele na SIP serveru.

INVITE – Dotaz na zahájení nové komunikace. Obvykle obsahuje SPD-popis relace (kodeky, porty a td.).

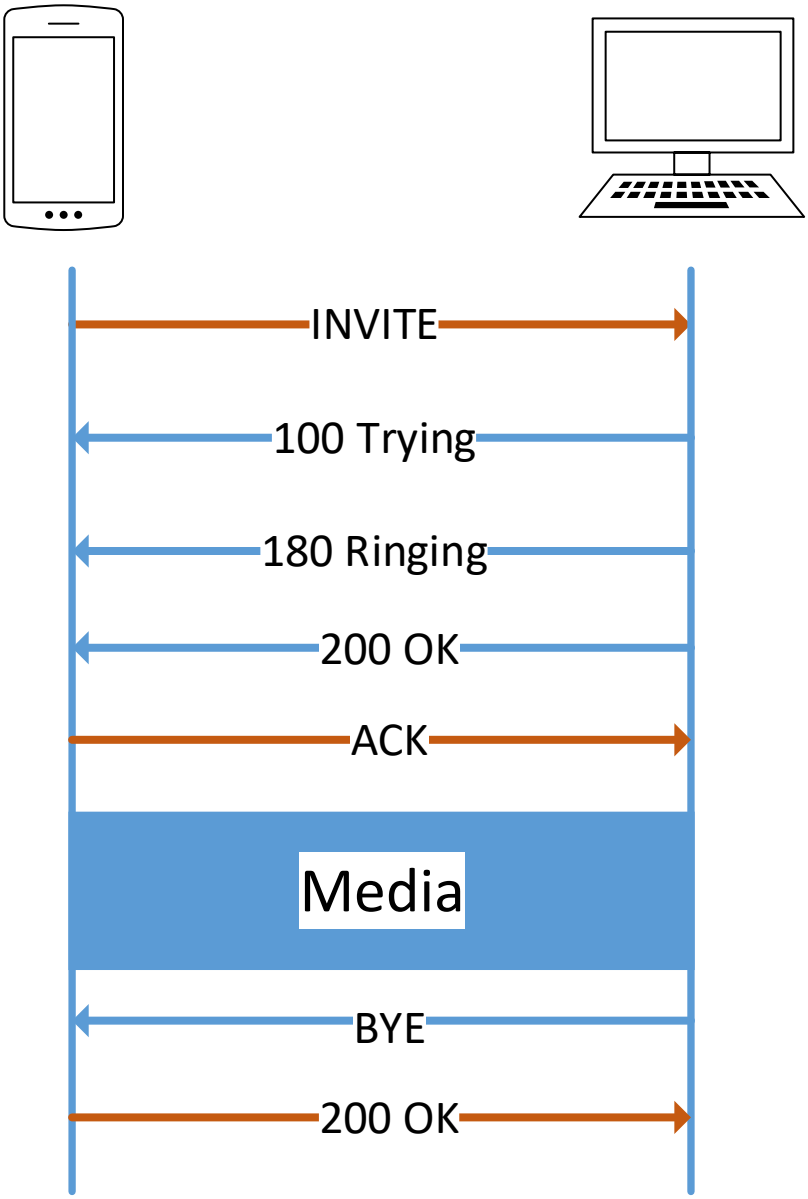
ACK – Potvrzení přijetí odpovědi na dotaz INVITE. Potvrzení startu relace.

CANCEL – Zruší zpracování dříve poslaných dotazů, ale neovlivňuje dotazy, které již byly zpracovány.

BYE – Ukončení relace. Může být převeden na kteroukoli ze stran, které se účastní relace.

OPTIONS – požádá informace o možnostech uživatele nebo serveru.

2.7.1.3 Příklad komunikace



Obr.5: Příklad SIP komunikace

2.7.2 Protokol DIAMETER

Jelikož jsme popsali SIP, nebude zbytečné představit protokol Diameter, který je druhým protokolem v IMS sítí. Jedna se o tak zvány protokol typu AAA, což znamená authentication, authorization and accounting, nebo český – autentizace, autorizace a účtovaní. Trochu podrobněji o tom, co znamená konkrétně každá z funkci protokolu:

- Autentizace – Proces, který umožňuje ověření pravosti subjektu na zaklade jeho identifikačních údajů. Například přihlašovacího jmena, telefonního čísla a hesla.
- Autorizace – Proces, který definuje oprávněni účastníka k přístupu k určitým objektům a servisům.
- Účtovaní – proces, který umožňuje sbíraní dat o použitých zdrojů. Primární jsou hodnoty příchozího a odchozího trafiku (v bajtech, megabajtech nebo gigabajtech).

Důležitou vlastností protokolu je jeho rozšiřitelnost a schopnost vytvářet nejen své vlastní atributy, ale i aplikací. Příkladem aplikace je RFC 4006 Diameter Credit Control Application, jehož vývoj začal v roce 2003, a konečný RFC byl propuštěn v roce 2005. Dále 3GPP na zaklade RCF 3588 a vytvořil celou řadu aplikací, jako je 3GPP 32.299 Diameter charging applications, což je důsledkem vývoje RFC 4006 DCCA.

Da se říct že dnes pro poskytnutí mobilních služeb téměř každému uživateli Diameter je použit v té či jiné podobě. A s vývojem 3G, IMS a LTE míra proniknutí má tendenci ke 100 %.

2.7.2.1 Typy uzlů Diametru

Specifikace definuje několik typů uzlů Diametru. Pro pochopeni roli uzlů, musíme zadat dva termíny, které budou podrobněji popsány níže.

Relace (Session) – kontroluje status účastníka a zahrnuje ty a pouze ty zprávy, které se týkají konkrétního účastníka. Každá relace má AVP SessionID, tento identifikátor je stejný pro všechny uzly, které se účastní v zpracování relace účastníka.

Spojeni (Connection) – kontroluje stav komunikace mezi uzly Diametru.

Klient – Diameter klient je obvykle síťové zařízení, kterým se přímo zpracovává trafik účastníka.

Server – Role serveru je zcela srozumitelná, měl by sledovat stav účastnických relací.

Agent – Diameter agenty jsou mezilehlými uzly mezi klientem a serverem a provádějí funkce správy provozu. Například mohou agregovat zprávy ze zařízení v jednom místě, vyrovnávat zatížení,

upravovat Diameter packety a sloužit jako bezpečnostní brány v přechodu z důvěryhodné sítě pro veřejné.

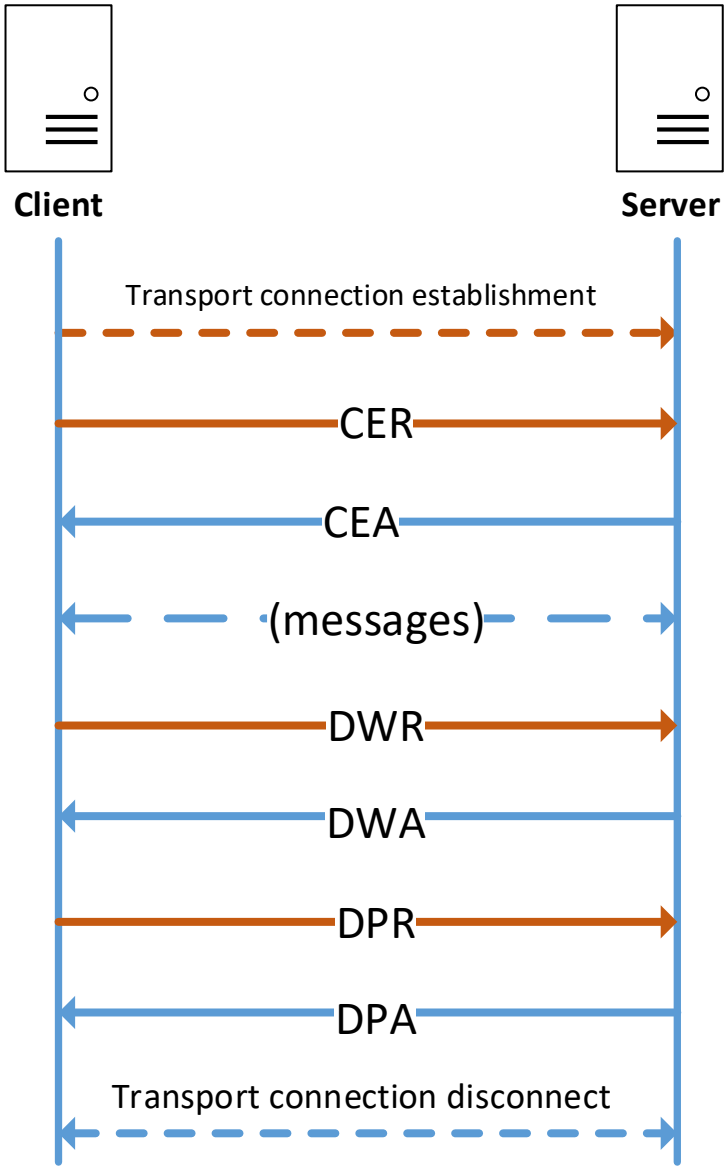
2.7.2.3 Rozdíl mezi RADIUS a DIAMETER

Diametr je dědicem jiného AAA protokolu – RADIUSU (tím se vysvětluji název), pro lepší porozumění moznosti DIAMETRu uvedu některý z hlavních rozdílu mezi nimi:

- Má dynamické objevovaní uzlu. Například používá DNS, což je standart v IMS sítí.
- Používá jen spolehlivé transportní protokoly – TCP nebo STCP, ale nikoli UDP.
- Oznámení o vyskytnutí chyb
- Má schopnost vyjednávaní mozností uzlů
- Je snadněji rozšiřitelný.
- Podpora TLS a IPsec

Jednou ze základních funkci Diametru v IMS síti je jeho použiti v komunikaci mezi S-CSCF a HHS. Protokol používá Cx a Dx rozhraní pro stahováni profilu účastníka a nahrávaní do S-CSCF. Ve velkých sítích, který nasazuji VOLTE (Voice over LTE) dnes se neobchází bez DRA (Diametr Routing Agent) pro podporu správného směrovaní Diametr správ.

2.7.2.4 Proud zprav DIAMETER

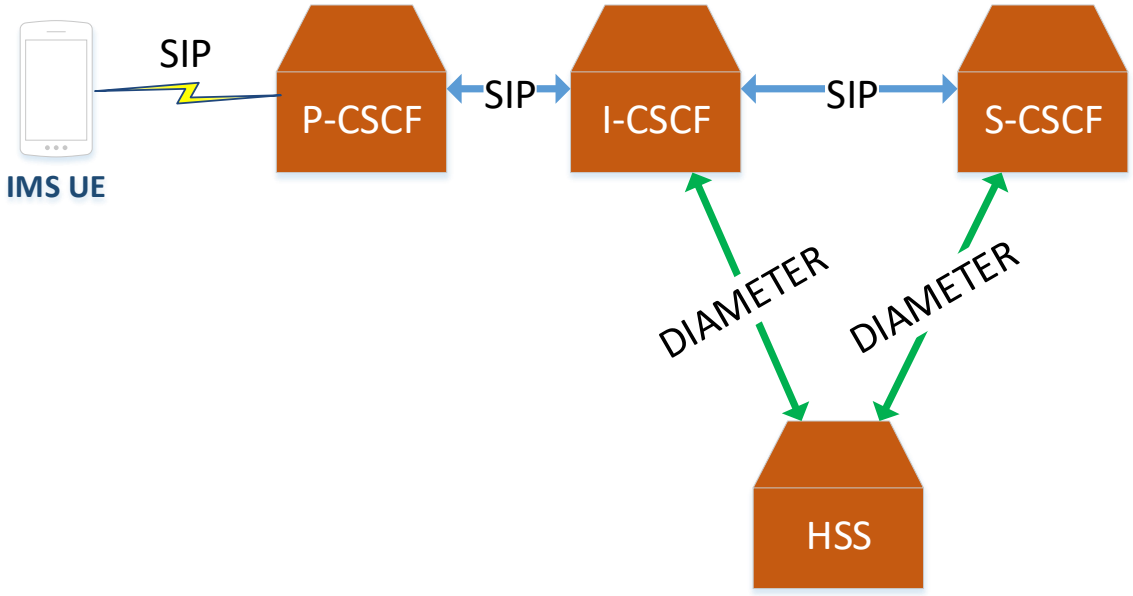


Obr.6: Proud zprav DIAMETER

- CER – Capabilities-Exchange-Request
- CEA – Capabilities-Exchange-Answer
- DWR – Device-Watchdog-Request
- DWA – Device-Watchdog-Answer
- DPR – Disconnect-Peer-Request
- DPA – Disconnect-Peer-Answer

2.8 Schéma Propojeni základních bodu IMS

Dál je uvedené schéma, ve kterém bude zobrazena komunikace základních prvků sítě IMS a protokolů, které jsou k tomu používány.



Obr.7: Schéma Propojeni základních bodu IMS

3. Otevřená implementace prvků IMS sítě

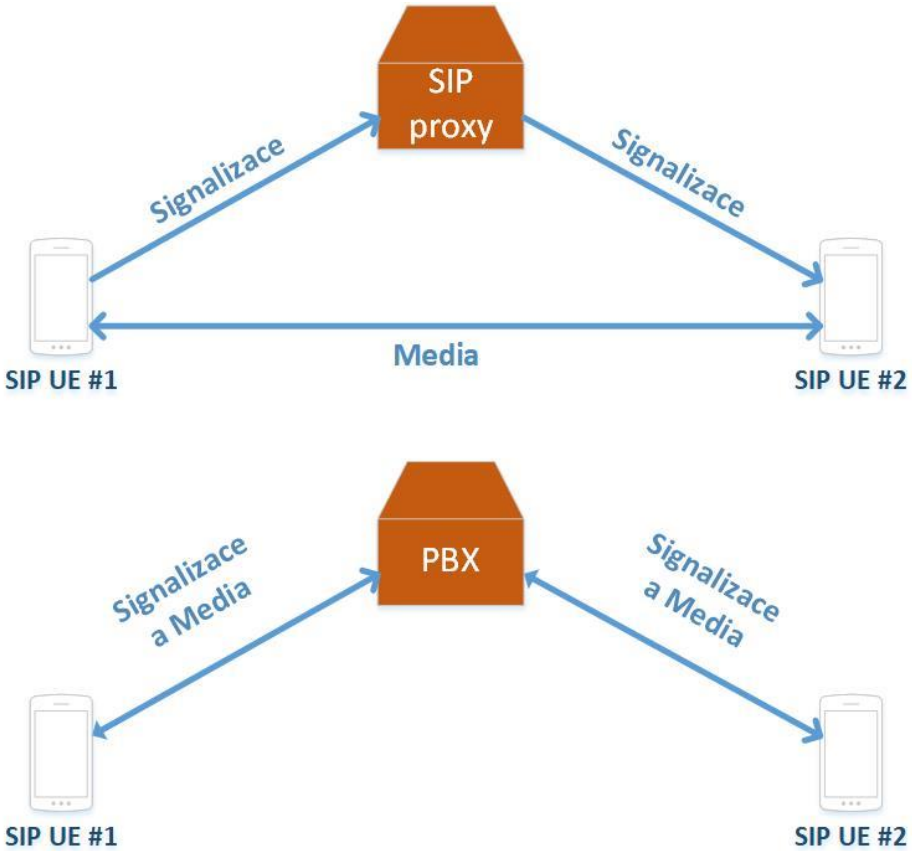
3.1 Kamailio

Kamailio je open source projekt který byl založen ještě v roce 2001 a v tu dobu se nazýval SER (SIP Router Express). Svůj nynější název dostal v roce 2008 kvůli problémům s obchodní značkou, kterou již měli jiné produkty. Dnes se na projektu primárně pracuje kolem 20 lidi s tím, že každý zájemce může podporovat a tvořit rozšíření a nove moduly.

Původně Kamailo je myšlen jako SIP Proxy. Ale kvůli modulárnímu designu a velmi flexibilní konfigurace má obrovské množství funkci. Dnes Kamailio podporuje víc než 100 rozšířeni včetně i

IMS core. Což dělá ho našim hlavním kandidátem pro realizace zakladu IMS sítí nutného pro testovaní SBC.

Hodně často lidi neúplně rozumí rozdíl mezi tím co je PBX (Private Branch Exchange) a SIP server. Nebo čím se Kamailio liší od často používaného ve VOIP Asterisku. Základním vysvětlením je to ze Asterisk je B2B (Back-to-back user agent), znamená to, že cela komunikace jednoho účastníka sítí s druhým probíhá jenom přes server. Neboli, da se říct, že každý účastník povídá s B2B agentem, který pak mezibodem pro oba účastníky, což umožňuje řadu takových funkci jako přehrávaní hlasových zprav, filtr volaní, přehrávaní hudby a hodně dalších funkcí. Kamailio vsak jako cisty SIP proxy tyto funkce nemá, protože dělá jenom ty funkce, které se tykají SIP protokolu, znamená to že je určen jenom pro signalizaci. Pro jednoduchost uvedu, níž par obrázků, ve kterých ukážeme, jak probíhá komunikace přes PBX a SIP server.



Obr.8:B2B a Proxy

Ale jak už bylo zmíněno Kamailio má obrovské množství modulů a rozšíření co předpokládá různé způsoby jeho využiti třeba i jako PBX.

Jedním ze základních problémů, na který jsem narazil při práci s Kamailiem je to že navzdory ke velkému množství rozšíření, Kamailio není na tolik široko používán jako Asterisk a má mnohem menší podporu, návodu a postupu při vzniku různých situací. Ještě hůř je to s projektem IMS core pro Kamailio. Je zřejmě že realizovaní IMS sítí na zaklade open source nedělá hodně lidi a tím počet článků a nápověd se ještě víc zužuje.

Ještě jedním minusem realizaci IMS na zaklade open source, je male množství IMS klientu a jejich docela slabý vývoj.

Pro realizace sítí se pokusíme nainstalovat Kamailio na virtuál s linuxem, který bude běžet na univerzitním serveru. Základní elementy P-CSCF, S-CSCF a I-CSCF budou se nacházet na stejném virtuálu, presto že v reálných sítích IMS tomu tak nebývá. Avšak ten scénář stačí nám pro testovaní a implementaci SBC do sítí.

3.2 BLOX

V otevřených zdrojích nám se podařilo najit několik open-source projektu které by se dalo použít pro implementaci do skloní IMS sítí. Konečně byl vybrán BLOX (blox.org), který představuje software udělaný na základě Linuxu a musí být nainstalován na server nebo počítač. Má taky GUI, které musí být nainstalovaný po instalaci základního softwaru.

Minimální Systémové požadavky [3] (až 90 současných hovorů):

- Dvoujádrový procesor Intel s 64 bitovou architekturou
- 2 GB RAM
- 2 síťové rozhraní (10/100/1000 Mbps). Interní a externí.
- 80 GB místa na pevném disku

Základní funkce:

- Odstraňuje špatnou VoIP signalizaci na hranici sítě.
- Vestavěný firewall, který může kontrolovat IP adresy a porty na základě filtrování, DOS / DDOS útoky, IP blacklist a NAT.
- Media bridging, který může zahrnovat Voice over IP a Fax over IP.
- Roaming rozšíření pro interní SIP PBX.
- Podpora SIP Outbound / Inbound trunku a politiky pro směrování hovorů.
- Podpora DTMF pro RFC2833 / INBAND / SIP INFO

Pokročilé funkce:

- Transcoding – SBC může také umožnit VoIP hovory mezi dvěma telefony pomocí transkódování mediálního toku, i když jsou používány různé kodeky.
- TLS / SRTP – podpora pro signalizaci a šifrování médií.
- Policy-based call routing, který umožnuje směrování hovorů podle politiky, včetně ukončení vadných hovorů.

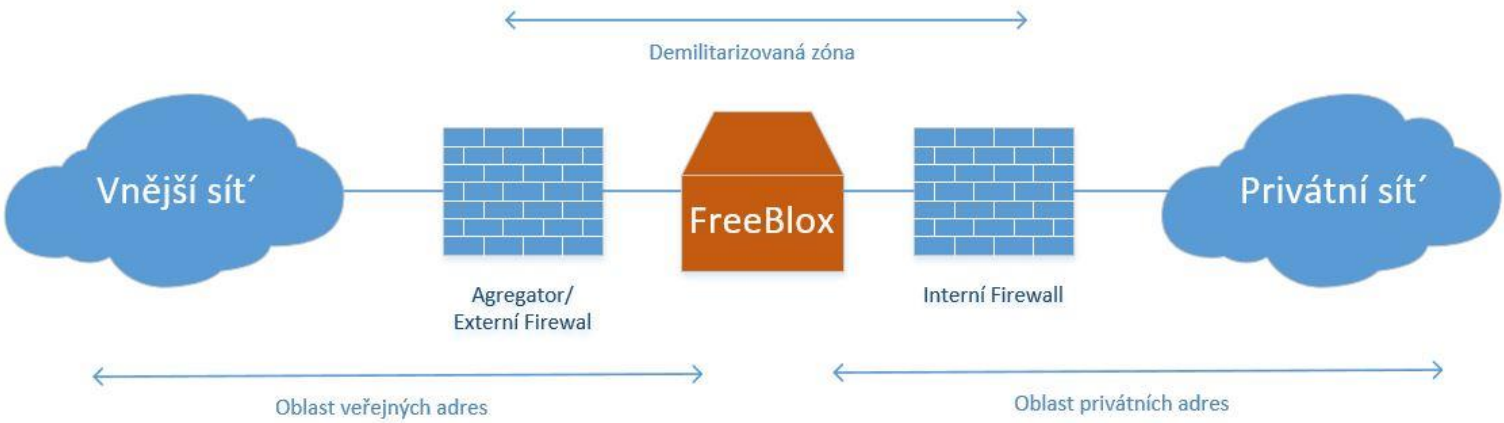
3.2.1 Scénáře, podle kterých BLOX může být použit

1 Scénář.

Obr.9: Umístěni SBC 1

Blox se používá pro kontrolu signalizaci mezi dvěma ústředny.

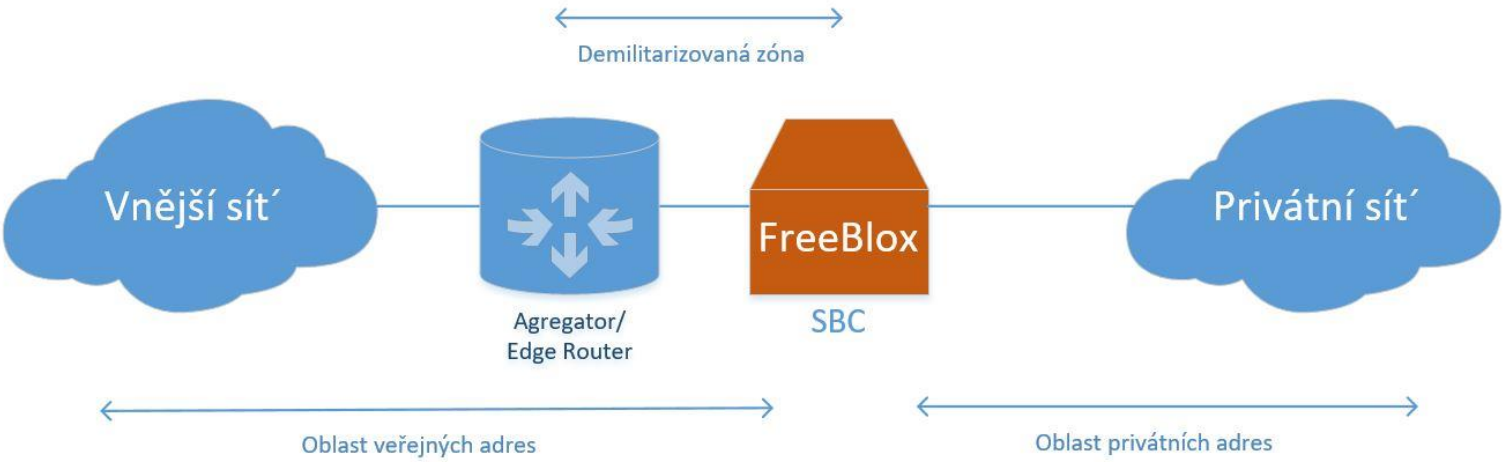
2 Scénář.



Obr.10: Umístěni SBC 2

Blox je nasazen v DMZ s providerem podporujícím VoIP volaní. V dalším povídaní o implementaci a spojeni SBC a IMS mezi sebou uvidíme, že budeme se držet druhého scénáře, který ale bude trochu rozšířen a udělán podle našich požadavku.

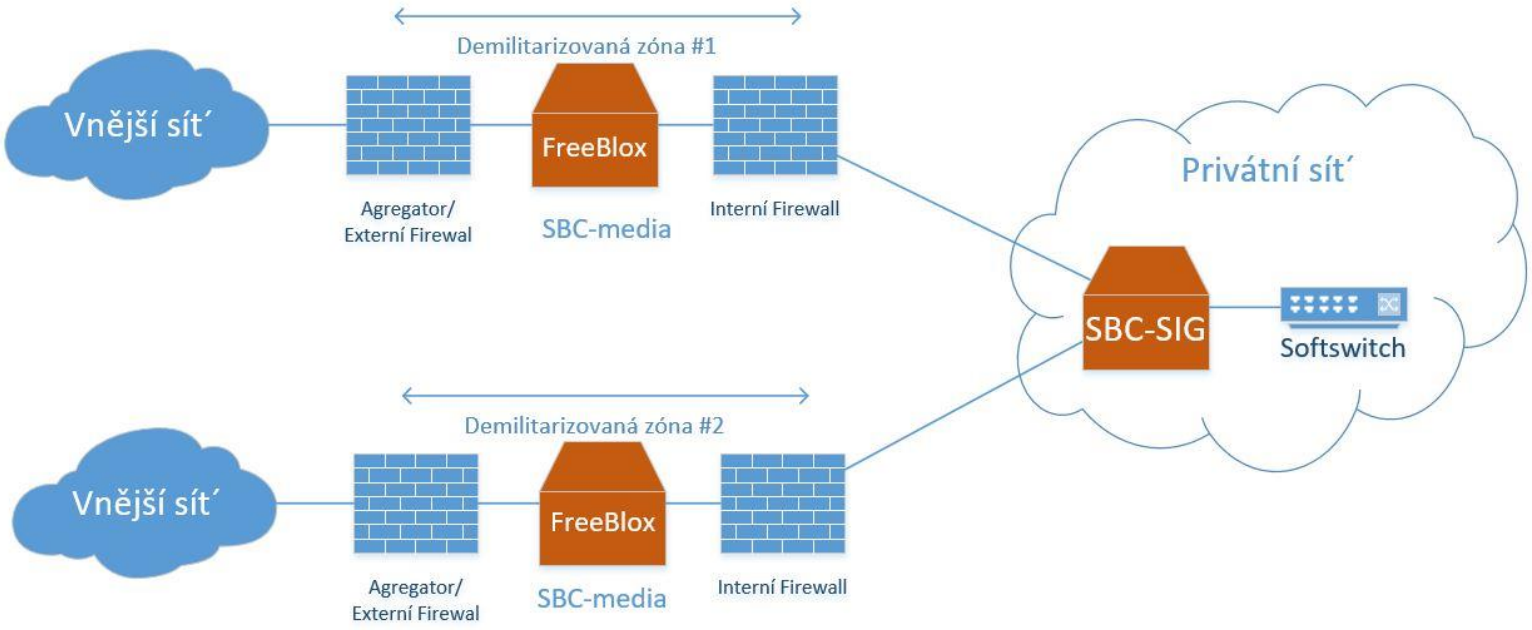
3 Scénář.



Obr.11: Umístěni SBC 3

V tomto scénáři Blox je jediné application-aware zařízení v DMZ. To znamená, že pokud aplikace v rámci privátní sítě vyžadují IP provoz procházet skrz DMZ se Blox přebírá odpovědnost za zajištění toho, aby ostatní zařízení v rámci DMZ propustili ten trafik.

4 Scénář.



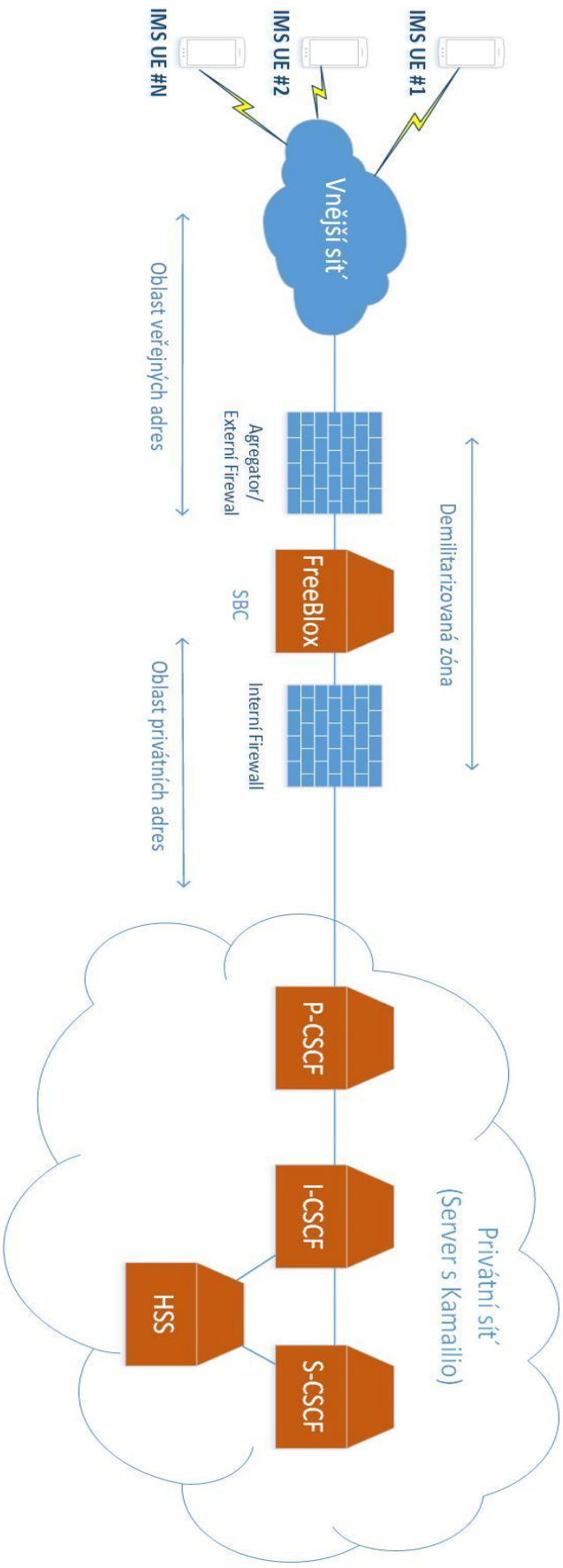
Obr.12: Umístěni SBC 4

V tomto scénáři veškera VoIP signalizace obdržena z veřejné sítě jde přes DMZ a směrována na Blox-SIG.

4. Implementace SBC BLOX do laboratorní sítě IMS

4.1 Plán sítí

Před tím, než přejdeme k popisu instalaci a konfiguraci SBC a Kamailio, představím plán sítí, kterého jsme se drželi od začátku. Jak vidíme SBC je hraničním bodem pro IMS, a všichni zprávy měli by jít přes něj. Klíčové body IMS sítí budou umístěny na jednom serveru na stejné IP adrese, ale s různými porty.



Obr.13: Plán sítí

4.2 Příprava k instalaci.

Instalace SBC a Kamailo jsme se rozhodli provádět na virtuálních strojích. Jako operační systémy ve virtuálech budou použité různé distribuce Linuxu. Linux jako hlavní operační systém byl zvolen nejen kvůli své flexibilitě a větší predispozice pracovat jako server, ale také jednoduše proto, že Blox a Kamailio můžou pracovat jenom na Linuxu. Všechny virtuální stroje s potřebnými komponenty, jsme se rozhodli instalovat vzdáleně, na **k332s.sinus.cz** serveru, který je fyzicky umístěn v naší fakultě. Pro přístup k serveru jsem použil ssh.

Níže uvedu některé vlastnosti serveru **k332s**, na kterém byly virtuální stroje emulovány:

```
SBC_Evgeny_Blox  debian-9.2.1-i386-xfce-CD-1.iso  vm-ims  vm-sbc  vremennoe
khalaevg@k332s:~$ free -m
             total       used       free     shared    buffers     cached
Mem:         16025      14258       1766          0        313      13414
-/+ buffers/cache:        530      15494
Swap:        15264          0      15264
```

Obr.14: RAM k332s

```
khalaevg@k332s:~$ df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/root        92G  4.3G   83G   5% /
devtmpfs        7.9G  564K  7.9G   1% /dev
/dev/sda3        32G   75M   30G   1% /var
/dev/sda4       670G   14G  623G   3% /home
tmpfs           128M     0  128M   0% /dev/shm
tmpfs           4.0G     0  4.0G   0% /tmp
```

Obr.15: Free space k332s

4.3 Virtualizace

Jako hypervisor pomoci, kterého budou emulovaný servery byl použít QEMU-KVM, jako jeden z nejefektivnějších a flexibilnějších nástrojů určených pro tyto účely.

4.3.1 Co je QEMU

Qemu je bezplatný open-source nástroj pro emulování a virtualizaci operačních systémů v počítači. Program může pracovat v systémech Windows, Linux, MacOS a dokonce i v systému

Android. Hostitelské zdroje, jako je procesor, pevný disk, paměť, jsou sdíleny mezi virtuálními počítači.

Qemu využívá hardwarovou virtualizaci, takže může pracovat s hostujícími operačními systémy téměř stejně rychle jako na hlavním hardwaru. Může používat modul jádra KVM, ale pouze v Linux.

QEMU může pracovat ve dvou režimech:

Plná emulace systému – v tomto režimu QEMU dělá úplnou emulaci zařízeni, například počítač, včetně všech jeho komponentů, procesoru a různých periferních zařízení. Může být použit pro spuštění více operačních systémů bez rebootování nebo ladění systémového kódu.

Emulace uživatelského režimu – funguje pouze pro hostitele Linuxu, umožňuje spouštět procesy Linux kompilované pro jednu architekturu v jiné, například ARM programy v x86. Užitečné pro vývoj a ladění.

Emulovat je možné tyto architektury: x86 (32 a 64 bitů), PowerPC (32 a 64 bit), ARM, MIPS (32 bitů), SPRAC (32 a 64 bit), Alpha, COLDFIRE (m68k), CRISv2 a MicroBlaze.

4.3.2 Některé komandy QEMU

Velmi užitečnou funkci QEMU jako u ostatních hypervizoru je moznost dělat snapshoty. Využil jsem tu funkci několik krát u instalovaní a konfiguraci SBC a Kamailio.

(qemu) **savevm** – uložit současný stav virtuální mašiny
(qemu) **loadvm** – nastartovat virtual v uloženém drive stavu
(qemu) **info snapshots** – zobrazit uložené snapshoty

```
List of snapshots present on all disks:
ID        TAG                 VM SIZE                DATE       VM CLOCK
--        snapshot1             1.3G 2017-12-21 20:14:24   01:40:44.974
--        snapshot2             262M 2017-12-28 12:12:18   00:00:41.049
```

4.4 Instalace komponent

4.4.1 Instalace SBC

Jak už bylo řečeno emulovaní SBC bude provedeno na serveru k332s, museli jsme teda na něj přihlásit a všechny další popsané kroky budou se provádět na něm. Pro instalaci Bloxu na virtuální server je nutné nejprve založit soubor který bude používán jako virtuální pevný disk pro systém. Nejjednodušší způsob, jak to udělat je pomoci komandu **touch**, formát souboru je cow2.

```
khalaevg@k332s:~/vm-sbc$ touch sbc.cow2
```

Na serveru musí taky byt nahrán soubor .iso ze kterého bude nainstalován systém, v našem případě to je Blox-1.0.4-8-stable-x86_64.iso. Potom je nutně vytvořit Shell script, který bude mít v sobě konfiguraci virtuálního počítače a bude používán pro jeho nastartovaní.

```
khalaevg@k332s:~/vm-sbc$ touch start
```

Editace jsme prováděli pomoci textového redaktoru vi, který je defaultně dostupný ve většině Linuxu. Dal se podíváme na obsah souboru a popíšeme jeho jednotlivé části.

Soubor start pro SBC

```
#I/bin/bash

qemu-system-x86_64 -enable-kvm -m 2048 -smp 2 -drive file=sbc.cow2,index=0,media=disk,if=virtio
-drive file=../SBC_Evgeny_Blox/Blox-1.0.4-8-stable-x86_64.iso,media=cdrom -net nic,model=rtl8139
-net tap, ifname=sbc-tap -net nic, model=rtl8139 -net tap, ifname=sbc-tap2 -vnc :XX,password -
monitor stdio
```

qemu-system-x86_64 -enable-kvm – Definuje architekturu systému, který bude emulován, enable-kvm mnohem urychli rychlost systému tím, že pro emulaci bude používán fyzické existující hardware hostitelského serveru.

-m 2048 – Operativní paměť v Mb.

-smp 2 – Symmetric Multiprocessing, jednoduše řečeno uvedli jsme, kolik bude použito jader ve virtuálním procesoru.

-drive file=sbc.cow2 – Definujeme soubor který bude pevným diskem virtuálního počítače. Tento soubor jsme založili v předchozích krocích

index = 0 – Tato opce určuje, kde je disk připojen pomocí indexu v seznamu dostupných konektorů daného typu rozhraní.

media=disk – Opce určuje typ připojeného media buď disk nebo cdrom.

if=virtio – Opce, která definuje, na který typ rozhraní je jednotka připojena. Dostupné varianty: ide, scsi, sd, mtd, floppy, pflash, virtio, none.

-drive file=../debian-9.2.1-i386-xfce-CD-1.iso – cesta k instalačnímu souboru

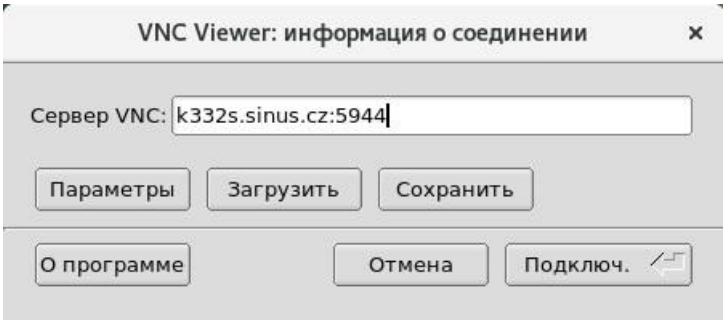
-net nic,model=rtl8139 – Definuje síťový interface, dal se definuje model sítové karty.

-net tap,ifname=ims-tap – Připojuje rozhraní TAP hostitele k VLAN číslo N, když vlan není označeno připojuje k VLAN 0. Dal se definuje název rozhraní které bude na straně hostitele.

-vnc :XX,password -monitor stdio – Port XX pro VNC (Virtual Network Computing), z bezpečnostních důvodu je označen jako XX. Password -monitor studio, tato opce děla to že bude vyžadovat heslo při přístupu k virtuální mašině přes VNC klient.

Dal spouštěni virtuálu bude se provádět pomoci komandu **sh.** Pak se dá pokračovat v instalaci pomoci VNC klientu, v jsme použili Tiger VNC.

```
khalaevg@k332s:~/vm-sbc$ sh start
```



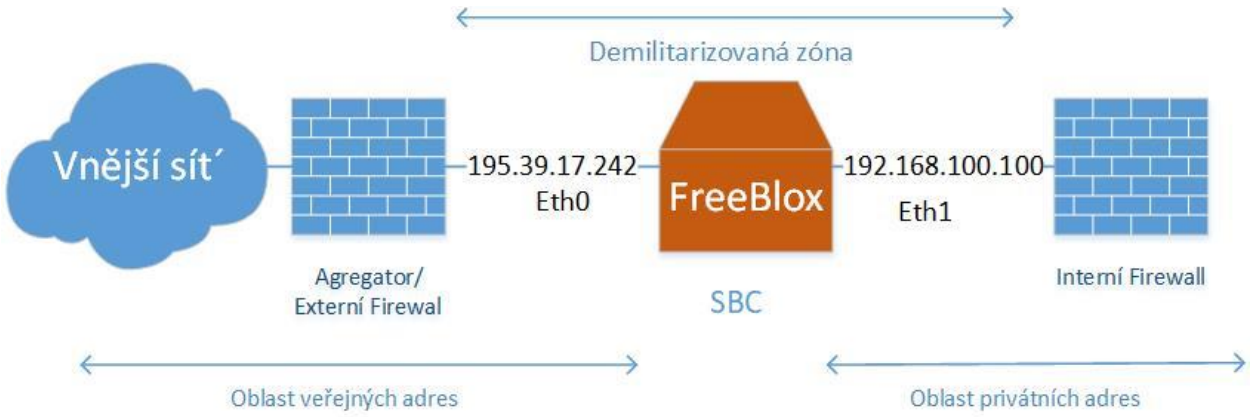
Obr.16: VNC client

Následuje pak instalace, která se snadno dělá krok za krokem.

Obr.17: Instalace SBC

4.4.2 Konfigurace síťových rozhraní SBC.

Jak už bylo uvedeno SBC je hraničním bodem sítí, má taky za úkol skrývat topologii sítí. Nejjednodušší způsob splnit ty požadavky je nakonfigurovat na SBC 2 sítové rozhraní. Jeden by měl mít přístup k veřejné sítí. Druhý bude mít lokální adresu ve stejné sítí jako IMS platforma.



Obr.18: Plán rozhraní SBC

Konfigurovat sít na Linuxu je možné dělat hodně způsoby. My jsme se to však dělali přes ten, který považujeme za nejjednodušší a nejefektivnější, a to je přes konfigurační soubory interfaců.

Cesta ke konfiguračním souborům sítí v různých verzích Linuxu se liší. V Bloxu tyto soubory se nachází v této složce:

```
[root@localhost ~]# cd /etc/sysconfig/network-scripts/
```

Dal přejdeme na konfiguraci souboru **ifcfg-eth0**. Tento interface bude mít veřejnou adresu a mít přístup k internetu. Z toho že SBC je hraničním bodem je taky očividně ze adresy interface můžou být jenom statické, nikoli dynamické.

```
DEVICE="eth0"
BOOTPROTO="static"
HWADDR="52:54:00:12:34:56"
IPADDR=195.39.17.242
NETMASK=255.255.255.248
GATEWAY=195.39.17.241
DNS1=89.250.251.129
NM_CONTROLLED="yes"
ONBOOT=yes
TYPE="Ethernet"
UUID="b4ddbebf-95f5-4a68-a25a-04925ec16952"
```

Několik poznámek: UUID je unikátní identifikátor zařízeni pro případy, že dva interfacy budou mít stejnou fyzickou adresu. Opce ONBOOT=yes znamená že interface bude se zapínat se spuštěním systému. Jako gateway jsme ukázali interface na serveru k332s, přes který SBC má přístup k veřejným sítím. Tim jsme ukončili konfigurace jednoho rozhraní.

Interface Eth1 museli jsme nejprve založit jako soubor ve složce network. Nastavovaní bylo provedeno obdobným způsobem, pomoci upravovaní založeného souboru. IP adresa rozhraní je tento krát lokální a bude ve stejné sítě jako IMS, gateway uveden není. Pak se nastaví pravidlo, podle kterého pakety z interface s lokálními adresy budou mít přistup do veřejné sítí přes NAT (Network Address Translation), čímž docílíme toho, že topologie IMS bude skryta.

```
DEVICE="eth1"
BOOTPROTO="static"
IPADDR=192.168.100.100
NETMASK=255.255.255.0
DNS1=89.250.251.129
NM_CONTROLLED="yes"
ONBOOT=yes
TYPE="Ethernet"
UUID="6ac56659-1b03-48e5-80db-cfc9cf9783d6"~
```

Až jsme měli všechny rozhraní nakonfigurované ověřili jsme jejich funkčnost pomoci komandů **ping** a **traceroute**.

Ještě jedním malým ale velmi důležitým krokem bylo přidáni níže uvedeného pravidla do IP tables.

```
Iptables -t nat -A POSTROUTING -o eth0 -s 192.168.100.0/24 -j MASQUERADE
```

Tim jsme docílí toho, že lokální adresy budou skryté za veřejnou adresou, v našem případě to bude veřejná adresa SBC.

Na Bloxu defaultně je nainstalován **ssh server**, takže po konfiguraci síťových rozhraní už se dalo vzdálené spravovat SBC přes **ssh**.

Níže uvedu to, jak konečné vypadalo nastaveni síťových rozhraní na SBC.

```
[root@localhost ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr 52:54:00:12:34:56
          inet addr:195.39.17.242  Bcast:195.39.17.247  Mask:255.255.255.248
          inet6 addr: fe80::5054:ff:fe12:3456/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:8828 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6478 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2030965 (1.9 MiB)  TX bytes:1906754 (1.8 MiB)
          Interrupt:11 Base address:0x8000

eth1      Link encap:Ethernet  HWaddr 52:54:00:12:34:57
          inet addr:192.168.100.100  Bcast:192.168.100.255  Mask:255.255.255.0
          inet6 addr: fe80::5054:ff:fe12:3457/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2372 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2831 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:393622 (384.3 KiB)  TX bytes:1484950 (1.4 MiB)
          Interrupt:11 Base address:0x4000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:4 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1252 (1.2 KiB)  TX bytes:1252 (1.2 KiB)
```

4.4.3 Instalace GUI pro Blox.

Pro instalaci GUI museli jsme nahrát na server s Bloxem tento soubor freeblox-0.9.0-15.x86_64.rpm. Potom všechno co je třeba udělat je nainstalovat zkopírovaný balík pomoci Redhat Package Manager (manažer pro balíky Redhat) a restartovat systém.

```
[root@localhost ~]# rpm -ivh freeblox-1.0.5-2.x86_64.rpm
```

Pro přístup do GUI stačí zadat do browseru veřejnou IP adresu SBC 195.39.17.242.

Obr.19: Graficky interface Blox

4.4.4 Instalace virtuálu pro Kamailio.

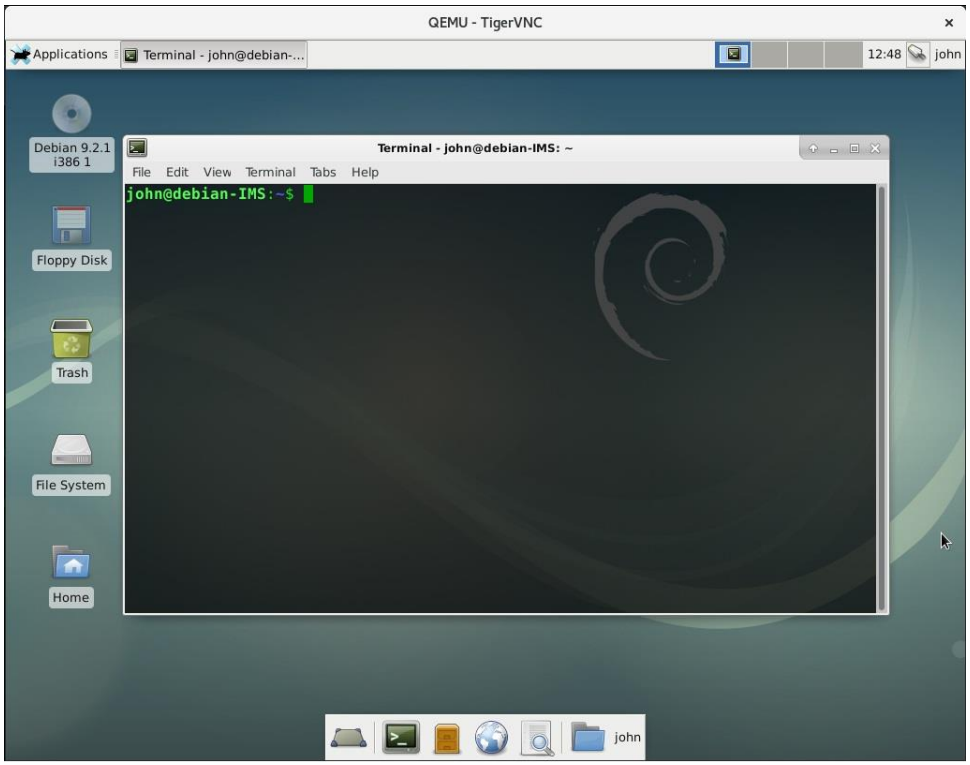
Jako operační systém, na kterém bude nainstalován Kamailio námi byl zvolen Debian s jako jeden z distributivů, které se nejvíc používají jako v serverech. Příprava k instalaci je podobna výše popsané přípravě k instalaci Bloxu. Museli jsme vytvořit soubor **ims.cow2.** Potom byl vytvořen **shell script.** Od předchozího konfiguraku se liší tím ze v tomto virtuálu bude jenom jedno rozhraní se statickou adresou.

```
#!/bin/bash

qemu-system-x86_64 -enable-kvm -m 2048 -smp 2 -drive
file=ims.cow2,index=0,media=disk,if=virtio -drive file=../debian-9.2.1-i386-xfce-CD-
1.iso,media=cdrom -net nic,model=rtl8139 -net tap,ifname=ims-tap -vnc :YY,password -
monitor stdio
```

```
khalaevg@k332s:~/vm-ims$ sh start
```

Následně se provádí obyčejná instalace. Pro přístup k systému se zapnutou grafickou obálkou jsme používali Tiger VNC viewer.



Obr.20: Graficky interface Debian s obálkou Xface

Hned po instalaci byla potřeba provést upgrade systému a nainstalovat **ssh** server v případě správy systému přes terminál, což se bude hodit.

```
root@debian-IMS:/home/john# apt-get update
root@debian-IMS:/home/john# apt-get upgrade
root@debian-IMS:/home/john# apt-get install openssh-server.
```

4.4.5 Konfigurace rozhraní Debian

Cesta **/etc/network/interfaces** ke konfiguračnímu souboru v Debianu se liší od Bloxu základem kterého je CentOS, Lisi se taky i způsobem nastaveni

```
# The loopback network interface
auto lo
iface lo inet loopback
auto ens3
  iface ens3 inet static
  address 192.168.100.101
  netmask 255.255.255.0
  gateway 192.168.100.100
  dns-nameservers 192.168.100.101 127.0.0.1 8.8.8.8
```

Jak vidíme máme nastavenou zase statickou adresu z ty samé lokální sítí jako u Eth1 SBC. Jako gateway je uvedena lokální adresa SBC. Znamená to, že vdechni pakety budou nejdříve poslaný uvnitř lokální sítí na rozhraní Eth1 SBC a pak na Eth0, které má přístup k internetu. Kvůli přidanému dříve pravidlu do IP tables SBC adresa IMS bude skryta za NATem, čímž jsme splnili požadavek, podle kterého SBC skrývá topologii IMS sítí.

4.5 Propojeni komponent

4.5.1 Propojeni serverů

Abychom propojili SBC s ostatními části IMS sítí použili jsme nastroj **brctl** na straně serveru **k332s,** na kterém běží oba virtuály. Měli jsme za cíl udělat 2 bridže. Jeden by mel propojovat Vnější rozhraní SBC, které na straně serveru k332s má název **sbc-tap** (tento název jsme definovali v konfiguračním souboru start) s rozhraním eth3 které má přistup k internetu. Druhy bridž propojuje rozhraní IMS a SBC které mají lokální adresy.

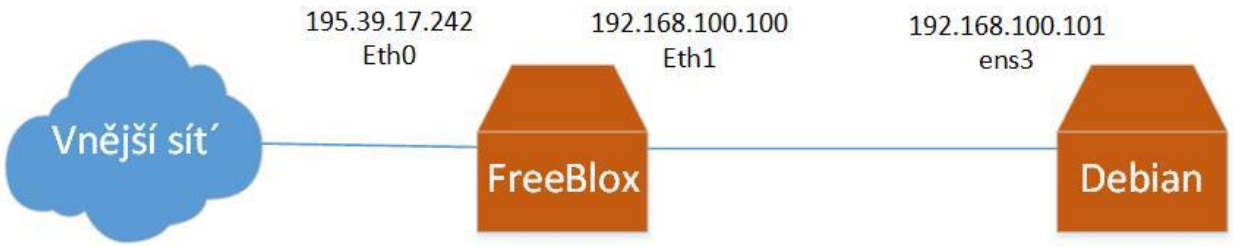
Zobrazíme provedené kroky:

```
khalaevg@k332s:~$ brctl addbr br0
khalaevg@k332s:~$ brctl addif br0 eth3
khalaevg@k332s:~$ brctl addif br0 sbc-tap
khalaevg@k332s:~$ ifconfig br0 up

khalaevg@k332s:~$ brctl addbr br1
khalaevg@k332s:~$ brctl addif br1 ims-tap
khalaevg@k332s:~$ brctl addif br1 sbc-tap2
khalaevg@k332s:~$ ifconfig br1 up
```

```
khalaevg@k332s:~$ /sbin/brctl show
bridge name     bridge id               STP enabled     interfaces
br0             8000.001e6849ff6b       no              eth3
                                                        sbc-tap
br1             8000.4ebb13548216       no              ims-tap
                                                        sbc-tap2
```

Jak vidíme oba bridže jsou nakonfigurované a schéma propojení serverů měla by v ten okamžik vypadat takto:



Obr.21: Propojeni Debian a Blox

4.5.2 Řešeni problém s křižováním

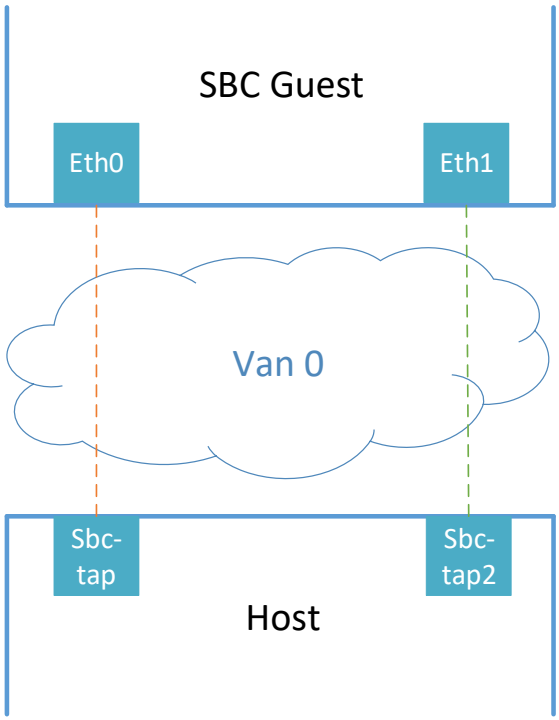
V průběhu konfiguraci síťových rozhraní jsme narazili na problém křižovaní rozhraní mezi virtuálem SBC a serverem K332 na kterém ten virtuál běží. Způsobeno to bylo nesprávné nakonfigurovaným hypervizorem QEMU. Předpokládali jsme, že rozhraní na SBC jsou napevno propojeny pomoci tunely s odpovídajícím rozhraním na K332 (Například eth0 na SBC propojeno s sbc-tap na k332s). Nicméně tomu tak nebylo a při pokusu pingovani nějaké vnější adresy z Debianu dostávali jsme pakety označené jako DUP!, což znamená, že vracena odpověď je bud duplikovaná nebo pokažena. Faktem je, že ve výchozím nastavení hypervisor vytváří mezi Hostem a Guestem jeden VLAN, ale jak už bylo řečeno na SBC máme 2 rozhraní jeden z kterých má veřejnou adresu a přístup k internetu, druhy má lokální adresu a se spojuje s IMS platformou. Výsledkem toho že oba rozhraní posílali data přes společnou VLAN bylo to ze data z jednoho rozhraní na SBC se přenášeli na oba "protilehlé" rozhraní na serveru. Aby informace se přenášela jen mezi dvojici odpovídajících rozhraní je nutně aby mezi touto dvojici byla samostatná VLAN síť. Za tímto účelem jsme opravili konfigurační soubor „start" do následujícího stavu:

qemu-system-x86_64 -enable-kvm -m 2048 -smp 2 -drive file=sbc.cow2,index=0,media=disk,if=virtio -drive file=../SBC_Evgeny_Blox/Blox-1.0.4-8-stable-x86_64.iso,media=cdrom -net nic,model=rtl8139 -net tap,**vlan=0**,ifname=sbc-tap -net nic,**vlan=1**,model=rtl8139 -net tap,**vlan=1**,ifname=sbc-tap2 -vnc :33,password -monitor stdio

Jak je vidět ve správné variantě je u každého rozhraní už označeno ve které VLAN leží.

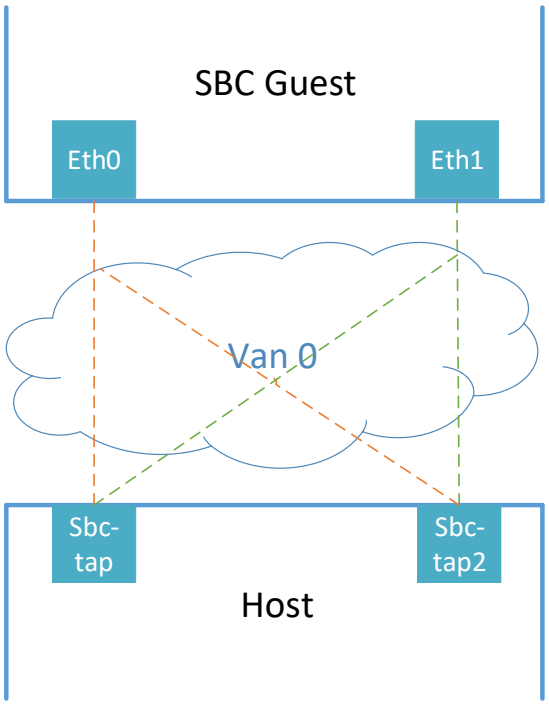
Aby bylo úplnou srozumitelnost tomu, co jsme provedli níže jsou uvedeny tři obrázky popisující následující stavy:

1. **To, jak jsme předpokládali, že rozhraní funguje:**



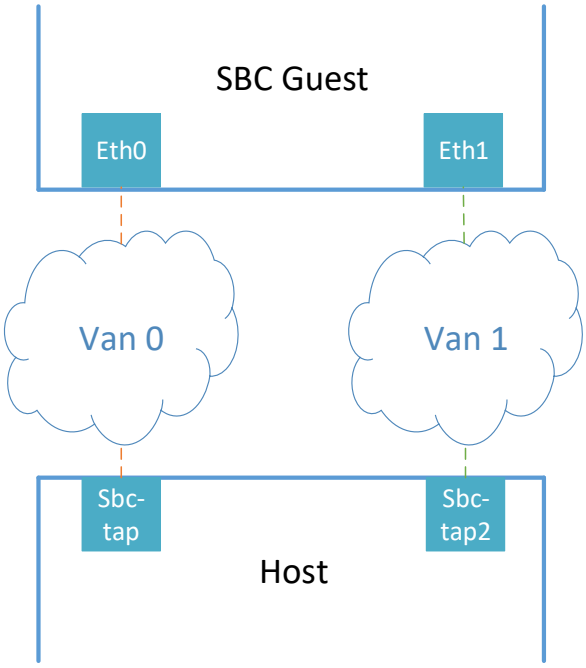
Obr.22: Konfigurace rozhraní 1

2. **Jak to skutečně fungovalo:**



Obr.23: Konfigurace rozhraní 2

3. **Správná konfigurace rozhraní:**



Obr.24: Konfigurace rozhraní 3

4.6 Instalace Kamailio a DNS serveru

4.6.1 Instalace Kamailio

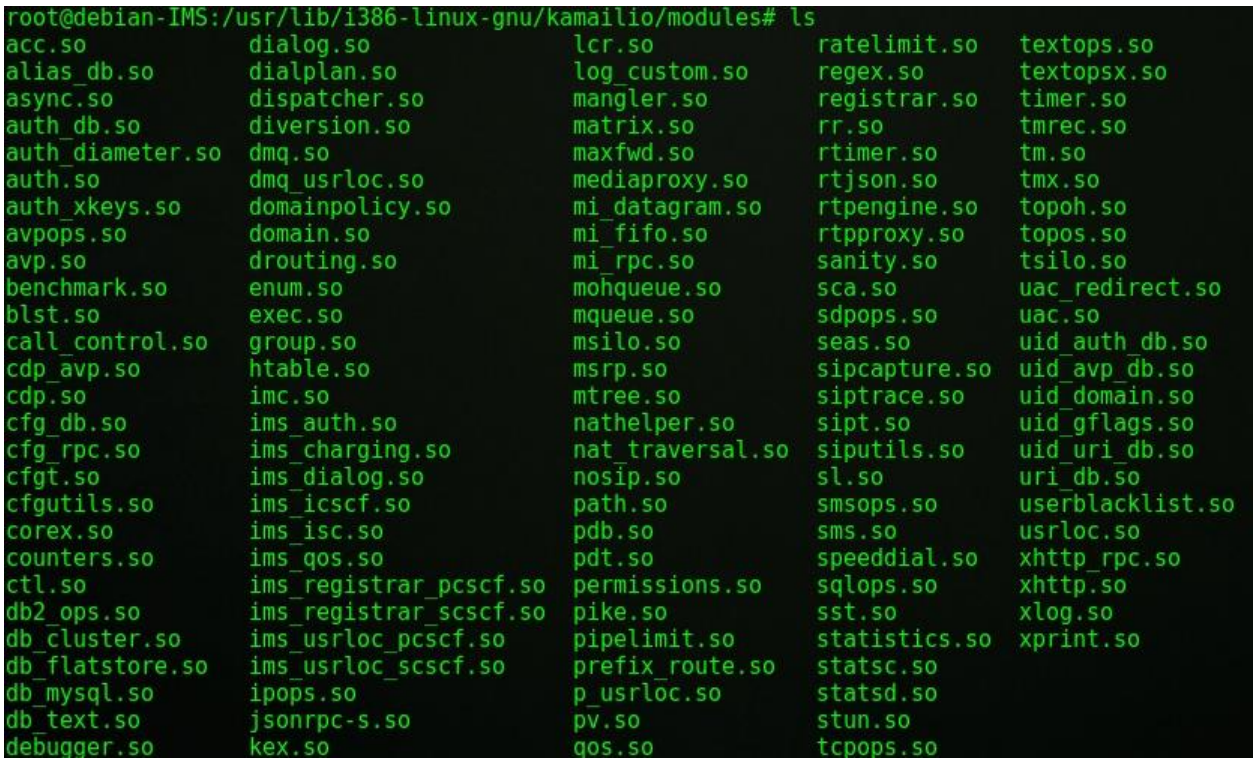
Instalace Kamailio sama o sobě těžká není. Existuje několik moznosti: zkompilovat samostatně nebo nainstalovat pomoci package manažeru. Druha moznost připadá jako nejlehčí, proto jsme ji využili. Nainstalujeme taky potřebné pro naší ucely moduly, hlavně IMS.

```
sudo apt-get install kamailio kamailio-mysql-modules kamailio-ims-modules
```

Po prvotním spouštěni Kamailio vypadá takto:

```
root@debian-IMS:/etc/kamailio# kamailio start
loading modules under config path: /usr/lib/i386-linux-gnu/kamailio/modules/
Listening on
        udp: 127.0.0.1:5060
        udp: 192.168.100.101:5060
        tcp: 127.0.0.1:5060
        tcp: 192.168.100.101:5060
Aliases:
        tcp: localhost:5060
        udp: localhost:5060
```

Po instalaci je dobry se podívat, jestli moduly IMS byli úspěšné nainstalované a v jaké složce jsou, bude se to hodit u konfiguraci.



Obr.25: Nainstalované moduly Kamailio

4.6.2 Instalace DNS

Jako DNS server byl použit BIND. Je to otevřená a nejběžnější implementace serveru DNS, která zajišťuje převedeni DNS nazvu na adresu IP a naopak. Je pojmenován spustitelný souborový démon serveru BIND.

Pro instalaci zadáme

```
sudo apt-get install bind9
sudo apt-get install bind9-doc
```

Pro zapnuti DNS serveru

```
systemctl start bind9
```

4.7 Konfigurace Kamailio a DNS serveru

4.7.2 Konfigurace Kamailio IMS

Po instalaci Kamailio/IMS je třeba vytvořit konfiguraci pro každý z elementu. Pro urychleni prací námi byli částečně použité konfigurační soubory za autorstvím pana Franz Edle [2].

```
root@debian-IMS:/home/john/Downloads/Kamailio-IMS config-files# ls
bind  configurator.sh  modules.lst  my-icscf  my-pcscf    my-scscf  ssh
```

Aby všechny prvky fungovali na stejném serveru naraz je potřeba aby každý mel svůj vlastní port na stejné IP adrese. Podíváme se na příklad z konfiguraku pcscf.cfg:

```
# SIP / UDP
# listen=udp:109.239.57.200:5060
listen=pcscf.net1.test
port=4060
```

Jak vidíme řádek s adresou, na kterou bude poslouchat **pcscf** je zakomentovány, místo toho je definován port a textový adres, který bude dal definován v DNS serveru. Porty pro **pcscf**, **icscf** a **scscf** budou **4060**,**5060** a **6060** přiměřeně.

V druhem konfiguračním souboru které mimo vše ostatní popisuje jaké moduly potřebuje prvek IMS je důležité změnit cestu ke složce s moduly.

```
# ---------- module loading ----------
mpath="/usr/lib/i386-linux-gnu/kamailio/modules/"
```

4.7.2 Konfigurace DNS

Přes to, že nainstalované námi dříve elementy IMS sítí běží na jednom serveru je nutné správně nakonfigurovat DNS aby prvky mohly mezi sebou komunikovat. Nová konfigurace v Bindu se vytváří ve složce /**etc/bind/**. Na následujícím obrázku je uvedena naši varianta:

```
$ORIGIN net1.test.
$TTL 1W
@                       1D IN SOA       serverkde. root.net1.test (
                                        2010081401      ; Serial YYYYMMDDXX
                                        3H              ; refresh
                                        15M             ; retry
                                        1W              ; expiry
                                        1D )            ; minimum

                        1D IN NS        server
server                                                          1D IN A         192.168.100.101
pcscf                   1D IN CNAME     server
icscf                   1D IN CNAME     server
scscf                   1D IN CNAME     server
hss                     1D IN CNAME     server
as1                     1D IN CNAME     server
as2                     1D IN CNAME     server
xcap                    1D IN CNAME     server

sbc                 1D IN A         192.168.100.100

net1.test.                  1D IN NAPTR 10 100 "s" "SIP+D2T"    ""      _sips._tcp.net1.test.
net1.test.                  1D IN NAPTR 20 100 "s" "SIP+D2T"    ""      _sip._tcp.net1.test.
net1.test.                  1D IN NAPTR 30 100 "s" "SIP+D2U"    ""      _sip._udp.net1.test.

_sip                    1D SRV 0 0 5060 server
_sips._tcp              1D SRV 0 0 5061 server
_sip._tcp               1D SRV 0 0 5060 server
_sip._udp               1D SRV 0 0 5060 server
```

Obr.26: Konfigurace DNS serveru

Na SBC serveru bylo nutné přidat adresu Debian serveru jako DNS adresu.

```
[root@localhost etc]# cat resolv.conf

nameserver 192.168.100.101
```

Pro jistotu z DNS opravdu funguje je možné udělat ping z SBC na libovolnou adresu, která už byla definovaná v konfiguračním souboru Bind.

```
[root@localhost etc]# ping pcscf.net1.test
PING server.net1.test (192.168.100.101) 56(84) bytes of data.
64 bytes from 192.168.100.101: icmp_seq=1 ttl=64 time=0.233 ms
64 bytes from 192.168.100.101: icmp_seq=2 ttl=64 time=0.298 ms
64 bytes from 192.168.100.101: icmp_seq=3 ttl=64 time=0.392 ms
64 bytes from 192.168.100.101: icmp_seq=4 ttl=64 time=0.293 ms
--- server.net1.test ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3133ms
```

Jak vidíme odkaz na **pcscf** je správně převeden na IP adresu IMS, což správně z předpokladu že **pcscf** a ostatní prvky budou pracovat na stejném serveru.

5. Závěr

Vzhledem k tomu že Kamailio/IMS se většinou používá velmi zřídka a jenom pro studijní účely, jeho balíčky mají experimentální stav a hodně často nefunguji. Bohužel veškerou snahu se dosud nepodařilo zprovoznit na 100% IMS, což je nutné pro testovaní SBC, jako hraničního prvku cele sítí. V současnem stavu máme spuštěny SBC a částečně IMS. Servery jsou mezi sebou propojeny a předávají pakety bez problémů. Topologie sítí se skrývá za NATem. Funguje nám taky DNS server, což je nutnou součásti VOIP komunikace.

V dnešní době bylo asi zbytečné říkat o aktualitě rozvoje IMS. Jako platforma, která by uměla VoLTE a mobilní komunikací předchozí generace, IMS už se relativně dávno začalo objevovat dokonce i v Česku.

Důležitost Session Border Controlleru jako jednotlivého prvku reálné IMS sítí je doporučeno především dodavateli sítí. Implementace funkcí SBC do jiného elementu v praktice dosud se neděla. Pro nás to znamená že přidáváni tohoto prvku do univerzitní sítí a jeho testovaní melo by význam v tom, že připlížilo by to fungovaní studijní sítí k reálným podmínkám. Proto pracovat nad jeho zprovozněním spolu s Kamailio/IMS budeme i dal.

6. Přilohy

Seznam obrazků

Seznam použitých zkratek a symbol

LTE	Long-Term Evolution
IMS	IP Multimedia Subsystem
VoLTE	Voice over LTE
SBC	Session Border Controller
P-CSCF	Proxy Call State Control Function
I-CSCF	Interrogating Call State Control Function
S-CSCF	Serving Call State Control Function
RAM	Random Access Memory
DNS	Domain Name System
VNC	Virtual Network Computing
B2B	Business to business
MSISDN	Mobile Subscriber Integrated Services Digital Number
UDP	User Datagram Protocol
TCP	Transmission Control Protocol
SIP	Session Initiation Protocol
Tap	virtual network kernel devices
IMSI	International Mobile Subscriber Identity
UUID	Universally Unique Identifier
GUI	Graphical User Interface
VLAN	Virtual Local Area Network
LAN	Local Area Network
ISDN	Integrated Services Digital Network
PSTN	Public Switched Telephone Network
NAT	Network Address Translation
IP	Internet Protocol
HSS	Home Subscriber Server
MRF	Media Resource Function
RADIUS	Remote Authentication in Dial-In User Service

7. Bibliografie

[1] – Obrazek: *Figure-3 3GPP/TISPAN IMS architectural overview* [online]. [cit. 21.12.2017]. Dostupný na WWW: http://dentisty.org/telecommunication-broadcasting-convergence.html

[2] – Kamailio IMS in a Box [online]. [cit. 21.12.2017]. Dostupný na WWW:

https://www.kamailio.org/w/2016/02/kamailio-ims-getting-started-box/

[3] – Blox [online]. [cit. 21.12.2017]. Dostupný na WWW:

http://www.blox.org/

[4] – SIP: Session Initiation Protocol rfc2543 [online]. [21.12.2017]. Dostupný na WWW:

https://tools.ietf.org/html/rfc2543

[5] – Diameter Base Protocol rfc6733 [online]. [21.12.2017]. Dostupný na WWW:

https://tools.ietf.org/html/rfc6733

[6] – RADIUS rfc3579 [online]. [21.12.2017]. Dostupný na WWW:

https://tools.ietf.org/html/rfc3579

[7] – Obrazek: *SIP call flow without SBC* [online]. [cit. 21.12.2017]. Dostupný na WWW:

https://www.frafos.com/resources/white-papers/understanding-session-border-controllers/

[8] – Obrazek: *SIP call flow with SBC* [online]. [cit. 21.12.2017]. Dostupný na WWW:

https://www.frafos.com/resources/white-papers/understanding-session-border-controllers/