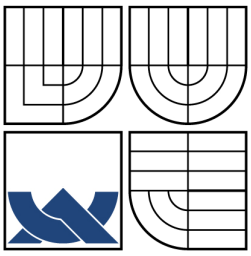
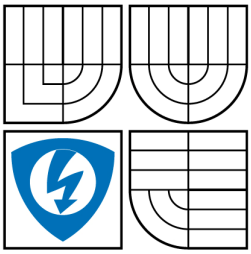



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

**BRNO UNIVERSITY OF TECHNOLOGY**


FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

**ÚSTAV TELEKOMUNIKACÍ**

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

**DEPARTMENT OF TELECOMMUNICATIONS**


# SPOJOVACÍ SYSTÉMY ZALOŽENÉ NA IP TELEFONII

**COMMUNICATION SYSTEMS BASED ON IP TELEPHONY**

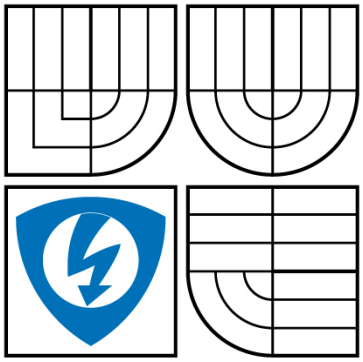
DIPLOMOVÁ PRÁCE

MASTER´S THESIS


AUTOR PRÁCE — **Bc. JOSEF ZIMEK**
AUTHOR

VEDOUCÍ PRÁCE — Ing. IVO HERMAN, CSc.
SUPERVISOR


BRNO 2007



**VYSOKÉ UCENÍ**
**TECHNICKÉ V BRNE**
**Fakulta elektrotechniky**
**a komunikacních technologií**
**Ústav telekomunikací**

# Diplomová práce

magisterský navazující studijní obor

**Telekomunikacní a informacní technika**

***Student:*** Zimek Josef Bc. ***ID:*** 89849

***Rocník:*** 2 ***Akademický rok:*** 2007/2008

**NÁZEV TÉMATU:**

## Spojovací systémy založené na IP telefonii

**POKYNY PRO VYPRACOVÁNÍ:**
Navrhnete komunikacní systém, jehož cílem je sdružit datový tok akustických signálu z více zdroju, tyto prenést pres internetovou sít a zase rozdelit na jednotlivé akustické signály. Systém musí být schopen komunikovat obousmerne a k akustickým signálum musí být pridružen minimálne jeden datový kanál s komunikacní rychlostí 32 kbit/s.

**DOPORUCENÁ LITERATURA:**
[1] Kallay, F., Peniak, P.: Pocítacové síte a jejich aplikace. 2. aktualizované vydání. GRADA, 2003.
[2] IP telephony: http://www.iptelephony.org/
[3] http://www.rad.com

***Termín zadání:*** 11.2.2008 ***Termín odevzdání:*** 28.5.2008
***Vedoucí práce:*** Ing. Ivo Herman, CSc.
**prof. Ing. Kamil Vrba, CSc.**
*predseda oborové rady*

# LICENČNÍ SMLOUVA
# POSKYTOVANÁ K VÝKONU PRÁVA UŽÍT ŠKOLNÍ DÍLO

uzavřená mezi smluvními stranami:

**1. Pan/paní**

Jméno a příjmení: Josef Zimek

Bytem: Práče 45

Narozen/a (datum a místo): 30.8.1984 ve Znojmě

(dále jen „autor“)

a

**2. Vysoké učení technické v Brně**

Fakulta elektrotechniky a komunikačních technologií

se sídlem Údolní 244/53, 602 00, Brno

jejímž jménem jedná na základě písemného pověření děkanem fakulty:

.....

(dále jen „nabyvatel“)

## Čl. 1
## Specifikace školního díla

1. Předmětem této smlouvy je vysokoškolská kvalifikační práce (VŠKP):
   - ~~□ disertační práce~~
   - □ diplomová práce
   - ~~□ bakalářská práce~~
   - ~~□ jiná práce, jejíž druh je specifikován jako~~ .....

   (dále jen VŠKP nebo dílo)

Název VŠKP: Spojovací systémy založené na IP telefonii

Vedoucí/ školitel VŠKP: Ing. Ivo Herman CSc.

Ústav: Ústav telekomunikací

Datum obhajoby VŠKP: .....

VŠKP odevzdal autor nabyvateli v*:

- □ tištěné formě – počet exemplářů ……2……..
- □ elektronické formě – počet exemplářů ……1……..

* hodící se zaškrtněte

2. Autor prohlašuje, že vytvořil samostatnou vlastní tvůrčí činností dílo shora popsané a specifikované. Autor dále prohlašuje, že při zpracovávání díla se sám nedostal do rozporu s autorským zákonem a předpisy souvisejícími a že je dílo dílem původním.
3. Dílo je chráněno jako dílo dle autorského zákona v platném znění.
4. Autor potvrzuje, že listinná a elektronická verze díla je identická.

## Článek 2
## Udělení licenčního oprávnění

1. Autor touto smlouvou poskytuje nabyvateli oprávnění (licenci) k výkonu práva uvedené dílo nevýdělečně užít, archivovat a zpřístupnit ke studijním, výukovým a výzkumným účelům včetně pořizovaní výpisů, opisů a rozmnoženin.
2. Licence je poskytována celosvětově, pro celou dobu trvání autorských a majetkových práv k dílu.
3. Autor souhlasí se zveřejněním díla v databázi přístupné v mezinárodní síti
   - □ ihned po uzavření této smlouvy
   - □ 1 rok po uzavření této smlouvy
   - □ 3 roky po uzavření této smlouvy
   - □ 5 let po uzavření této smlouvy
   - □ 10 let po uzavření této smlouvy

   (z důvodu utajení v něm obsažených informací)
4. Nevýdělečné zveřejňování díla nabyvatelem v souladu s ustanovením § 47b zákona č. 111/ 1998 Sb., v platném znění, nevyžaduje licenci a nabyvatel je k němu povinen a oprávněn ze zákona.

## Článek 3
## Závěrečná ustanovení

1. Smlouva je sepsána ve třech vyhotoveních s platností originálu, přičemž po jednom vyhotovení obdrží autor a nabyvatel, další vyhotovení je vloženo do VŠKP.
2. Vztahy mezi smluvními stranami vzniklé a neupravené touto smlouvou se řídí autorským zákonem, občanským zákoníkem, vysokoškolským zákonem, zákonem o archivnictví, v platném znění a popř. dalšími právními předpisy.
3. Licenční smlouva byla uzavřena na základě svobodné a pravé vůle smluvních stran, s plným porozuměním jejímu textu i důsledkům, nikoliv v tísni a za nápadně nevýhodných podmínek.
4. Licenční smlouva nabývá platnosti a účinnosti dnem jejího podpisu oběma smluvními stranami.

V Brně dne: ……24.5.2008…….

…….. …….Josef Zimek……

Nabyvatel Autor

## ANOTACE

Ve své práci jsem se zabýval návrhem a realizací komunikační sítě, která umožňuje dvěma nezávislým sítím využívat hlasové služby skrze šifrovaný kanál. Řešení je postaveno na routerech, které jsou tvořeny staršími PC s operačním systémem FreeBSD. Mezi routery je vytvořen statický šifrovaný tunel s využitím protokolu IPSec. Hlasové služby zajišťuje paketově orientovaná pobočková ústředna Asterisk s podporou signalizačního protokolu SIP. Toto řešení může sloužit např. pro připojení vzdálené pobočky k centrále firmy, které tak může využívat sdílené prostředky. K centrále se rovněž mohou připojit zaměstnanci z domova nebo na cestách. V tom případě se k autentizaci využívá SSL certifikátů. Tento způsob připojení je v dnešní době velmi žádánn.



## ABSTRACT

My master's thesis is focused on designing and creating communication network, which provides communication between two independent networks through encrypted tunnel. My solution is based on routers formed by older personal computers with FreeBSD like a operating system. Between routers is created static encrypted tunnel by using IPSec protocol. Voice services provides packet oriented exchange Asterisk with support of signaling protocol SIP. This solution can be used eg. for connecting remote branch to headquarters of company and then can branch utilize shrared resources. To headquarters can connect also remote workers from their home. In this case are used SSL certificates to authentication of user. This scenario is very required today.

# PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma "Spojovací systémy založené na IP telefonii" jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.“

V Brně dne ……24.5.2008……..
Josef Zimek

PODĚKOVÁNÍ

Děkuji vedoucímu diplomové práce Ing. Ivu Hermanovi, CSc., zaměstnanci Ústavu telekomunikací , za velmi užitečnou metodickou pomoc a cenné rady při zpracování diplomové práce.

**V BRNĚ DNE 24.5.2008** **JOSEF ZIMEK**

## OBSAH

Seznam obrázků



.

# 1. Úvod

## 1.1 Problematika komunikačních sítí

V poslední čtvrtině 20. století došlo k významným pokrokům v oblasti informačních technologií, což mělo za následek vybudování malých komunikačních sítí, které se postupně spojovaly až do podoby jakou známe dnes např. v podobě Internetu. V dnešní době existuje nepřeberné množství telefonních a počítačových sítí, které mají jeden primární účel – umožnit uživatelům z celého světa spolu komunikovat co nejsnadnějším způsobem. Milióny lidí i firem jsou na komunikaci mezi sebou přímo závislí, a proto je nutno se touto problematikou zabývat.

Trendem posledních let je vytvořit komunikační síť využívající jednotná pravidla pro různé služby. Tuto snahu o vytvoření jednotné komunikační sítě nazýváme „konvergence“. Cílem konvergence je zajištění vysoké efektivity využívání síťových zdrojů. První snahou o konvergenci bylo vytvoření digitální sítě integrovaných služeb, označované ISDN (Integrated Services Digital Network), která ve své době zaznamenala určitý „boom“, ale brzy se však ukázalo, že nároky na kapacitu přesahují hranice ISDN. Navíc tato technologie využívá pro přenos tzv. spojování okruhů, což má za následek pomalejší paketový přenos. Proto se tato koncepce dnes přestává používat.

Dalším krokem konvergence je technologie ATM (Asynchronous Transport Mode). Jedná se o síť s přepojováním datových jednotek o stejné velikosti. Komunikace má spojovaný charakter a má implementovánu podporu kvality služeb. Bohužel tato technologie je poměrně nákladná a komplikovaná. Zřejmě proto nedosáhla takového rozšíření jako sítě založené na protokolu IP (Internet Protocol). IP sítě využívají k přenosu tzv. přepojování paketů, kdy každý paket postupuje sítí tou cestou, která je v danou chvíli nejefektivnější. Využívá se sdílení síťových prostředků což umožňuje rozložení zátěže provozu.

V neposlední řadě se o konvergenci zasloužil Internet, který se během krátké doby rozšířil prakticky po celé Zemi. Tento masivní nárůst uživatelů samozřejmě způsobil poptávku po nejrůznějších službách. Tyto služby kladou na síť rozdílné požadavky. Datový přenos je charakteristický proměnnými nároky na šířku pásma a na spolehlivost spojení. Hlas a video naproti tomu požadují relativně konstantní pásmo a garantovanou dobu doručení, částečná ztráta informace do určité míry není důležitá a lze ji kompenzovat různými opravnými metodami. Těmto parametrům je přizpůsobena i konstrukce příslušných sítí. V datových sítích je ponechána každému

možnost maximálně využít dostupné pásmo. Tím je kapacita linek efektivně využita, není ale zaručena doba doručení. V telefonních sítích se naproti tomu pro každý hovor pásmo rezervuje a to bez ohledu na to, zda se na lince něco přenáší. Je tak sice zaručena doba doručení, ale pásmo se nevyužívá efektivně.

Konvergence tedy znamená ucelení různých typů služeb tak, aby byly efektivně využity v rámci jedné společné technologie – v IP síti.

Většina firem a podniků, ať velkých či menších, má v dnešní době potřebu efektivně komunikovat směrem ke svým pracovníkům, obchodním partnerům i zákazníkům. Také velký nárůst poskytovatelů připojení k Internetu znamená jeho snadnou dostupnost široké veřejnosti, která tím pádem má možnost využívat nejrůznější typy služeb. To je další příčina pro rozvoj komunikačních systémů a služeb. Jedním typem služby, kterým se zde budu podrobněji zabývat je tzv. VoIP (Voice over IP) - hlasová komunikace přes IP sítě (obecně IP telefonie).

# 2. Co je VoIP

## 2.1. VoIP obecně

Voice over IP [28],[29]- jedná se o přenos hlasu prostřednictvím datových sítí založených na protokolu IP, který je dnes standardem počítačových sítí, doplněný o QoS [19] (Quality of Services). Hlas je tedy přenášen v jedné síti společně s jinými daty. Pro uživatele je tento hovor identický s hovorem, na který je zvyklý z běžného telefonu. VoIP pracuje na paketovém principu a k přenosu využívá komprimovaný digitalizovaný hlas. Je alternativou ke klasické telefonii, založené na přepojování okruhů.

## 2.2.Výhody VoIP :

- Nízké náklady
- Jednotná komunikační síť
- Efektivnější využití již vybudovaných počítačových sítí
- Mobilita účastníka
- Možnost zavádění QoS pro přenos hovorových signálů do lokálních i rozlehlých počítačových sítí
- Efektivnější administrace

## 2.3. Situace v ČR

V České Republice v posledních letech můžeme zaznamenat velmi vysoký nárůst poptávky po komunikačních službách. Dříve to byla doména spíše rozsáhlejších firem, ale s rozšířením dostupnosti připojení k Internetu do českých domácností se otevírá každému možnost využívat nepřeberné množství služeb. Není tedy divu, že k nejoblíbenějším činnostem českého uživatele Internetu patří : vyhledávání informací, nakupování, ilegální sdílení dat a komunikace ve formě emailů, Instant Messaging a VoIP.

Tomuto trendu přispívá zejména snižování cen síťového hardwaru což má za následek vznik nových malých firem, které se zabývají poskytováním připojení k Internetu (ISP). Díky tomu může vzniknout ISP i v lokalitách, kde doposud připojení k Internetu nikdo neposkytoval nebo bylo realizováno firmou, která byla pro danou lokalitu monopolem. Tímto způsobem jsou vytvářeny vhodné podmínky pro konkurenční boj a tedy i snižování cen za připojení. Také na českém trhu vznikají noví operátoři, kteří se přímo soustřeďují na IP telefonii a nabízejí hlasové služby za nižší ceny než konkurence. To je důkazem, že můžeme očekávat dynamické rozšíření VoIP technologie do českých domácností.

V tabulce 1 je znázorněna situace připojení českých domácností k Internetu pro 2. čtvrtletí 2007 – hodnoty udávají procentuální zastoupení domácností, které mají připojení k Internetu. Je však nutno zdůraznit, že některé typy připojení nejsou vhodné pro přenos hlasu a dat citlivých na zpoždění. Z tabulky je patrné, že existuje stále velké procento uživatelů, kteří používají stávající telefonní síť tzn. nepoužívají VoIP. Také většinový podíl připojení patří technologii xDSL, která však v sobě implementuje možnost komunikace přes klasickou telefonní sít. Dalším typem připojení, které má u nás velké zastoupení především díky snadné realizaci a mobilitě, jsou bezdrátové technologie (Wi-Fi 2,4GHz). Ty jsou však náchylné na rušení a kolísání zpoždění čímž může být neblaze ovlivněna kvalita hovoru. Kompenzací může být např. použití pásma 5GHz nebo licencovaného pásma WiMAX. Připojení přes kabelovou televizi přináší konstantní zpoždění takže je pro VoIP vhodné.

| **Způsob připojení** | **Domácnosti v %** |
|---|---|
| Standardní telefonní linka | 18,30% |
| ISDN linka | 3,00% |
| ADSL nebo jiné DSL technologie | 26,00% |
| Kabelová TV | 22,70% |
| Bezdrátové připojení (WLAN, Wi-Fi, WiMAX) | 22,30% |
| Nízkorychlostní mobilní připojení (GPRS, HSCSD) | 1,30% |
| Připojení přes mobilní telefon - vysokorychlostní | 7,10% |
| Jiný typ vysokorychlostního připojení | 6,10% |

Tab.1: Způsob připojení domácností k Internetu – 2. čtvrtletí 2007 [7]

## 2.4. Ekonomické aspekty

Srovnáme-li použití klasické telefonní sítě a IP telefonie, určitě nalezneme klady i zápory na obou stranách. Avšak jednou z vlastností, kterou lákají VoIP operátoři své potencionální klienty je cena. Základní myšlenkou je : „proč platit za připojení k Internetu a zároveň za připojení k telefonní síti, když lze volat přes Internet?“. Jak jsem již zmínil výše, díky vysoké dostupnosti konektivity, snadná realizace připojení už není výsadou jen telefonní sítě.

Proto lze využít stávající počítačovou síť a nejsou potřeba žádné speciální telefonní rozvody. VoIP umožňuje připojit hardwarové IP telefony přímo ke standardním síťovým prvkům. Také lze využít softwarovou formu IP telefonů a pobočkových ústředen, které jsou dostupné také zdarma (open source SW), což významným způsobem příznivě ovlivňuje náklady na výstavbu síťové infrastruktury. Dalším aspektem je škálovatelnost a rozšiřitelnost sítě. U telefonní sítě většinou platí, že čím více se přidává linek nebo přípojek, tím jsou zapotřebí dražší rozšíření hardwaru. V některých případech je nakonec třeba pořídit zcela nový telefonní systém. V případě telefonního systému VoIP nic takového nehrozí. Běžný počítač dokáže snadno zvládnout velké množství telefonních linek.

VoIP architektura také umožňuje snadný roaming a přenositelnost čísla. Uživatel může vzít svůj telefon, připojit jej do nejbližšího ethernetového portu a ponechat si své číslo. Ale zřejmě nejpodstatnějším rysem systémů založených na principu VoIP je snížení nákladů na volání. Díky příznivé cenové politice VoIP operátorů lze využíváním jejich služeb ušetřit značné prostředky na volání jak v rámci České Republiky tak i do zahraničí. Pokud se například firma rozhodne propojit VoIP systémy mezi svými pobočkami a obchodními partnery, může volat zdarma nejen v rámci konkrétní pobočky, ale i vzájemně mezi dílčími pobočkami . Toto řešení se v současnosti stále více využívá a proto se jím budu dále zabývat i ve své práci.

# 3. Základní principy IP telefonie

## 3.1. Převod hlasu na digitální informaci

Ve VoIP sítích je hlas po síti přenášen ve formě jedniček a nul, ale samotná řeč vycházející z úst má analogový (spojitý) charakter. Je tedy nutné aby došlo k určitému převodu do binárního tvaru a to takovým způsobem, aby jej bylo možno po přenosu převést zpět do analogové podoby. Tento převod spočívá ve třech základních krocích : vzorkování, kvantování, kódování [5].

### 3.1.1. Vzorkování - pulsně amplitudová modulace

Proces vzorkování analogového signálu, jak název napovídá, spočívá v odebírání určitého počtu vzorků hlasu, tak aby se zachovala souvislost řeči, ale byly odstraněny redundantní informace. Vzorkovací frekvence musí být zvolena tak, aby nedocházelo k jevu zvanému aliasing. Při nedostatečném počtu vzorků může být totiž výsledná informace deformovaná nebo v horším případě úplně znehodnocená. Naopak při nadbytečném počtu vzorků spotřebujeme větší šířku pásma než potřebujeme a to v dnešní době, kdy šířka pásma je téměř tak cennou komoditou jako kyslík a voda, je velice neefektivní.

Vzorkovací frekvence byla stanovena definicí známou jako Shannonův teorém (někdy též označován jako Nyquistův teorém, Nyquistův-Shannonův teorém, Shannonův-Nyquistův-Kotělnikův teorém), který udává, že spojitý signál je možné spolehlivě rekonstruovat pouze tehdy, byl-li vzorkován frekvencí alespoň dvakrát vyšší než je maximální hodnota původního signálu. Lidský sluch je schopen vnímat zvuky až do frekvence 20KHz, ale samotná lidská řeč obsahuje frekvence v pásmu 0,3 až 3,4KHz tudíž vzorkovací frekvence je zvolena na 8KHz tedy vzorek je odebrán každou 1/8000 sekundy.

### 3.1.2. Kvantizace

V tuto chvíli již máme z analogového signálu odebrány vzorky a potřebujeme jim přiřadit hodnoty, které budou představovat jejich amplitudy. Tento proces je nazýván kvantizace a spočívá v zaokrouhlování amplitud na hodnoty dle stupnice. Zaokrouhlování způsobuje kvantizační chybu, která se na lince projevuje „sykotem“. Tento sykot se projevuje především u nižších amplitud, které se vyskytují v řeči častěji než vysoké amplitudy. Proto můžeme odebírat více vzorků při nižších intenzitách a méně vzorků při vyšších intenzitách, čímž potlačíme kvantizační chybu aniž bychom využili větší šířku pásma. K dosažení tohoto výsledku se pro zaokrouhlování amplitud používá logaritmická stupnice a proto se tomuto typu kvantizace říká „logaritmická kvantizace“. Existují dva typy logaritmické kvantizace označované jako „α-Law“(Severní Amerika, Japonsko) a „μ-Law“(ostatní země).

Logaritmická stupnice je rozdělena 3 segmentů a každý segment se dělí dále na tzv. kroky. Při přiřazování hodnot je každému vzorku přidělena polarita (1 bit), segment (3 bity) a krok (4 bity). Každý vzorek je tedy reprezentován osmi bity a je odebíráno 8000 vzorků za vteřinu čili potřebujeme šířku pásma 64kb/s (pouze pro hlas).

### 3.1.3. Kódování

Jak jsem již zmínil výše, šířky pásma není nikdy dost, a proto se snažíme co nejvíce „stlačit“ přenášené informace. K tomu slouží proces kódování a dekódování, který je realizován pomocí kodeků. Účelem kodeků tedy je komprimovat přenášené hlasové informace.

Typy kodeků:

- Pulsně-kódová modulace PCM - Ve skutečnosti nekomprimuje analogový signál, ale vzorkuje a provádí kvantizaci způsobem popsaným výše. Tuto modulaci provádí kodek G.711

- Adaptivní diferenční pulsně-kódová modulace (ADPCM) - používá tzv. diferenční signál tzn. jsou přenášeny pouze rozdíly aktuálního vzorku oproti předchozímu. Příkladem kodeku ADPCM je G.726.

- CS-ACELP (Conjugate Structure Algebraic Code Excited Linear Predication) - tato metoda komprese je zajímavá tím, že na základě vzorků řeči vytváří tzv. knihu kódů. Poté pomocí vyrovnávací paměti (look ahead buffer) zjišťuje jestli se aktuální vzorek nachází v knize kódů. Pokud ano nepřenáší se celý vzorek, ale pouze informace o jeho pozici (kód) v knize kódů. Tento princip se uplatňuje nejvíce v situacích kdy se opakuje určitá část slov daná jazykovou skladbou např. slova vzorková**ní**, kvantová**ní**, kódová**ní**, tudíž jinou strukturu budou mít knihy kódů pro různé jazyky a zároveň se budou lišit knihy kódů např. při hovoru obchodních partnerů a hovoru dítěte s rodičem. Tuto metodu využívá kodek G.729, který se pro svou kvalitu a nízké požadavky na šířku pásma (8kb/s) hojně využívá v sítích typu WAN.

  Existují dva typy tohoto kodeku G.729a a G.729b. Kodek G.729a používá jednodušší algoritmus pro šetření prostředků procesoru za cenu nepatrného snížení kvality hovoru. Kodek G.729b umožňuje detekci hlasové aktivity (VAD), která během doby kdy nikdo nemluví po 250ms zastaví přenos dat, čímž ve výsledku lze touto metodou ušetřit až 30% šířky pásma.

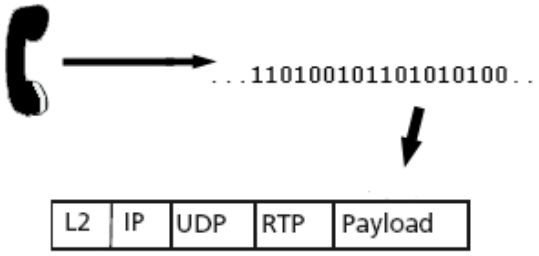
- LD-CELP (Low-Delay Conjugate Excited Linear Predication) – tento kodek je velmi podobný kompresi typu ACELP avšak používá menší kódovou knihu, což má za následek menší zpoždění, ale na druhou stranu vyžaduje větší šířku pásma. Příkladem je kodek G.728.

Doposud jsem hovořil o šířce pásma, kterou zabírá pouze samotný přenos hlasu. V praxi však musíme uvažovat daleko více faktorů, které ovlivňují velikost paketů. Jsou to např. typ přenosové technologie (Ethernet, ATM, Frame Relay), dále zda se komprimují informace v hlavičce či ne, zda je hlasový provoz realizován prostřednictvím VPN tunelu, jaký kodek je vybrán a podobně.

## 3.2. Přenos hlasu po síti

### 3.2.1. Struktura VoIP paketu

V této fázi jsme již získali digitalizovaný hlas, tedy užitečná data, která budeme přenášet sítí. V sítích založených na protokolu IP je základní přenosovou entitou IP paket. Data jsou však přenášena prostřednictvím datagramů i jiných vrstev. Na obrázku 1. je znázorněna struktura VoIP paketu s hlavičkami dílčích protokolů.




Obr.1 : Struktura VoIP paketu [5]

Nejnižší vrstvou jejíž datovou jednotku využijeme při přenosu je spojová (data linková) vrstva. Základní entitou je rámec, který je tvořen hlavičkou, datovou částí a zakončením rámce (trailer). Hlavička obsahuje zdrojovou a cílovou MAC adresu a typ protokolu vyšší vrstvy. Datová část v sobě nese hlavičku vyšší vrstvy a vlastní přenášená data.

| Preamble | SFD | Source MAC Address | Destination MAC Address | Type | data | FSC |
|---|---|---|---|---|---|---|
| 7B | 1B | 6B | 6B | 2B | 46–1500B | 4B |

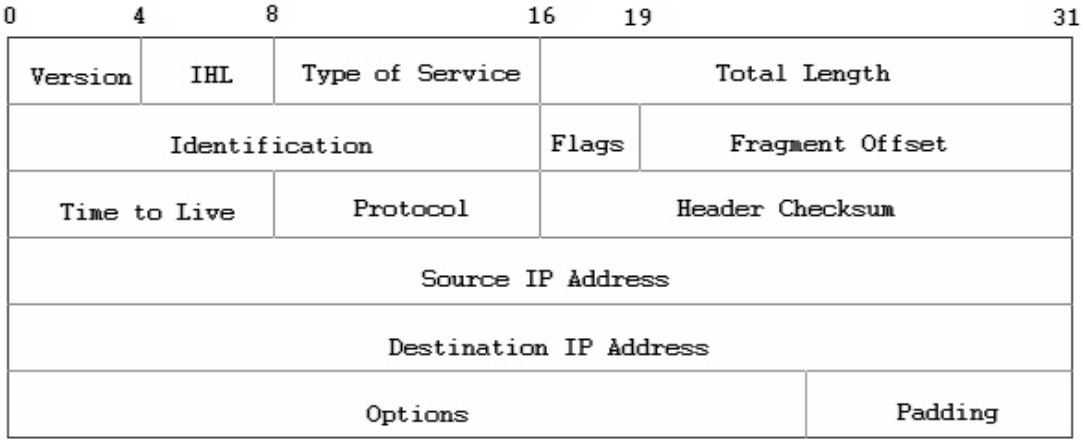
Obr.2 : Struktura hlavičky Ethernet rámce [1]

**Význam jednotlivých polí hlavičky Ethernet rámce :**

- **Preamble –** slouží k synchronizaci vysílající stanice a přijímajících stanic (56 bitů)
- **SFD –** Start Frame Delimiter, 8-bitová hodnota značící konec preamle (8 bitů)

- **Source MAC Address –** zdrojová MAC adresa (48 bitů)
- **Destination MAC Address –** MAC adresa příjemce (48 bitů)
- **Type –** toto pole indikuje, který protokol je přenášen v rámci např. 0x0800 pro IPv4 (16 bitů)
- **data (payload) –** vlastní přenášená data
- **FCS –** frame check sequence, kontrolní výpočet pro detekci a korekci chyb (24 bitů)

Dalším datagramem sloužícím pro přenos hlasu je IP paket. Pracuje na síťové vrstvě modelu TCP/IP a skládá se z vlastních přenášených dat, ke kterým je připojena 32-bitová hlavička. Hlavička paketu obsahuje informace, podle kterých aktivní prvky sítě určují cestu paketu skrze síť. Obecná struktura IP paketu je znázorněna na následujícím obrázku.



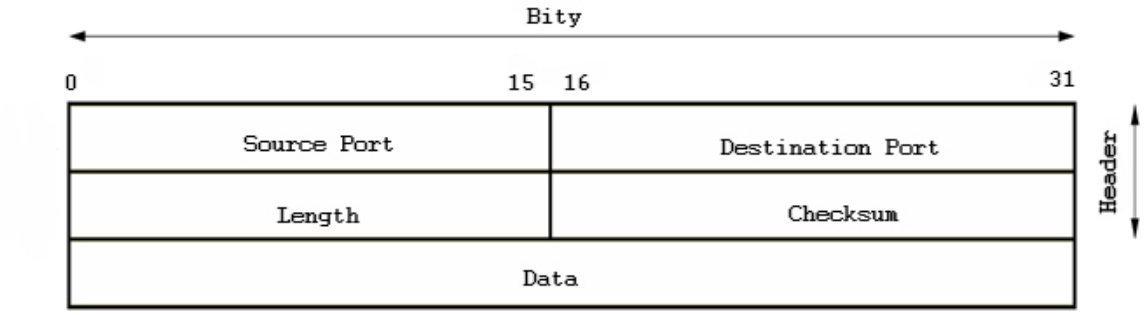

Obr.3 : Struktura hlavičky IP paketu [10], [29]

**Význam jednotlivých polí hlavičky IP paketu :**

- **Version –** určuje verzi hlavičky IP paketu (4 bity)
- **IHL –** Internet Header Length, délka hlavičky ve 32-bitových slovech. Správná hodnota je 5 (4 bity)
- **Type of Service –** tento parametr se používá pro detekci aktuální služby (8 bitů)
- **Total Length –** celková délka paketu včetně hlavičky i datové části (16 bitů)
- **Identification –** tato hodnota spolu se zdrojovou IP adresou jednoznačně identifikuje paket. Používá se při zpětném sestavování paketu na přijímající straně (16 bitů)

- **Flags –** sekvence tří znaků pro fragmentaci datagramu (3 bity)
- **Fragment Offset –** toto pole indikuje, kam patří fragment v daném rámci (13 bitů)
- **Time to live –** maximální doba doručení paketu (8 bitů)
- **Protocol –** indikuje typ protokolu na transportní vrstvě (8 bitů)
- **Header Checksum –** kontrolní součet pro kontrolu hodnot hlavičky (16 bitů)
- **Source IP Address –** zdrojová IP adresa (32 bitů)
- **Destination IP Address –** IP adresa příjemce (32 bitů)
- **Options –** přídavné možnosti, běžně se toto pole nepoužívá

Samotný IP paket nám však, ale k přenosu nestačí, protože je vyžadována spolupráce dalších vrstev TCP/IP modelu. Následující vyšší vrstva je transportní, která využívá protokoly TCP a UDP. Pro přenos hlasu se používá UDP (User Datagram Protocol), který zajišťuje nespolehlivý přenos tzn. nezaručuje, zda se přenášený paket neztratí, nezmění se pořadí paketů, nebo zda některý paket nebude doručen vícekrát. Na druhou stranu díky tomu, že odpadá režie na potvrzování zpráv, je protokol UDP rychlý, efektivní a proto je vhodný pro aplikace citlivé na zpoždění. Základní přenosovou jednotkou UDP protokolu je UDP datagram, který je složen z hlavičky a přenášených dat. Obr.2 představuje strukturu UDP hlavičky.

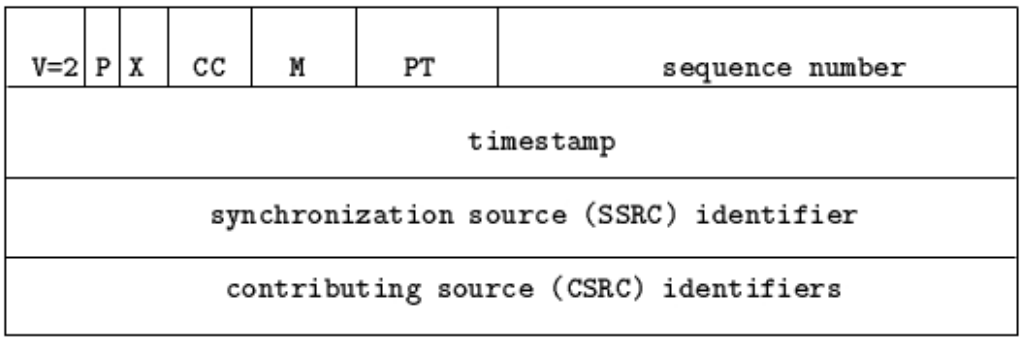



Obr.4 : Struktura hlavičky UDP datagramu [10], [29]

**Význam jednotlivých polí hlavičky UDP datagramu :**

- **Source port –** specifikace číslo portu aplikace, která vysílá uživatelská data (16 bitů)
- **Destination port –** číslo portu náležící přijímající aplikaci (16 bitů)
- **Length –** celková délka datagramu včetně hlavičky i přenášených dat (16 bitů)
- **Checksum –** kontrola integrity dat (16 bitů)

Dalším typem protokolu, který je použit pro vlastní přenos hlasu je RTP [16] (Real-time Transport Protocol), který je navržen pro streamování multimediálních dat v reálném čase. Je v podstatě nadstavbou protokolu UDP. Oproti UDP zajišťuje navíc doručování paketů ve správném pořadí, toho se dosahuje pomocí číslování paketů a časových razítek. Protokol RTP také umožňuje kompresi na úrovni multimediálních dat i na úrovni zmenšení objemu záhlaví.




Obr.5 : Struktura záhlaví RTP datagramu [18]

**Význam jednotlivých polí hlavičky RTP datagramu :**

- **version (V)–** indikuje verzi protokolu RTP (2 bity)
- **padding (P)–** záhlaví – je-li tento bit nastaven tak datagram obsahuje za datovou částí zápatí, které udává ukončení datagramu (1 bit)
- **extension (X)–** rozšíření hlavičky dle definovaného formátu. Většinou není nastaveno (1 bit)
- **CSRC count (CC) –** počet CSRC identifikátorů, které následují po opravené hlavičce (4 bity)
- **marker (M) –** značí hranici rámce ve streamu paketů (1 bit)
- **payload type (PT) –** typ zátěže – určuje formát přenášených dat (7 bitů)
- **sequence numer –** sekvenční číslo, s každým dalším datagramem se zvyšuje o 1. Používá se pro detekci ztracených datagramů a zajištění správného pořadí (16 bitů)
- **timestamp –** hodnota pro zajištění synchronizace a detekci změny zpoždění (32 bitů)
- **SSRC –** identifikátor zdroje synchronizace, je vybírán náhodně a je společný pro každou RTP relaci (32 bitů)
- **CSRC –** identifikátor zdrojů přenášených dat (u více zdrojů). Maximum 15 zdrojů (32 bitů pro každý zdroj)

Nyní už známe strukturu jednotlivých datových jednotek, které potřebujeme pro přenos hlasu přes IP síť. Na obr.5 je znázorněna struktura VoIP paketu, který je složen z dílčích výše uvedených datových jednotek.

## 3.3 Signalizační protokol SIP

Doposud byla řeč pouze o komunikačních protokolech [16] (RTP), které slouží k vlastnímu přenosu hlasu. Dalším typem jsou protokoly, které se používají k navazování spojení, řízení toku a jeho ukončování. Mezi signalizační protokoly patří H.323, SIP, MGCP, Megaco.

Doporučení H.323 patří do skupiny standardů H.32x, které řeší problematiku multimediálních přenosů v sítích založených na IP protokolu. Určuje standardy pro komunikaci v sítích, které neposkytují zaručenou kvalitu služeb (QoS), a které nejsou vzájemně kompatibilní. Standard H.323 však sám o sobě obsahuje další dílčí protokoly, normy a doplňky, které jsou určeny např. pro kódování, signalizaci nebo pro přenos zvuku a videa.

SIP (Session Initiation Protocol) [1],[17] je signalizačním protokolem pro sestavení, řízení a ukončení spojení mezi účastníky komunikace. Pracuje na bázi klient-server. Při srovnání s protokolem dle standardu H.323 je SIP jednodušší a přesto má mnohem širší škálu využití. Jeho jednoduchost spočívá v tom, že je textově orientovaný a tím pádem umožňuje podstatně jednodušší protokolovou analýzu bez potřeby drahých analyzátorů. Svojí strukturou je podobný protokolům HTTP a SMTP. Struktura zprávy SIP je tvořena hlavičkou a vlastním tělem zprávy. Protokol jako takový nemá přesně definovaný typ informace, kterou přenáší, proto má ve své struktuře zapouzdřen další protokol např. SDP (Session Description Protocol), který určuje všechna konfigurační data (kódování, adresu, port, jméno a účel relace, doba aktivního spojení, kontaktní informace) pro následující přenos informace.

### 3.3.1 Komponenty sítě pro VoIP dle standardu SIP

- SIP – User Agent
- SIP – Servery

**SIP-User agent** – koncové zařízení, které má na starost sestavování spojení s dalšími UA. Jedná se zejména o SIP telefony ve formě klasického telefonu nebo SW aplikace na počítači a brány do jiných sítí. Klientskou část v UA zajišťuje User Agent Client (UAC), který má na starosti inicializaci spojení. Serverovou část tvoří User Agent Server (UAS), který reaguje na příchozí žádosti a posílá odpovědi. UA má buď pevně nastavenou adresu Proxy či Redirect serveru, nebo si ji musí vyžádat z DNS serveru.

**SIP-Servery** - jsou zařízení, jejichž úkolem je zprostředkovat komunikaci mezi UA. Rozlišujeme tyto typy SIP serverů:

- Proxy server - tento server přijme žádost o spojení od UA nebo od jiného proxy serveru a předá ji dalšímu proxy serveru (pokud volanou stanici nemá ve své správě)
- Redirect server - stejně jako proxy přijímá žádosti o spojení od UA nebo proxy serverů, ale nepřeposílá je dále ve směru volaného. Posílá tázajícímu informaci, komu má žádost poslat, aby se dostala k adresátovi.
- Registrar server - přijímá registrační žádosti od UA a aktualizuje podle nich databázi koncových zařízení (location service), které jsou v rámci domény spravovány.
- Location Server - server zajišťující službu zjišťování aktuální adresy volaného

Volající klient může kontaktovat volaný server přímo nebo prostřednictvím jiného serveru, který pracuje jako proxy nebo redirect server. Přímé volání se používá, pokud klient zná IP adresu serveru, u kterého se nachází volaný uživatel.

**Typické signalizační zprávy :**

INVITE – žádost o navázání spojení nebo o změnu parametrů již existujícího spojení

BYE – žádost o rozpojení spojení

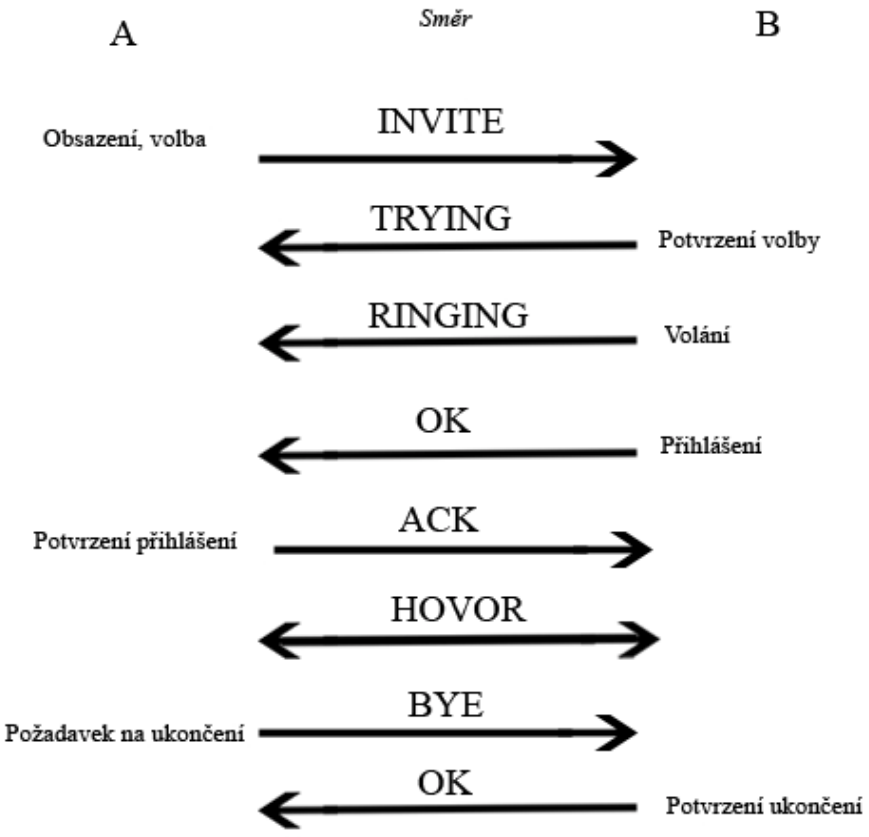
ACK – žádost, kterou klient potvrzuje, že obdržel odpověď na žádost INVITE

REGISTER – žádost o registraci klienta na registrar serveru

CANCEL – žádost o zrušení probíhající žádosti INVITE

OPTIONS – žádost o zaslání podporovaných funkcí na serveru

INFO – přenos informací během hovoru (rozšíření protokolu SIP popsané v RFC2976)




Obr.6 : SIP signalizace během hovoru [4]

# 4. VoIP a Quality of Service (QoS)

Pojem kvalita služby (Quality of Service, QoS) [19],[20] vyjadřuje jeden ze směrů vývoje technologií a služeb počítačových sítí - poskytovat uživatelům služby s definovanou kvalitou. Rychlý rozvoj služeb, které kladou na síť rozdílné požadavky, vyžaduje aby byla uživateli poskytnuta určitá kvalita těchto služeb.

Například multimediální aplikace jako videokonference nebo VoIP jsou citlivé na zpoždění. Lidské ucho je nedokonalé a zpoždění v desítkách milisekund není schopno zaznamenat. Větší zpoždění se však projevuje jako ozvěna. Opačné požadavky na síť má přenos souborů či e-mailů. Zpravidla není důležité, zda se soubor přenese za pár sekund nebo za pár minut. Důležité je, aby se přenesl bez ztráty jediného bitu. Mail odeslaný s velkou přílohou dokáže zahltit celou dostupnou šířku pásma. Obchodní aplikace většinou nemají příliš velké požadavky na šířku pásma, ale jsou citlivé na zpoždění. Tyto požadavky lze splnit pouze pokud se vhodným způsobem mapují parametry QoS.

## 4.1 Nejvýznamnější parametry QoS [1],[20]:

- **Šířka pásma** (bandwith) - představuje kapacitu pro přenos dat, vyjadřuje se většinou v kilobitech za sekundu (kbit/s) nebo megabitech za sekundu (Mbit/s); představuje teoretickou maximální kapacitu spojení, s jeho zvyšováním může být za stejnou jednotku času přenesen větší objem dat.
- **Ztrátovost paketů** (packet loss) - je vyjadřována většinou procenty a představuje průměrnou ztrátu paketů za určité období.
- **Průchodnost** (clearness)- objem dat v bajtech přenesený za jednotku času.
- **Zpoždění** (delay) - je čas, který uplyne od odeslání zprávy zdrojovým uzlem po její přijetí na uzlu cílovém; zahrnuje zpoždění v přenosové trase, zpoždění způsobená průchodem zařízením, kompresí/dekompresí atd; jsou aplikace, kterým zpoždění nevadí, ale jsou i aplikace na zpoždění náchylné.
- **Změna zpoždění** (jitter) - představuje variabilitu v doručování paketů cílovému uzlu (tedy ve zpoždění při přenosu); stejně jako u zpoždění i zde lze nalézt typy aplikací, které nemají s rozptylem problém, a naopak i aplikace, u kterých problém působí.
- **Dostupnost** (availability) - je vyjadřována většinou procenty a představuje průměrnou dostupnost služby za určité období ( např. 99,9 % pracovní doby).

- **Chybovost** (error rate) – procentuální vyjádření chybných přenosů.

QoS může být implementováno v různých vrstvách v rámci modelu počítačové sítě. Imementace se používá buď na úrovni ATM (je-li technologie ATM) nebo na úrovni protokolu IP (síťová vrstva). Při implementaci na úrovni protokolu IP existují tyto řešení:

- IS (Integrated Services, IntServ) - integrované služby
- DS (Differentiated Services, DiffServ) - diferencované služby
- MPLS (Multi-Protocol Label Switching) - přepínání paketů podle návěští
- SBM (Subnet Bandwidth Management) - správa přenosové kapacity v podsítích

## 4.2 Techniky zajištění QoS

**Integrated services –** tato metoda přístupu umožňuje zavádění rozdílných tříd přenosových služeb a poskytuje nejvyšší úroveň QoS v IP sítích. Aplikace si mohou rezervovat přenosové prostředky dle svých požadavků. Díky této vysoké garanci služeb je IS složitější a tím pádem náročnější na síťové prvky i aplikace. A protože se značné množství informace ukládá ve směrovačích, nehodí se tato metoda pro páteřní sítě, ale spíše pro LAN.

**Differentiated services –** diferencované sužby v IP sítích zajišťují předvídatelnou kvalitu přenosových služeb. Pakety se rozlišují podle druhu poskytovaných služeb a ne podle datových toků, jako tomu je u IS. Oproti IS je tato metoda poměrně jednoduchá a garance QoS jsou dlouhodobé a statické. Před vlastním přenosem dat není nutné nastavovat garance jednotlivým datovým tokům, je však třeba použít směrovače, které podporují plánování a řízení front, řízení priorit odchozích paketů a řízení délky fronty. DS neklade vysoké nároky na síťové prvky a proto jsou implementovány i v páteřních směrovačích.

**Multi-Protocol Label Switching -** přepojování paketů řízené návěstím se snaží o standardizované, zjednodušené a zrychlené směrování při propojení směrovaných a přepojovaných sítí. Přepínání paketů je založené na návěstích (label) a probíhá na základě směrovacích informací IP protokolu na směrovačích i přepínačích (mezi 2. a 3. vrstvou RM OSI –nahrazuje směrování přepínáním). Směrovače, které podporují MPLS se označují jako LSR (Label Switching Router).

**Subnet Bandwidth Management -** správa přenosové kapacity v podsítích se zaměřuje na řízení toku dat a zajištění QoS v prostředí sítí LAN. Na 2. vrstvě LAN sítí definuje pravidla pro prioritizaci rámců. Rámce se označí podle druhu poskytovaných služeb, poté se prioritně doručí rámce aplikací citlivých na přenosové zpoždění. Požadavky na garanci QoS v LAN sítích (především Ethernet) vedly k zavedení těchto norem :

- 802.1p je klíčový standard, definuje vytvoření 8-mi stupňové prioritizace na sítích Ethernet a Token Ring.
- 802.1q definuje vytváření virtuálních sítí LAN – VLAN.
- 802.1d definuje vlastnosti mostů sítí Ethernet.

Vzhledem k tomu, že výše zmíněné metody se aplikují na úrovni různých vrstev, lze jejich vhodnou kombinací dosáhnout efektivnějšího garantování QoS. Způsob konfigurace, kombinace a spolupráce metod pro zajištění QoS (IS, DS, MPLS, SBM) se nazývá architektura QoS a slouží k zajištění síťové politiky.

# 5. Návrh řešení

Při zkoumání problematiky VoIP sítí jsem se rozhodl pro návrh sítě, pomocí které bude umožněno komunikovat firmě s její vzdálenou pobočkou prostřednictvím sítě Internet a bude tak efektivněji využívat šířku pásma, přičemž odpadnou poplatky za provoz telefonních linek. Tato situace je v České Republice je poměrně rozšířená. Firma platí konektivitu k Internetu formou měsíčního paušálu a zároveň měsíční poplatky za připojení ke klasické telefonní síti, kterou využívá např. pouze pro komunikaci se vzdálenou firemní pobočkou. Velký počet firem také v dnešní době využívá pro komunikaci se zákazníky mobilních telefonů, což vede ke stále menšímu použití klasické telefonní sítě. Taková neefektivita je z hlediska managementu firmy nežádoucí a je potřeba ji eliminovat.

## 5.1 Návrh sítě

Návrh sítě je analytický proces. Ve valné většině případů se ovšem návrhem sítě myslí nejen zpracování návrhu budoucí počítačové infrastruktury, ale i její samotná realizace a uvedení do provozu. Požadavky na počítačové sítě se velice různí, jak podle velikosti firmy a počtu uživatelů, kteří ji budou využívat, tak podle účelu ke kterému má sloužit. Nelze proto předem zobecňovat principy, kterými se návrh sítě řídí. Zjednodušeně lze ale říci, že se jedná o návrh konkrétní infrastruktury a určení kolik a jakých hardwarových či softwarových prostředků bude potřeba, aby síť nebyla zbytečně naddimenzovaná či naopak kapacitně nedostačující.

### 5.1.1 Základní postup při návrhu:

- Analýza současného stavu – je rozdíl zda navrhujeme síť od základu nebo se provádí redesign stávající fungující sítě.
- Zjištění požadavků firmy – požadovaný typ služeb, počet uživatelů, kapacita.
- Výběr technologií, hardware a software
- Návrh topologie
- Finanční odhad
- Návrh konceptu a jeho konzultace s odpovědným představitelem firmy
- Zpracování dokumentace
- Realizace

### 5.1.2 Požadavky na šířku pásma

Při analýze jsem vycházel z předpokladu, že návrh bude realizován pro novou síť, tudíž odpadají problémy spojené s komplikacemi při navýšení kapacity provozu, tedy výměna stávajících zařízení, které např. nepodporují novější technologie, na kterých je návrh založen a podobně. Návrh je koncipován pro firmu s řádově max. několik desítek uživatelů, jde tedy o tzv. Small bussines solution. Samozřejmostí je ponechání rezervy pro případné pozdější rozšíření kapacity.

Při výpočtu šířky pásma je nutno zjistit předpokládanou vytíženost linek. To je snadnější u již vybudovaných sítí, kde je možné jednoduše z měsíčních výpisů vyčíst počet provolaných minut. U návrhu nové sítě se musí vycházet z tabulkových hodnot. Ze statistických údajů vychází, že průměrný měsíc má 22 pracovních dnů a během nich je cca 15% hovorů uskutečněno v nejvytíženější hodině dne. Podělíme-li tedy počet provolaných minut v měsíci (Ppm) hodnou 22 a výsledek vynásobíme 0,15 dostaneme počet minut volání v nejvytíženější hodině (Pnh). Vzhledem k tomu, že hodina má 60 minut, podělíme získaný počet minut v nejvytíženější hodině 60-ti a získáme hodnotu Erlang, kterou dále využijeme při výpočtu šířky pásma.

$$Pnh = (Ppm/22)*0,15 \ [m] \qquad [5]$$

$$Erlang = Pnh/60 \ [h] \qquad [5]$$

Pro další výpočet je potřeba software Erlang B Calculator, do kterého se zadá právě získaná hodnota Erlang a hodnota GoS. Gos (Grade of Service) [5] tzv. kvalita obsluhy udává přijatelné procento hovorů, které mohou být během nejvytíženější hodiny odmítnuty. Je to především z důvodu prevence proti předimenzování kapacity sítě, jelikož mimo nejvytíženější hodinu je počet hovorů menší. Po vložení hodnoty Erlang a GoS dostaneme z Erlang B Calculatoru počet linek potřebných pro uskutečnění daného objemu hovorů.

Hodnotu počtu linek budeme potřebovat pro další SW, tentokrát z dílny společnosti Cisco, který přímo z vybraných parametrů a počtu linek spočítá požadovanou šířku pásma. Jedná se o Voice Bandwidth Calculator, ve kterém je nutné vybrat informaci o použitém kodeku, hlasovém protokolu, velikosti hlasového užitečného zatížení, komprimace hlaviček RTP, přístupové technologii a zabezpečení. Výsledná vypočítaná hodnota je optimální šířka pásma pro hlasovou

komunikaci podle zadaných parametrů. Důležité je uvědomit si, že tato hodnota je určena pouze pro přenos hlasové komunikace a je nutné ji sečíst s požadovanou šířkou pásma pro ostatní datové služby.

## 5.2 Přenosové technologie

### 5.2.1 Virtuální privátní síť

VPN [27],[30] je zkratkou pro virtuální privátní síť - virtual private network. Jedná se o počítačovou síť, která využívá fyzické komponenty existujících privátních (vnitropodnikových) a veřejných sítí (Internet) a existuje tedy pouze virtuálně.

Typickým příkladem uživatele VPN je společnost, která má několik privátní sítí - v hlavním sídle a na několika pobočkách. Každá z těchto privátních sítí má vlastní IP adresaci a je připojena do Internetu. V hlavním sídle je nainstalován VPN server a na každé z poboček pak VPN klient. Tím vzniká na privátní adresaci virtuální privátní síť, díky níž mohou pobočky komunikovat se servery v hlavním sídle popř. jakýkoli počítač na jakékoli pobočce s jakýmkoli počítačem na jiné pobočce.

Nespornou výhodou VPN je šifrování datového přenosu, díky němuž nemůže nikdo neoprávněně odposlechnout ani měnit data během jejich přenosu po veřejném Internetu. Jednotlivé stanice nebo sítě připojované na VPN server jsou navíc ověřeny bezpečnou metodou (sdílený klíč, tzv. pre-shared key, nebo klientský certifikát), což zabraňuje přístupu neoprávných osob.

Do virtuální privátní sítě se nemusí připojovat celá síť, ale může být do ní připojen i samostatný počítač. Díky tomu se např. na služební cestě můžete připojit bezpečně do vnitrofiremní sítě a pracovat s dokumenty uloženými ve firemním serveru, případně se pomocí VPN může připojit vzdáleně administrátor a provádět servisní úkony.

**Základní příklady použití VPN :**

- připojení uživatelů na pobočkách k aplikaci běžící centrálně
- vzájemné propojení všech poboček (možnost přístupu k počítačům, tiskárnám a dalším zařízením na jednotlivých pobočkách pomocí vnitřních IP adres)
- pohodlné připojení do vnitrofiremní sítě z domova, z "terénu" (např. z mobilního připojení) nebo na služební cestě pomocí internetové přípojky v hotelovém pokoji
- management VPN, díky níž se může provádět vzdálená správa (např. konfigurace serverů)

Základní princip VPN spočívá v zapouzdřování IP paketů do paketů s novou hlavičkou. Podle toho na jaké vrstvě síťového modelu se přidá nová hlavička rozdělujeme i typy VPN.

**VPN na spojové vrstvě** - Přenosový síťový systém je použit pro spojení na fyzické a spojové vrstvě, tato síť je funkční analogií konvenční privátní datové síti. Jsou používány infrastruktury Frame Relay a ATM.

**VPN na síťové vrstvě** - Přenášené pakety IP protokolu jsou zabalené do jiných IP paketů, které mají vlastní hlavičku a mohou tak podléhat jiné směrovací politice. Tunelují se celé IP pakety a minimálně se zvětšuje režie přenášených dat. Rozlišujeme 3 základní typy VPN na síťové vrstvě:

- IPSec – Umí privátní tunel zabezpečit jak na úrovni ověření autenticity obou komunikujících uzlů, tak ve smyslu utajení přenášených dat šifrováním.
- PPTP - Využívá pro svou činnost protokol PPP (Point to Point Protocol) a je podstatně méně bezpečný oproti IPSecu (a L2TP). Jeho šifrovací klíče jsou odvozeny přímo z uživatelova hesla. Je tedy náchylný na offline útoky na uživatelovo heslo.
- L2TP - Layer 2 Tunneling Protocol, přenáší PPP skrz sítě, které nejsou PPP tím, že zapouzdřuje PPP datagramy pro přenos transportní sítí a v cílové adrese je datagram rozbalen

**VPN na transportní a aplikační vrstvě** – Využívá se protokolu SSL (Secure Socket Layer) a TLS (Transport Layer Security), které definují způsob šifrování a autentizaci přenášených dat na úrovni vyšší vrstvy v OSI modelu. Aplikace, která data přenáší musí mít implementovánu podporu SSL popř. TSL.

### 5.2.2 Protokol IPSec

IPsec [8],[30] je rozšíření IP protokolu, které poskytuje bezpečnost pro IP protokol a protokoly vyšších vrstev. Nejdříve se vyvinul pro nový standard IPv6 a následně byl zpětně implementován na IPv4. Při realizaci návrhu jej využívám pro vytvoření tunelu mezi hraničními routery jednotlivých sítí čímž je zaručena integrita dat, autentizace a důvěryhodnost komunikace. Ipsec má dvě fáze svého běhu - Main mód a Quick mód. V Main modu pracuje IKE (Internet Key Exchanger), který ověří obě strany šifrovaného spojení a ustanoví prvotní bezpečnou komunikaci. Quick mod se pak již používá pro samotný přenos dat. Existuje ještě jeden mód - Agressive. Je to zjednodušená kombinace main a quick s trochu sníženou bezpečností. Tento mód se používá pro připojení klientů, kteří nemají pevnou IP adresu.

Pro pochopení fungování protokolu IPSec je nutno zavést ještě další pojmy:

- HMAC (Hash Message Authentication Code) – Používá se pro ochranu neporušenosti (integrita) dat. Aby IPSec protokol získal tento HMAC, užívá hashovaní funkce např. MD5 a SHA a vypočte hash na základě tajného klíče a obsahu IP datagramu. Tento HMAC je potom zahrnut do hlavičky IPsec protokolu a příjemce paketu může kontrolovat HMAC, pokud má přístup k tajnému klíči.

- Pro ochranu důvěryhodnosti (vlastní utajení dat) IP paketů se používají standardní symetrické šifrovací algoritmy. Dnes jsou nejběžnější silnější algoritmy jako 3DES, AES a Blowfish.

- Dieffiel-Hellmanovo číslo - Je to velké prvočíslo používané pro výběr tajného symetrického klíče.

**Princip komunikace:**

Main mód za účelem navázání spojení a autentizace sestavuje spojení na UDP 500. Port 500 je cílový i zdrojový. Je navázáno veřejné spojení[12],[14] :

a) Pokud chce strana A navázat spojení, pošle straně B své parametry. (sdílený klíč, hashovací algoritmus, šifrovací metodu, velikost klíče, DH číslo pro generování náhodného velkého, daba platnosti klíče).
b) Strana B odpoví straně A, jaké parametry akceptovala. Parametry putují ještě jako čistý text.
c) Před vysláním třetí zprávy dojde na základě parametrů k vygenerování velkého prvočísla, které poslouží k odvození tajného klíče. Generování jakéhokoliv náhodného čísla je pro počítač problém, a tak se jako generátory náhodnosti berou různé zdroje např. aktuální hodnota teploty procesoru a podobně. Toto číslo se pošle straně B.
d) Strana B také vygeneruje číslo, ze kterého se tajný klíč počítá, a pošle ho straně A.
e) V tuto chvíli obě strany vypočítají tajné klíče a pomocí nich se autentizují.

Tímto se obě strany bezpečně ověřily a vytvořily šifrovací klíče pro přenos zpráv pro Quick mód, který se stará o výpočet vlastního šifrovacího klíče určeného pro přenos dat. Zprávy Quick módu proudí dvěma jednosměrnými tunely. To je dáno asymetrickým šifrováním, data se zašifrují jedním klíčem, ale dešifrovat se musí druhým. U asymetrické šifry se klíč na šifrování nedá použít na dešifrování zprávy.

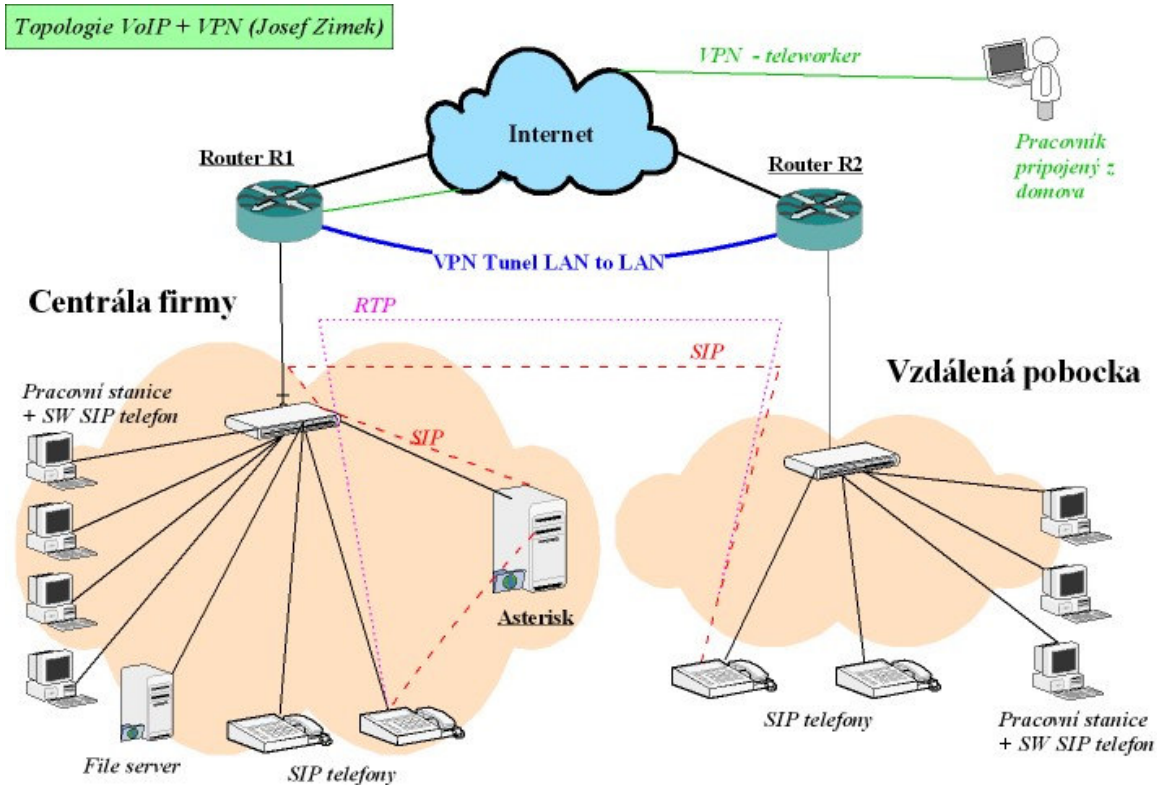
Vlastní přenos dat v Quick módu probíhá pomocí dílčího protokolu ESP - Encapsulated Security Payload [14],[30]. ESP protokol může zajistit jak integritu paketu pomocí HMAC, tak i důvěryhodnost pomocí šifrování. Po zašifrování paketu a výpočtu HMAC se vytvoří ESP hlavička a přidá se do paketu.

Při transportu dat se IPSec může chovat dvěma způsoby. Zašifruje datovou část a hlavičku nechá bez změny, a nebo zašifruje celý paket a přidá novému paketu nové záhlaví. První, transportní, mód vyžaduje, aby pakety mezi zařízeními mohly chodit i bez šifrování. Druhý, tunelovací mód, umožňuje propojit dvě sítě s neveřejnými adresami. Pro své účely používám při realizaci tunelovací mód.

## 5.3 SIP server

Pokud chceme využít výše popsané výhody serveru, budeme potřebovat na některém účet. Je celá řada veřejně dostupných SIP serverů, ale pro firemní řešení, kde je potřeba nezávislost a spolehlivost není tato cesta vhodná. Lepším řešením je zprovoznění vlastního SIP serveru v privátní síti. Existuje celá řada HW i SW řešení v komerční i nekomerční podobě. Pro svůj návrh jsem si vybral open-source variantu Asterisk [6],[32], který v sobě implementuje VoIP pobočkovou ústřednu, SIP Registrar, Location, Proxy a Redirect server.

## 5.4 Návrh topologie




Obr.7 : Návrh topologie

# 6.Vlastní realizace

Problematiku realizace navrhovaného řešení jsem rozdělil do dvou fází. První fáze zahrnuje problémy spojené s vytvořením šifrovaného tunelu mezi dvěma nezávislými sítěmi pomocí dvou směrovačů a druhá fáze obsahuje vytvoření prostředků pro hlasovou komunikaci přes šifrovaný tunel.

## 6.1 Vytvoření směrovačů

Aby mohl jeden počítač najít jiný počítač v síti, musí existovat mechanismus, který počítači definuje jakým způsobem se lze s jiným počítačem spojit. Tento mechanismus nazýváme „směrování“. Tento „směr“ (route) je tvořen dvojicí adres – cílovou adresou (adresa počítače, se kterým se chceme spojit) a adresou směrovací brány. Tato dvojice určuje pravidlo, že chceme- li se dostat na adresu cíle, musíme se vydat směrem přes směrovací bránu. Existují tři typy cílů – individuální počítač, podsíť a implicitní směr. Implicitní směr je použit v případě, pokud nejsou ostatní směry uplatněny. Pro počítače zapojené v lokální síti je jako směrovací brána nastaven jakýkoliv počítač, který má přímé spojení se světem mimo lokální síť. Konfiguruje-li se implicitní směr pro počítač, který je zároveň směrovací bránou, pak implicitní směr bude počítač poskytovatele připojení k Internetu.

Směrovací brány (též směrovače, routery) jsou zařízení, která ve svém jádře mají speciální část kódu, která dle definovaných pravidel rozhoduje jakým způsobem budou příchozí pakety přeposílány nebo zamítnuty. V dnešní době existují směrovače v podobě specializovaných zařízení určených pouze k tomuto účelu. Ve své práci jsem však zvolil metodu použití osobních počítačů. Routery jsou tedy realizovány pomocí starších počítačů, které dnes nesplňují nároky ani na běžnou kancelářskou práci, ale pro moje účely postačí bohatě. Konkrétně se jedná o dvě PC s procesory Intel Celeron 266MHz, 224MB RAM a Intel Celeron 366MHz 128MB RAM. Samotný počítač sám o sobě nedisponuje funkcemi směrovací brány a proto jsem každý z nich vybavil dvěmi síťovými kartami - jedna pro připojení do lokální sítě a druhá k veřejné síti. Poslední nezbytný prvek každého směrovače je operační systém. Některé operační systémy např. zatím nejrozšířenější Microsoft Windows v sobě implementují funkce směrovací brány, ale vzhledem k mému požadavku na nekomerční charakter řešení jsem zvolil open source OS - FreeBSD.

## 6.2 Operační systém FreeBSD

Před mnoha lety potřebovala společnost AT&T pro zajištění svých obchodních aktivit velké množství počítačového softwaru. Nebylo jim však dovoleno soutěžit na poli počítačového průmyslu, a proto poskytli licence na různé druhy vlastního softwaru i s jejich zdrojovými kódy univerzitám a to za velmi výhodné cenové relace. Univerzitní studenti tak získali přístup k těmto technologiím a díky zdrojovým kódům i možnost naučit se, jak fungují. Na oplátku získala AT&T jistý druh reklamy a generaci informatiků, kteří nabrali zkušenosti s jejich technologiemi a všechny strany byly spokojené. Nejznámějším produktem zahrnutým do tohoto licenčního plánu byl právě UNIX.

Ve srovnání s moderními operačními systémy nebyl původní UNIX příliš dobrý. Ale protože tolik studentů mělo přístup k jeho zdrojovým kódům a učitelé potřebovali pro své studenty projekty, byl jejich společným úsilím UNIX vylepšován. Postupně vznikaly užitečné příkazy, byla přidána schopnost ovládat běžící programy, dnes známá jako správa úloh. V průběhu mnoha let byly celé kusy původního operačního systému UNIX vyjímány a nahrazovány. Různé univerzity, které na UNIXu pracovaly, sdílely svá vylepšení a zdokonalení se skupinou Computer System Research Group (CSRG) působící na Kalifornské univerzitě v Berkeley. Ta se stala střediskem všech úprav kódu původního UNIXu. Jejich kód byl distribuován zdarma pro všechny, spolu s platnou licencí AT&T. Výsledná kolekce záplat se stala známá jako Berkeley Software Distribution – BSD UNIX. Vývojový proces pokračoval dlouho podle tohoto schématu avšak v roce 1992 se finanční prostředky CSRG začaly tenčit a to spolu s politickými tahanicemi uvnitř Kalifornské univerzity způsobilo uvolnění kódu pro veřejnost pod licencí, která je dnes známá jako BSD licence a obsahuje tři klauzule, které se dají shrnout takto:

- Netvrďte, že jste tento kód napsali sami
- Neobviňujte nás, když se něco pokazí
- Nepoužívejte naše jméno k propagaci Vašich produktů

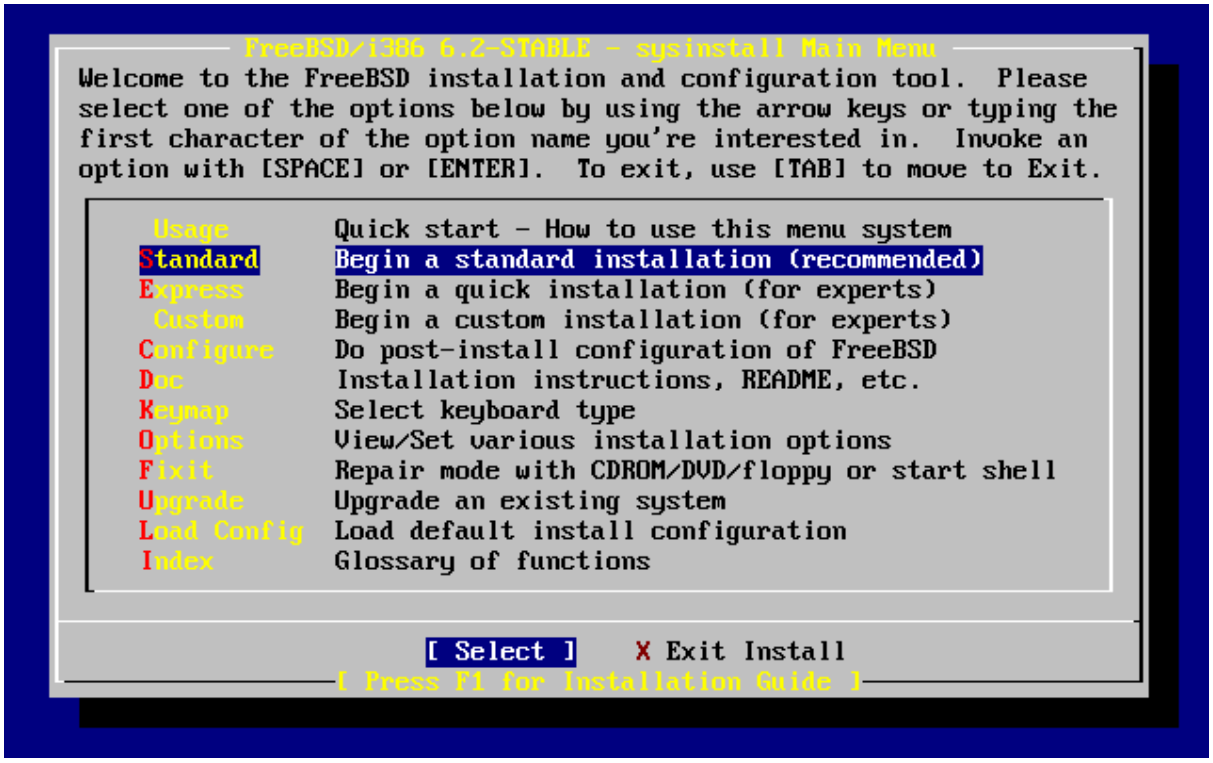
Uvolnění kódu se setkalo s velkou nevolí ze strany Unix System Laboratoriem (USL). Spor spočíval v autorských právech, na která si dělaly nárok obě strany. Mezitím vzali různí lidé uvolněný kód a začali na jeho základě stavět komerční i volně dostupné produkty, mezi které patří i systém 386BSD, který se stal základnou pro operační systém FreeBSD [2],[3],[25].

FreeBSD je vyspělý unixově orientovaný operační systém, který je k dispozici zdarma na Internetu. Jedná se o multiuser multitasking OS, čili umožňuje práci ve víceuživatelském režimu a současné zpracovávání více procesů. Poskytuje moderní funkce operačních systémů, a proto se hodí pro širokou škálu aplikací, od vestavěných systémů až po nejvýkonnější multiprocesorové servery. Je s oblibou využíván jako internetový i intranetový server, protože i pod největší zátěží poskytuje robustní síťové služby a efektivně využívá paměť pro udržení rychlé odezvy pro tisíce současně běžících uživatelských procesů. Mezi jeho další výhody patří přenositelnost, tedy podpora nejrůznějšího hardware od architektury Intel x86(386,486,Pentium I – IV, Celeron), AMD přes 64-bitové architektury až po PowerPC od Motoroly či SPARC od Sun Microsystems. Díky více než 17.000 instalovatelných knihoven a aplikací je FreeBSD vhodné pro využití v prostředích stolních počítačů, serverů, zařízení a vestavěných systémů, to vše při zachování výkonnosti a bezpečnosti.

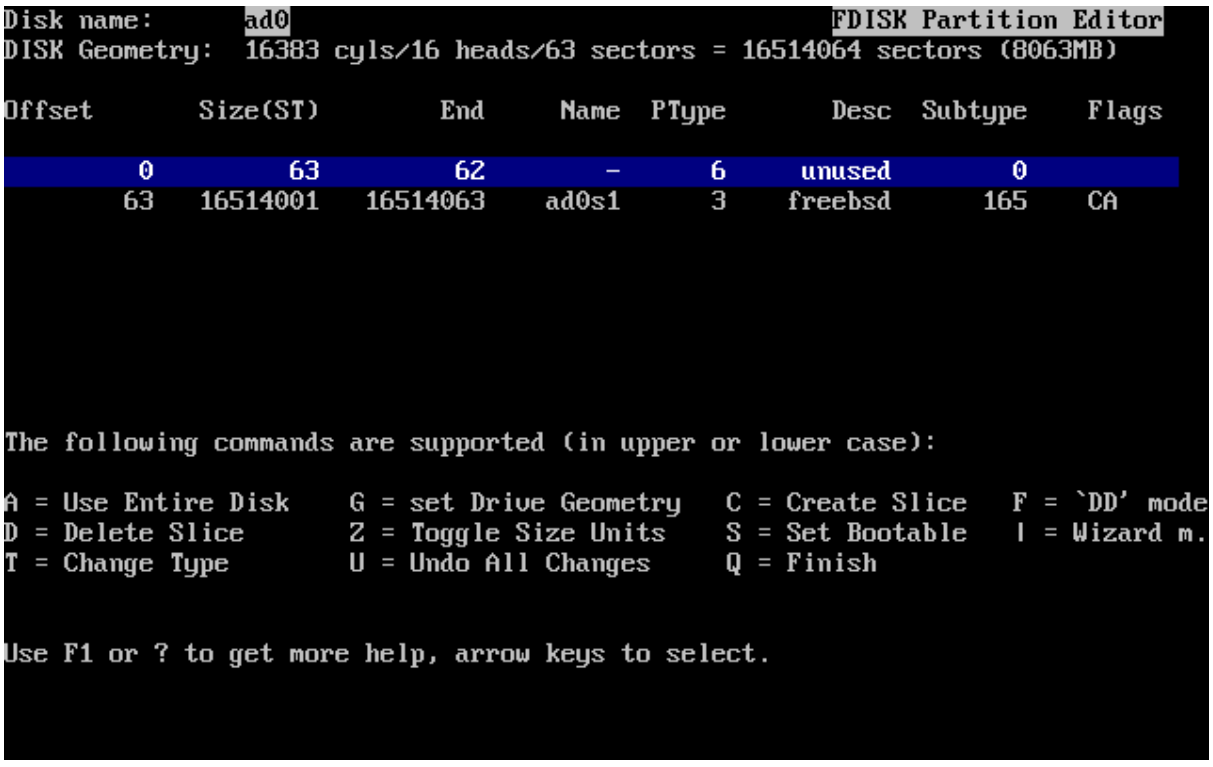
### 6.2.1 Instalace FreeBSD

FreeBSD lze instalovat z mnoha médií, například z CD, DVD, nebo přímo přes síť pomocí FTP nebo NFS. Zvolil jsem možnost instalace z CD (FreeBSD 6.2), které je ke stažení na stránkách projektu. Před vlastní instalací se doporučuje seznámit se s hardware daného počítače a dále se doporučuje v menu BIOSu vypnout volbu „Plug-and-Play OS“ – díky tomu provede BIOS některá nastavení hardware, která pak dále nemusí řešit operační systém. Nutné je povolit v BIOSU možnost bootovat systém nejprve z CD/DVD mechaniky. Po zavedení systému z CD nás uvítá instalační nabídkový systém zvaný **sysinstall**. Působí jako starý program typu MS Dos, ale přes svůj vzhled je rychlý, poměrně intuitivní a hlavně pracuje dobře.

Jako první výzvu sysinstallu obdržíme požadavek na výběr regionu – v mém případě Czech Republic potvrzeno klávesou Enter. Tím se dostaneme do hlavního menu instalace, kde je možno si přečíst dokumentaci popř. nastavit rozložení klávesnice, ale těmito kroky se zde nebudu zabývat jelikož nejsou klíčové pro úspěšné dokončení instalace. Spuštění vlastního procesu instalace(obr. 8) se provede volbou Standard, která spustí program Fdisk (obr.9), který umožňuje rozdělení disku a výběr bootovacího oddílu.

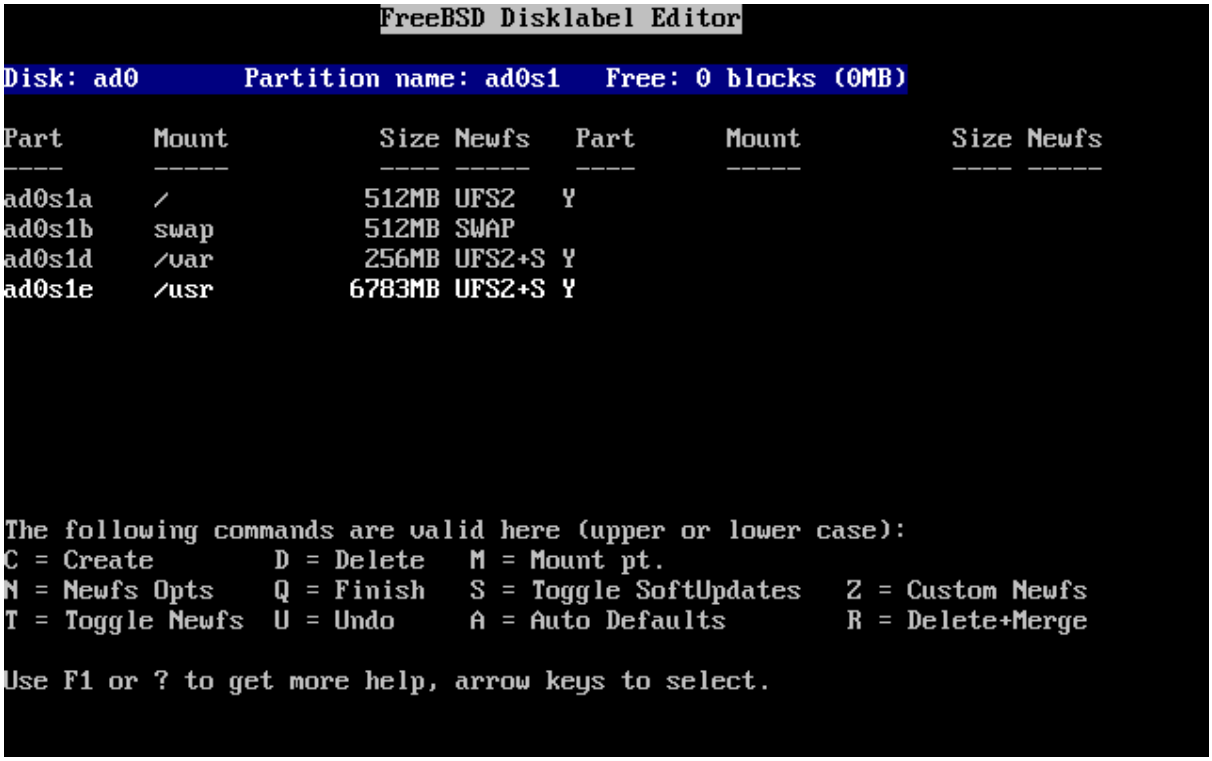



Obr. 8 : Sysinstall – spuštění instalace




Obr. 9 : Rozdělení disku - Fdisk

Rozdělení disku je nutné věnovat pozornost zvláště v případě, zda je systém FreeBSD instalován na stroj kde je již jiný operační systém, který se bude z nějakého důvodu také používat. Značení jednotlivých disků i oddílů se liší od značení rozšířenějších OS Windows či Linux. První disk (IDE) je označen jako ad0 druhý ad1 apod. Jednotlivé oddíly na discích se pak značí s1a (slice), s1b atd. Tedy výsledná podoba je ad0s1a pro první oddíl na prvním disku. Pro své účely nepotřebuji zavádět jiný systém a celý disk je vyhrazen pro FreeBSD, čili klávesou „D“ jsem smazal všechny existující oddíly a klávesou „A“ jsem vybral volbu – použít zbylé místo na disku. Ještě je nutné v tomto kroku vytvořený oddíl označit a stiskem klávesy „S“ ho nastavit jako bootovací. Klávesou „Q“ opustíme Fdisk a jsme požádáni o výběr umístění zavaděče jádra systému (boot loader). Vzhledem k tomu, že nepotřebuji zavádět jiný systém než FreeBSD vyhovující potvrdit volbu Standard a tím se spustí Disklabel Editor (obr.11), kde je nutné vytvořit základní adresářovou strukturu a přiřadit jí místo na disku. To se nejsnáze provede stiskem klávesy „A“ a instalátor provede dle dostupných prostředků optimální výběr. Stiskem klávesy „Q“ ukončíme Disklabel Editor a jsme vyzváni k výběru distribuce. Je zde na výběr např. vývojářská nebo uživatelská distribuce, která dle volby nainstaluje doplňkové aplikace. Vybral jsem volbu „Minimal“, čímž se nainstaluje vše potřebné pro běh systému a nic víc. Potřebný software si později zkompiluji ve funkčním systému.




Obr. 10 : Disklabel Editor

Tím je nejdůležitější část za námi a zbývá vybrat instalační médium, k dispozici jsou např. FTP, NFS, a hlavně CD/DVD, ze které instaluji. V tuto chvíli se z CD nainstalují potřebné komponenty a poté jsme vyzváni ke konfiguraci dílčích částí systému, což provedu až dle potřeby později. Posledním nutným krokem je zvolení hesla pro superuživatele „root“ a můžeme systém restartovat. Je nutné vyjmout CD z mechaniky aby nedošlo k opětovnému naběhnutí instalace. Po restartu a úspěšném přihlášení nás přivítá příkazová řádka (shell) funkčního operačního systému FreeBSD.

V této fázi máme dobrý základ pro vytvoření směrovače, ale je nutné provést ještě několik podstatných úprav. Jelikož FreeBSD používá vlastní souborový systém (UFS2) je důležité se seznámit s jeho hierarchií. Základní adresářová struktura systému vypadá následovně [25] :

| | |
|---|---|
| / | Kořenový adresář souborového systému, který se při spouštění systému připojuje jako první a obsahuje základní systém, který je nezbytný pro spuštění dalších částí systému. |
| / bin / | V tomto adresáři se nachází zásadní uživatelské služby a procesy pro obě prostředí (jednouživatelské a multi-uživatelské). |
| / boot / | Programy a konfigurační soubory, použité při zavádění operačního systému. |
| / dev / | Tato část obsahuje informace vztahující se k zařízením, a ovladačům pro použitý hardware. |
| / etc / | Velice podstatný adresář, který obsahuje systém konfiguračních souborů a skriptů pro používané služby a aplikace. |
| / mnt / | Prázdný adresář běžně používaný správcem systému jako dočasný přípojný bod. |
| / proc / | Jedná se o virtuální souborový systém, který umožňuje ladění systémového nastavení za běhu systému. |
| / root / | Domovský adresář pro superuživatele „root“. |
| / sbin / | Systémové programy a utility pro správu základních prostředí. |
| / tmp / | Dočasné soubory. Obsah / tmp obvykle není zachovány v systému jakmile se restartujte počítač. |
| / usr / | Většina uživatelských nástrojů a aplikací. |
| / var / | Víceúčelový adresář obsahující dočasné a přechodné soubory (např. logovací soubory). |

### 6.2.2 Kompilace vlastního jádra

Proces kompilace jádra (kernel) je v podstatě vytvoření nového jádra operačního systému podle zadaných parametrů. Nabízí se otázka proč vytvářet nové jádro v právě nainstalovaném systému. Je to proto, že standardní jádro, označované jako **GENERIC**, v sobě obsahuje podporu nejrůznějšího hardware, který je pro mé potřeby nevyužitý, a také hlavně proto, že neobsahuje podporu protokolu IPSec, který potřebuji k vytvoření šifrovaného tunelu. Proto je nutné podporu přidat a vytvořit tak vlastní (modifikované, negenerické) jádro.

Dřívější jádra FreeBSD byla tzv. „monolitická“, tedy jádro se chovalo jako jeden velký program, který podporoval fixní seznam hardware a pokud byla snaha změnit chování jádra, bylo nutné ho znovu sestavit – zkompilovat, a poté restartovat systém s novým jádrem. Oproti tomu se dnes rychle přechází k modelu, kdy velká část funkcí jádra je obsažena v modulech, které mohou být dynamicky zaváděny popř. odebírány z jádra dle potřeby. To umožňuje jádru, aby se mohlo přizpůsobit novému hardware, který je k dispozici (např. PCMCIA karty v notebooku) nebo novým funkcím, které mají být do jádra zavedeny bez potřeby sestavovat nový kornel. Tato koncepce je dnes známá jako modulární jádro.

Navzdory tomu je stále nutné provádět některé statické konfigurace jádra. Zpravidla je tomu proto, že ještě nikdo nenapsal dynamicky zaveditelný modul pro danou funkci. Tento scénář nastává i v případě podpory protokolu IPSec a je tedy nutné ji nastavit staticky. Ačkoliv je kompilace jádra časově náročný proces, přináší mému systému řadu výhod. Na rozdíl od generického kernelu, který musí podporovat celou řadu technického vybavení, podporuje vlastní jádro pouze hardware obsažený v konkrétním počítači. To umožňuje rychlejší start systému, jelikož jádro operačního systému bude kontrolovat pouze technické vybavení mého systému, čímž se podstatně zkrátí čas potřebný k zavedení systému. Dále upravené jádro má obvykle menší nároky na využití paměti, což je důležité, protože jádro je jeden z procesů, který musí být přítomen v paměti. Z tohoto důvodu je vlastní jádro vhodné i pro počítače s nižšími hodnotami paměti RAM. Důležitým krokem u kompilace jádra není však jen odebrání nepotřebného hardware, ale naopak i přidání podpory hardware, který v generickém jádře není obsažen. Typickým příkladem tohoto hardware jsou zvukové karty. Tímto je vysvětlen důvod tvorby vlastního jádra a nyní přejdu k vlastnímu procesu kompilace.

Ke kompilaci je potřeba CD, ze kterého jsme instalovali samotný operační systém. Je to proto, že při instalaci jsem vybral tzv. minimální distribuci, čili zdrojové kódy jádra nebyly zkopírovány na pevný disk počítače. To se provede z příkazové řádky příkazem `sysinstall`, a poté se spustí již známý průvodce instalací. V menu průvodce vybereme **Configure – Do post-install configuration** a v další nabídce zvolíme **Distributions** a **src**. V tuto chvíli mezerníkem označíme položky **base, ports** a **sys** a potvrdíme tlačítkem **install**. Zdrojové kódy se zkopírují na pevný disk do adresáře /usr/src/sys/i386/conf/. Nyní zkopírujeme zdrojové kódy jádra do souboru MOJEJADRO, který budeme posléze editovat. Z praktických důvodů se doporučuje zkopírovat kódy jádra mimo adresář /usr/src/, protože může být smazán např. při aktualizaci systému na novou verzi. Nejlepší je tedy zkopírovat kódy jinam např. /root/jadra/ a poté vytvořit symbolický odkaz na soubor v adresáři /usr/src/sys/i386/conf/. To se provede sérií příkazů:

```
# cd /usr/src/sys/i386/conf
# mkdir /root/jadra
# cp GENERIC /root/jadra/MOJEJADRO
# ln –s /root/jadra/MOJEJADRO
```

Nyní je nutné editovat soubor MOJEJADRO. Vzhledem k tomu, že veškerá nastavení provádím z příkazové řádky, je potřeba použít textově orientovaný textový editor. V základní instalaci FreeBSD je obsažen editor **vi**. Jeho ovládání vydá na samotnou kapitolu, ale to je nad rámec této práce. V souboru je nutné přidat podporu protokolu IPSec a odebrat podporu nepotřebného hardware. Soubor je rozdělen do několika částí. V první je možné definovat podporované technologie. Zde je vhodné zakomentovat nepotřebné možnosti např. ATM, IPv6, apod. Komentář se provádí znakem #. Následuje výčet podporovaného vybavení kde opět zakomentujeme nepotřebný hardware např. SCSI disky, RAID řadiče atd.

```
# vi /root/jadra/MOJEJADRO
.
.
.
#options     IPv6
options      IPSEC
options      IPSEC_ESP
```

Pro další optimalizaci výsledného kernelu se doporučuje editovat také soubor **/etc/ make.conf**, kde se stejným způsobem zakomentují ty moduly, které nechceme v novém jádře zavádět např. poštovní server Sendmail apod. Tímto způsobem upravené a uložené soubory MOJEJADRO a make.conf jsou připraveny k sestavení zdrojových kódů pro nové jádro. To se provede v adresáři /usr/src/ následujícím příkazem:

```
# Make buildkernel KERNCONF = MOJEJADRO
```

Proces sestavení nových zdrojových kódů je časově náročný a trvá několik hodin. Po úspěšném sestavení už stačí dát pokyn k instalaci nového jádra z vytvořených kódů:

```
# Make installkernel KERNCONF = MOJEJADRO
```

Tím se nové jádro nainstaluje a po dokončení stačí zadat příkaz `reboot` a systém nabootuje již podstatně rychleji a s podporou protokolu IPSec. Je důležité podotknout, že tento proces kompilace je nutné provést na obou počítačích, které mají sloužit jako router. V tuto chvíli již máme platformu pro vytvoření směrovače a zbývá jen vytvořit a nakonfigurovat šifrovaný tunel.

### 6.2.3 Instalace aplikací

Jelikož bude potřeba doinstalovat některé další aplikace, je nutné se seznámit se způsoby instalace ve FreeBSD. Existují dva záladní systémy instalace tzv. systém portů (ports) a systém balíčků (packages). Porty jsou instrukce pro kompilaci softwaru a balíčky jsou předkompilované porty. Balíčky se tedy instalují mnohem rychleji, naproti tomu porty se instalují pomalu, ale akceptují specifické parametry daného systému, zejména změny uvedené v /etc/make.conf. Vzhledem k tomu, že jsme při kopírování zdrojových portů pro kompilaci jádra vybrali i možnost ports, máme nyní v adresáři /usr/ports/ tzv. strom portů, který obsahuje kategorie softwaru. Každá z kategorií pak obsahuje další úroveň adresářů s konkrétním software. Při aktivaci portu systém automaticky stáhne příslušné zdrojové kódy z ověřeného internetového serveru. Pak zkontroluje integritu stažených souborů, rozbalí kód do pracovního adresáře, aplikuje záplaty, sestaví program, který dále nainstaluje a zaznamená instalaci do adresáře /var/db/pkg. To vše se provede jediným příkazem spuštěným v adresáři příslušného portu :

# make install

V případě instalace pomocí balíčků se ušetří spoustu času, jelikož neprobíhá kompilace zdrojových kódů aplikace. Možností instalace je v tomto případě více. Lze instalovat balíčky obsažené na instalačním cd operačního systému nebo vzdáleně z ftp serveru. Lepší volba je instalace z oficiálního ftp serveru FreeBSD, kde jsou umístěny oficiální balíčky. Tento server se nachází na území USA, ale jeho obsah je zrcadlen na servery po celém světě. Pokud tedy známe název programu, který chceme nainstalovat pomocí balíčkovacího systému použijeme příkaz:

```
# pkg_add –r  NAZEV_APLIKACE
```

K odinstalování balíku ze systému slouží příkaz :

```
# pkg_delete  NAZEV_APLIKACE
```

Z výše uvedeného popisu instalace vyplývá, že pro úspěšnou instalaci je nutné být připojen k Internetu. K tomu je potřeba disponovat konektivitou a příslušným nastavením síťové karty. Zpravidla je vyžadována IP adresa stanice, síťová maska, adresa výchozí brány a adresa DNS serveru. Toto nastavení se provádí následujícími příkazy:

```
# ifconfig rl0 inet IP_ADRESA_STANICE netmask SITOVA_MASKA
# route add default  IP_ADRESA_VYCHOZI_BRANY
# cat nameserver IP_ADRESA_DNS_SERVERU > /etc/resolv.conf
```

### 6.2.4 Vytvoření šifrovaného tunelu

V této sekci bude provedeno vytvoření šifrovaného IPSec tunelu pomocí routerů postavených na platformě FreeBSD. Uživatelům bude pomocí tohoto tunelu umožněno komunikovat bezpečným způsobem. Použité adresní schéma IPv4 je uvedeno na obr.12.


Obr. 11 : IPSec tunel

Vlastní vytvoření šifrovaného tunelu je rozděleno na dvě části :

- vytvoření virtuální linky mezi dvěma sítěmi
- aplikování bezpečnostní politiky pro zajištění transparentně šifrované komunikace

Následující konfiguraci je nutné provést na obou strojích, které mají sloužit jako směrovače, avšak pro každý router musí být zadány příslušné IP adresy. Uvedené příkazy budou uvažovány pouze pro router R1. Pro konfiguraci tunelu musí být IP adresy na routeru R2 zadány v opačném pořadí. Vytvoření tunelu se realizuje pomocí generického síťového rozhraní **gif0.** Toto rozhraní je nutné uvést jako parametr (device gif) při kompilaci jádra starších verzí FreeBSD. Ve verzi 6.2, kterou používám je tento parametr zahrnut automaticky. Nejprve je nutné nastavit základní síťové parametry:

```
# ifconfig rl0 inet 10.0.0.1 netmask 255.0.0.0
# route add default  10.0.0.2
# ifconfig rl1 inet 192.168.1.1 netmask 255.255.255.0
```

Dále je nutné vytvořit rozhraní gif0 a nakonfigurovat tunel:

```
# ifconfig gif0 create
# ifconfig gif0 tunnel 10.0.0.1 10.0.0.2
# ifconfig gif0 inet 192.168.1.1 192.168.2.1 netmask 255.255.255.0
```

Na druhém stroji tedy analogicky, pouze s převráceným pořadím IP adres :

```
R2# ifconfig gif0 create
R2# ifconfig gif0 tunnel 10.0.0.2 10.0.0.1
R2# ifconfig gif0 inet 192.168.2.1 192.168.1.1 netmask 255.255.255.0
```

V tuto chvíli je rozhraní gif0 vytvořeno a jeho nastavení ověříme příkazem :

```
# ifconfig gif0
gif0: flags=8051<UP,POINTOPOINT,RUNNING,MULTICAST> mtu 1280
        tunnel inet 10.0.0.1--> 10.0.0.2
        inet 192.168.1.1 --> 192.168.2.1 netmask 255.255.255.0
```

Z výstupu je patrné, že byl vytvořen tunel mezi adresami 10.0.0.1 a 10.0.0.2 a je povolena komunikace mezi adresami 192.168.1.1 a 192.168.2.1. Tento záznam se uloží taktéž do routovací tabulky směrovače :

```
# netstat -rn
Routing tables
Internet:
Destination      Gateway       Flags    Refs    Use    Netif  Expire
192.168.2.1      192.168.1.1   UH       0       0      gif0
```

Nyní jsou směrovače schopny komunikovat jeden s druhým přes veřejná rozhraní, ale na úrovni rozhraní vnitřních sítí ne. Tento problém se vyřeší přidáním statického záznamu do směrovací tabulky :

```
# route add 192.168.2.0 192.168.2.1 netmask 255.255.255.0
```

Ověření schopnosti komunikovat skrze vnitřní rozhraní routerů lze provést nástrojem **tcpdump** Nejprve se přihlásíme k novému terminálu (Alt+F2) a následujícím příkazem spustíme zachytávání komunikace s adresou 192.168.2.1 :

```
# tcpdump dst host 192.168.2.1
```

Poté se přepneme zpět do původní konzole (Alt+F1) a provedeme příkaz :

```
# ping 192.168.2.1
```

V terminálu zachytávajícím komunikaci lze zkontrolovat úspěšnost spojení :

```
18:21:24.018080 192.168.1.1 > 192.168.2.1: icmp: echo request
18:21:24.018109 192.168.1.1 > 192.168.2.1: icmp: echo reply
18:21:25.018814 192.168.1.1 > 192.168.2.1: icmp: echo request
18:21:25.018847 192.168.1.1 > 192.168.2.1: icmp: echo reply
18:21:26.028896 192.168.1.1 > 192.168.2.1: icmp: echo request
18:21:26.029112 192.168.1.1 > 192.168.2.1: icmp: echo reply
```

V této fázi jsou vytvořeny dvě třetiny VPN mezi dvěma sítěmi – virtuální síť. Z výstupu tcpdumpu je patrné, že ICMP pakety příkazu ping procházejí sítí v otevřené podobě. Aby bylo dosaženo plnohodnotné sítě VPN, je nutné komunikační tunel zabezpečit.

Proces zabezpečení komunikace je rozdělen do dvou oblastí. V prvním kroku se musí definovat mechanismus, pomocí kterého si komunikující strany sdělí jaký šifrovací algoritmus použijí (asociace). Ve druhém kroku je nutné určit, která komunikace má být šifrována. Asociace zahrnuje volbu šifrovacího algoritmu a šifrovacího klíče a lze ji nastavit staticky manuálně. Lze, ale také využít některého z démonů (aplikace poskytující službu systému), které používají Internet Key Exchange protokol (IKE), a také automatizují proces asociace. Použití této varianty se mi jeví jako efektivnější, jelikož šifrovací algoritmy a klíče se průběžně obměňují a zmenšuje se tak riziko odposlechu, protože v případě získání klíče útočníkem se již bude používat jiný.

Existuje řada démonů poskytujících tuto službu, jedním z nich je např. **racoon** [21],[25], kterého jsem použil. V kolekci portů se nachází v **/usr/ports/security/ipsec-tools/racoon,** odkud jej již známým způsobem (make install) nainstalujeme. Konfigurační soubory se pak nachází v /usr/local/etc/racoon/ a z nich nás nejvíce zajímá soubor **/usr/local/etc/racoon/psk.txt,** ve kterém je definován tajný klíč a **/usr/local/etc/racoon/racoon.conf** kde okomentujeme parametry, které chceme použít (3DES, MD5 atd.). Tento klíč však není určen k šifrování přenášených dat přes VPN tunel, pouze zajišťuje určitý stupeň důvěryhodnosti komunikujících stran, které se na jeho základě autentizují. Soubor psk.txt obsahuje informaci o adrese zařízení, se kterým má racoon dohodnout šifrovací parametry, a také tajnou posloupnost znaků – tajný klíč (pre-shared key), který je totožný pro obě strany.

Pro router R1 a R2 bude tedy soubor psk.txt vypadat následovně :

```
R1 : 10.0.0.2 UTAJOVANA_POSLOUPNOST_ZNAKU
R2 : 10.0.0.1 UTAJOVANA_POSLOUPNOST_ZNAKU
```

Z bezpečnostních důvodů je nutné před spuštěním démona racoon změnit přístupová práva k souboru (čtení a zápis pro superuživatele) psk.txt příkazem :

```
# cd /usr/local/etc/racoon/
# chmod 600 psk.txt
```

Nyní můžeme na obou routerech spustit démon racoon s parametrem cesty ke konfiguračnímu souboru :

```
# /usr/local/sbin/racoon -f /usr/local/etc/racoon/racoon.conf
pfkey UPDATE succeeded: ESP/Tunnel 10.0.0.2->10.0.0.1
pk_recvupdate(): IPSec-SA established: ESP/Tunnel 10.0.0.2->10.0.0.1
get pfkey ADD message IPsec-SA established: ESP/Tunnel 10.0.0.1->10.0.0.2

IKE Log: Phase 1 (aggressive) completion. 3DES/MD5/Pre shared secrets Negotiation
scheme: IKE methods: Combined ESP: 3DES + MD5 + PFS (phase 2 completion)
```

Nyní je zaručena šifrovaná komunikace a zbývá definovat, aby byla šifrována pouze komunikace, která je realizována přes vytvořený tunel. Využije se při tom vlastnosti protokolu IPSec, který zapouzdří data do nového paketu s novým záhlavím. Díky tomu můžeme definovat, aby byl šifrován pouze ten paket, který putuje od 10.0.0.1 a je určen pro 10.0.0.2 **a zároveň** zapouzdřuje jiný paket. Samozřejmě je nutné definovat toto pravidlo i pro opačný směr komunikace - šifrovat pouze ten paket, který putuje od 10.0.0.2 do 10.0.0.1 **a zároveň** zapouzdřuje jiný paket. Stejná pravidla platí i pro dešifrování paketů. To se provede pomocí nástroje **setkey** [25], který definuje chování protokolu IPSec. Konfigurace pomocí setkey může být provedena jako příkaz ze standardního vstupu (klávesnice) nebo může být načtena ze souboru, kde je uveden výčet pravidel pro chování protokolu IPSec. Pro načtení konfigurace ze souboru se použije přepínač **–f** a jako parametr se zadá cesta k souboru. Vytvoříme tedy soubor /etc/ipsec.conf, definujeme v něm pravidla a poté je načteme nástrojem setkey :

```
# vi /etc/ipsec.conf

spdadd 10.0.0.1/8 10.0.0.2/8 ipencap -P out ipsec esp/tunnel/10.0.0.1-
10.0.0.2/require;

spdadd 10.0.0.2/8 10.0.0.1/8 ipencap -P in ipsec esp/tunnel/10.0.0.2-10.0.0.1/require;
```

První řádek definuje pravidla šifrování pro odchozí pakety a druhý řádek pro příchozí pakety. Je nutné provést stejnou konfiguraci na routeru R2, ale se přehozenými záznamy o IP adresách. Nyní probíhá komunikace skrze vytvořený tunel šifrovaně. Ověřit to lze opět nástrojem tcpdump v jednom terminálu a ping v druhém :

```
# tcpdump dst host 192.168.2.1
# ping 192.168.2.1
```

Z výstupu je zřejmé, že nástroj tcpdump není schopen definovat typ komunikace, protože již probíhá šifrovaně (protokol ESP a AH) :

```
14:45:39.373005 192.168.1.1 > 192.168.1.2: AH(spi=0x00000200,seq=0x1):
ESP(spi=0x00000201,seq=0x1) (DF)
14:45:39.448636 192.168.1.2 > 192.168.1.1: AH(spi=0x00000300,seq=0x1):
ESP(spi=0x00000301,seq=0x1)
14:45:40.542430 192.168.1.1 > 192.168.1.2: AH(spi=0x00000200,seq=0x2):
ESP(spi=0x00000201,seq=0x2) (DF)
14:45:40.569414 192.168.1.2 > 192.168.1.1: AH(spi=0x00000300,seq=0x2):
ESP(spi=0x00000301,seq=0x2)
```

Je velmi pravděpodobné, že v reálném provozu bude potřeba definovat pravidla firewallu pro povolení komunikace přes VPN tunel. IPFirewall (IPFW) [25], dodávaný s FreeBSD je systém paketového filtrování a účtování, který je volitelně součástí jádra. Ovládá se pomocí uživatelského příkazu ipfw. IPFW se skládá ze dvou částí, kde vedle filtrovacího mechanismu existuje i monitorovací systém, který umožňuje získávat přehled o provozních podmínkách routeru. IPFW lze celkem jednoduše použít hned po instalaci systému, protože je již zkompilován jako modul a lze jej dynamicky zavádět :

```
# kldload ipfw
```

Nastavený je implicitně na „zakázat vše“. Konfigurace se standardně načítá ze souboru /etc/rc.firewall. Pro konfiguraci firewallu se používá vedle konfiguračního souboru i klientský program ipfw. Jeho fungování lze shrnout do čtyř oblastí: přidávání (add) /mazání (delete) pravidel, vypisování pravidel (show), jejich skupin  a čítačů paketů (list), smazání všech pravidel

(flush) a resetování čítačů (clear). Tímto způsobem se definují pravidla i pro mnou vytvořenou síť. Nejprve je nutné povolit veškerou komunikaci přes šifrovaný tunel, která je realizována pseudorozhraním gif0 :

```
# ipfw add 1 allow ip from any to any via gif0
```

Dále je nutné povolit přenos paketů, které jsou určeny k dohodnutí šifrovacích pravidel před vlastním sestavením šifrovaného tunelu. Jsou to pakety typu UDP - ISAKMP (Internet Security Association Key Management Protocol) :

```
# ipfw add 1 allow udp from 10.0.0.1 to 10.0.0.2 isakmp
# ipfw add 1 allow udp from 10.0.0.2 to 10.0.0.1 isakmp
```

Nyní zbývá už jen povolit pakety přenášející šifrovaná data zapouzdřená v nových paketech :

```
# ipfw add 1 allow esp from 10.0.0.1 to 10.0.0.2
# ipfw add 1 allow esp from 10.0.0.2 to 10.0.0.1
# ipfw add 1 allow ipencap from 10.0.0.1 to 10.0.0.2
# ipfw add 1 allow ipencap from 10.0.0.2  to 10.0.0.1
```

Z uvedených příkazů vyplývá, že pravidla jsou symetrická, čili je lze zadat na obou routerech ve stejné podobě. Tímto je vytvořen šifrovaný tunel mezi dvěmi nezávislými sítěmi. Aby se po restartu systému nemuselo vše znovu nastavovat, je nutné definovat automatické spouštění firewallu a nástroje racoon. Pro firewall se jednoduše přidá záznam do souboru **/etc/rc.conf**, který shromažďuje globální nastavení systému :

```
# cat firewall_enable=”yes” >> /etc/rc.conf
```

Automatické spouštění utility racoon se realizuje přidáním záznamu do souboru **/etc/rc.local,** který má na starost zavádění skriptů při startu systému :

```
# vi /etc/rc.local

### !/bin/sh
```

```
### racoon key exchange server
if [ -x /usr/local/sbin/racoon ]; then
  echo -n "racoon "
  /usr/local/sbin/racoon -f /usr/local/etc/racoon/racoon.conf
fi
```

## 6.3 Zajištění hlasové komunikace

Tato část se zabývá vytvořením prostředků pro hlasovou komunikaci mezi dvěmi sítěmi s využitím technologie Voice over Internet Protokol. Při realizaci je využit signalizační protokol SIP a metoda klient-server. Jednotliví klienti se registrují u SIP serveru a po úspěšné registraci mají možnost využívat jeho služeb. Jako platforma pro SIP server je opět použit počítač s operačním systémem FreeBSD (6.2 Stable). Jeho hardwarová konfigurace je následující : CPU – 650MHz, 256MB RAM, a nezbytná je síťová karta. Jelikož operační systém FreeBSD nedisponuje nativní podporou protokolu SIP jako tomu je např. v případě routeru a protokolu IPSec, je nutné využít software třetích stran. Mezi špičku v oblasti integrovaného telekomunikačního softwaru se považuje open source řešení Asterisk. Jedná se o flexibilní rozšiřitelnou pobočkovou ústřednu, která je určená platformám založeným na UNIXu. Asterisk je nejčastěji nasazován v těchto aplikacích [6] :

- VoIP gateway (MGCP, SIP, IAX, H.323)
- Pobočková ústředna (PBX)
- Voicemail služby s adresářem
- Interaktivní hlasový průvodce (IVR) server
- Softwarová ústředna (Softswitch)
- Konferenční server
- Packet voice server

Asterisk je navržen tak, aby povoloval použití nových rozhraní a umožňoval snadno přidávat nové technologie. Jeho cílem je podpora veškerých možných typů současných i budoucích telefonních technologií. Obecně jsou rozhraní rozdělená do tří základních skupin:

- Zaptel hardware (propojení s klasickou telefonní sítí – vyžaduje použití speciálního hardware, TDM)
- Non-Zaptel hardware (alternativní řešení bez technologie TDM)
- Packet voice (komunikace přes paketové sítě)

Při realizaci jsem využil pouze rozhraní typu packet voice, jedná se tedy o „čistokrevné“ VoIP řešení.

### 6.3.1 Instalace a konfigurace PbX Asterisk

Jelikož port Asterisku obsahuje ve svém souboru Makefile (instrukce pro kompilaci) určitá rozšíření, nelze Asterisk nainstalovat klasicky, dokud není nainstalován program gmake, který je o tyto rozšíření obohacen. Poté již lze klasickým způsobem nainstalovat Asterisk z kolekce portů v adresáři /usr/ports/net/asterisk (nutné je funkční připojení k Internetu viz. 6.2.3):

```
# pkg_add –rv gmake
# cd /usr/ports/net/asterisk
# make install
```

Po úspěšné instalaci se konfigurační soubory nachází v **/usr/local/etc/asterisk.** Důležité jsou především soubory **/usr/local/etc/asterisk/extensions.conf** a **/usr/local/etc/asterisk/sip.conf.** Nyní je nutné provést základní konfiguraci síťových služeb a povolení spouštění asterisku při startu systému :

```
# cat ifconfig_rl0="inet 192.168.1.2 netmask 255.255.255.0" >> /etc/rc.conf
# cat defaultrouter="10.0.0.1" >> /etc/rc.conf
# cat asterisk_enable=”YES” >> /etc/rc.conf
```

Konfigurace parametrů klientů [6] komunikujících s Asteriskem se provádí v souboru sip.conf. Soubor je rozdělen na dvě části, v první se nastavují globální parametry shodné pro všechny klienty a za touto částí se definuje nastavení pro konkrétní klienty. Jedná se zejména o tyto vlastnosti :

[general] ;globální část – komentář se provádí znakem „;“

- disallow =all - zakázání všech kodeků
- allow=ulaw - povolení pouze vybraných kodeků
- port=5060 - UDP Port (pro SIP 5060)
- insecure=port,invite - při autentizaci se nebere ohled na port, ze kterého přichází požadavek a není potřeba počáteční výzva INVITE
- nat=yes - používá se při použití překladu adres

[100] ;klientská část – založení účtu 100

- type=peer - funguje jako signalizační server (user - funguje jako klient, friend - funguje oběma způsoby)
- secret=TAJNE_HESLO - heslo použité při autentizaci klienta na serveru
- qualify=yes - funkce pomocí níž definujeme jestli Asterisk bude zkoušet, zda-li je účet aktivní (online)
- host=dynamic - definuje IP adresu klienta; volba dynamic určuje proměnnou adresaci
- context=NAZEV_KONTEXTU - definice domény, do které je účet zařazen
- mailbox=100@default - nastavení čísla pro hlasovou schránku
- userid=jméno příjmení <100> - popis účtu např. iniciály uživatele

Z hlediska usnadnění konfigurace je vhodné pojmenovat jednotlivé účty přímo číslem jejich klapky. Tím je vyřešena konfigurace účtů klientů a nyní je potřeba definovat pravidla, podle kterých se bude řídit přepojování vlastních hovorů. To se provádí v konfiguračním souboru extensions.conf, který je srdcem celého systému. Zde definujeme chování všech spojení v ústředně tzv. číslovací plán (dialplan), který je realizován posloupností funkce exten, která se provádí v pořadí podle zvolené priority :

exten => číslo,priorita,příkaz(parametry)

Číslem je myšlena volaná klapka, priorita (1=nejvyšší) určuje posloupnost vykonávání následného příkazu s příslušnými parametry. Obsah souboru sip.conf je poměrně dlouhý a je umístěn v tištěné podobě v přílohách na konci této práce. Pro své účely jsem v něm vytvořil dva kontexty – jeden pro hlavní sídlo firmy (headquarters – reprezentováno LAN za R1; klapky 100-103) a druhý pro vzdálenou pobočku firmy (branch – reprezentováno LAN za R2; klapky 200-203). Je tedy patrné, že komunikace mezi kontexty bude probíhat přes dříve vytvořený šifrovaný

tunel. Pravidla pro číslovací plán jsou definována pro každý kontext zvlášť v souboru extensions.conf :

```
[headquarters]
exten => _1XX,1,Dial(SIP/${EXTEN},20)
exten => _1XX,2,VoiceMail(${EXTEN}@default)
exten => _2XX,1,Goto(branch,${EXTEN:1},20)
exten => _2XX,2,VoiceMail(${EXTEN}@default)
exten => 9999,1,VoiceMailMain(${CALLERID(num)}@default)

[branch]
exten => _2XX,1,Dial(SIP/${EXTEN},20)
exten => _2XX,2,VoiceMail(${EXTEN}@default)
exten => _1XX,1,Goto(headquarters,${EXTEN:1},20)
exten => _1XX,1,VoiceMail(${EXTEN}@default)
exten => 9999,1,VoiceMailMain(${CALLERID(num)}@default)
```

Princip je následující : pokud v rámci kontextu headquarters vytočí některý z klientů třímístnou klapku, začínající číslem 1, provede se příkaz Dial – volání na zadané číslo prostřednictvím protokolu SIP po dobu 20 sekund. Jestliže do uplynutí této doby volaný účastník nepřijme hovor, provede se následující příkaz s vyšší hodnotou priority. To je v tomto případě příkaz VoiceMail – zanechání hlasové zprávy ve schránce volaného.

Pokud by volaná třímístná klapka začínala číslem 2, provedl by se příkaz GoTo s parametrem druhého kontextu – tedy volání zadaného čísla přes kontext branch. V případě nečinnosti po dobu 20 sekund se hovor opět přesměruje do hlasové schránky volaného. Volaný si pak může zprávy vyzvednout ve své schránce na klapce 9999. Tento princip je obdobný i pro volání v rámci pobočky.

Funkce hlasové schránky se definuje v souboru **voicemail.conf,** kde se definují záznamy o jednotlivých účtech a jejich autentizačních údajích pro výběr hlasových zpráv. Soubor má následující strukturu:

```
číslo_klapky=>heslo, jméno, email, pager, speciální_parametr(např. časová zóna)
```

Tímto je Asterisk nainstalován a nakonfigurován a lze jej spustit prostřednictvím následujícího příkazu, který spustí Asterisk v tzv. konzolovém módu :

```
# asterisk –vvvc
Asterisk Ready.
*CLI>
```

V tomto módu můžeme kontrolovat probíhající hovory a ladit systém podle konkrétních výstupů. Pro načtení aktuálního (upraveného) číslovacího plánu slouží příkaz dialplan reload. Příkazem help získáme nápovědu o dalších možných příkazech.

Nyní už jen zbývá na klientských stanicích nainstalovat a nakonfigurovat hardwarové IP telefony nebo potřebnou klientskou VoIP aplikaci např. Linphone, YATE, SippySkype, Ekiga apod. Způsob jejich nastavení se liší dle výrobce, ale principielně je postup stejný – v konfiguračním menu je nutné zvolit IP adresu SIP serveru, uživatelské jméno a příslušné heslo a po úspěšné registraci na SIP serveru lze realizovat hovory dle pravidel v číslovacím plánu.

Tímto jsou splněny požadavky pro umožnění šifrované komunikace mezi dvěmi nezávislými sítěmi. Toto řešení má však jeden nedostatek, který vyplývá z topologie sítě (obr. 7). Totiž, pokud chce volat klient v rámci sítě pobočky, musí se registrovat u SIP serveru, který se nachází v sídle firmy. Veškerá signalizace se tedy přenáší šifrovaným tunelem, ale hovořící se nachází ve stejné síti. To snižuje efektivitu sítě, jelikož se využívá větší přenosová kapacita. Tento fakt je dán mými omezenými prostředky při realizaci sítě. V praxi se tento problém vyřeší tak, že se použije pro každou síť vlastní SIP server a mezi nimi se nakonfiguruje speciální spoj tzv. trunk linka. Trunk se konfiguruje v souboru sip.conf :

```
[general]
register => trunksip:heslo@IP_adresa_dalšího_sip_serveru
```

Pak se musí definovat účet trunksip způsobem uvedeným výše (typ, heslo, host atd.). Takto upravené konfigurační soubory na obou SIP serverech jsou základem pro úspěšnou komunikaci mezi nimi.

## 6.4 Připojení vzdáleného pracovníka

V dnešní době je stále častější scénář kdy firma potřebuje komunikovat nejen s pobočkou, ale vyžaduje aby ke sdíleným prostředkům firmy mohli přistupovat pracovníci připojení z domova nebo na cestách – souhrnně označovaní jako **teleworkers** nebo **road warriors**. To přináší řadu problémů, jelikož pracovník na cestách se připojuje pokaždé s jinou IP adresou, není možné vytvořit statický tunel, jako tomu bylo v případě linky mezi směrovači. Je tedy nutné zvolit jiný způsob autentizace pro přístup do lokální sítě. Řešení nabízí použití certifikátů. Využije se opět nástroje racoon a pro generování certifikátů **OpenSSL** [11],[23]. SSL je protokol zabezpečující data na přechodu mezi aplikační a transportní vrstvou. Umožňuje zajistit šifrování přenášených dat a autentizaci serveru pomocí digitálních certifikátů. SSL používá klasickou kryptografii s párem soukromý + veřejný klíč. Princip sestavení spojení (handshake) probíhá následovně [14] :

1. Klient pošle serveru požadavek na SSL spojení, spolu s různými doplňujícími informacemi (verze SSL, nastavení šifrování atd.).
2. Server pošle klientovi odpověď na jeho požadavek, která obsahuje stejný typ informací a hlavně certifikát serveru.
3. Podle přijatého certifikátu si klient ověří důvěryhodnost serveru. Certifikát také obsahuje veřejný klíč serveru.
4. Na základě dosud obdržených informací vygeneruje klient základ šifrovacího klíče, kterým se bude šifrovat následná komunikace. Ten zašifruje veřejným klíčem serveru a pošle mu ho.
5. Server použije svůj soukromý klíč k rozšifrování základu šifrovacího klíče. Z tohoto základu vygenerují jak server, tak klient hlavní šifrovací klíč.
6. Klient a server si navzájem potvrdí, že od teď bude jejich komunikace šifrovaná tímto klíčem. Fáze handshake tímto končí.
7. Je ustaveno zabezpečené spojení šifrované vygenerovaným šifrovacím klíčem.

V mém případě se sdílené prostředky nachází v síti za routerem R1, je tedy nutné jej příslušně nakonfigurovat. Do již známého souboru /usr/local/etc/racoon.conf je potřeba přidat definice a určující parametry, které jsou použity při navazování spojení. Soubor je rozdělen na části dle konfigurovaných fází spojení. Následující paramerty jsou nutné pro počáteční fázi – tedy cesta k certifikátu a tajnému klíči, jejich typy, šifrovací algoritmy, které se mají použít :

```
# vi /usr/local/etc/racoon/racoon.conf
path certificate "/usr/local/etc/racoon";
remote anonymous {
    exchange_mode aggressive,main;
    certificate_type x509 "cert.pem" "key.pem";
    generate_policy on;
    dpd_delay 20;
    ike_frag on;
    proposal {
            encryption_algorithm aes;
            hash_algorithm md5;
            authentication_method hybrid_rsa_server;
            dh_group 2;
    }
}
```

Další část definuje IP adresy, které se mají přiřadit úspěšně autentizovanému klientovi :

```
mode_cfg {
    network 192.168.1.100;
    pool_size 10;
    netmask 255.255.255.0;
}
```

Nakonec je nutné určit parametry pro šifrování vlastních přenášených dat :

```
sainfo anonymous {
    pfs_group 2;
    lifetime time 1 hour;
    encryption_algorithm aes;
    authentication_algorithm hmac_md5;

}
```

Dalším krokem je vygenerování certifikátů. K tomu slouží nástroj OpenSSL, který se nainstaluje z kolekce portů :

```
# cd /usr/ports/security/openssl
# make install
```

Vlastní proces generování kořenového certifikátu a tajného klíče se provede následujícím příkazem, který určuje požadavek na vytvoření nového certifikátu ve formátu x509, názvy výstupních souborů a platnost certifikátu 365 dní :

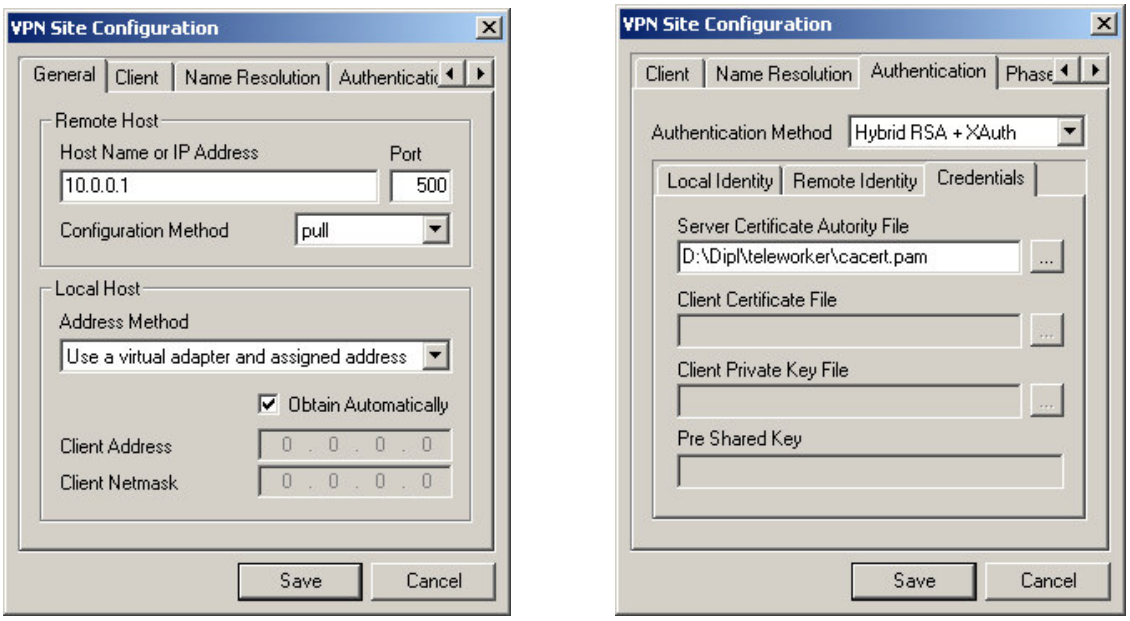
```
# openssl req -new -x509 –out cacert.pem -keyout cakey.pem -days 365
```

Další dvojicí příkazů se vytvoří a podepíše certifikát **cert.pem**, kterým se bude prokazovat R1 a jeko tajný klíč **key.pem** :

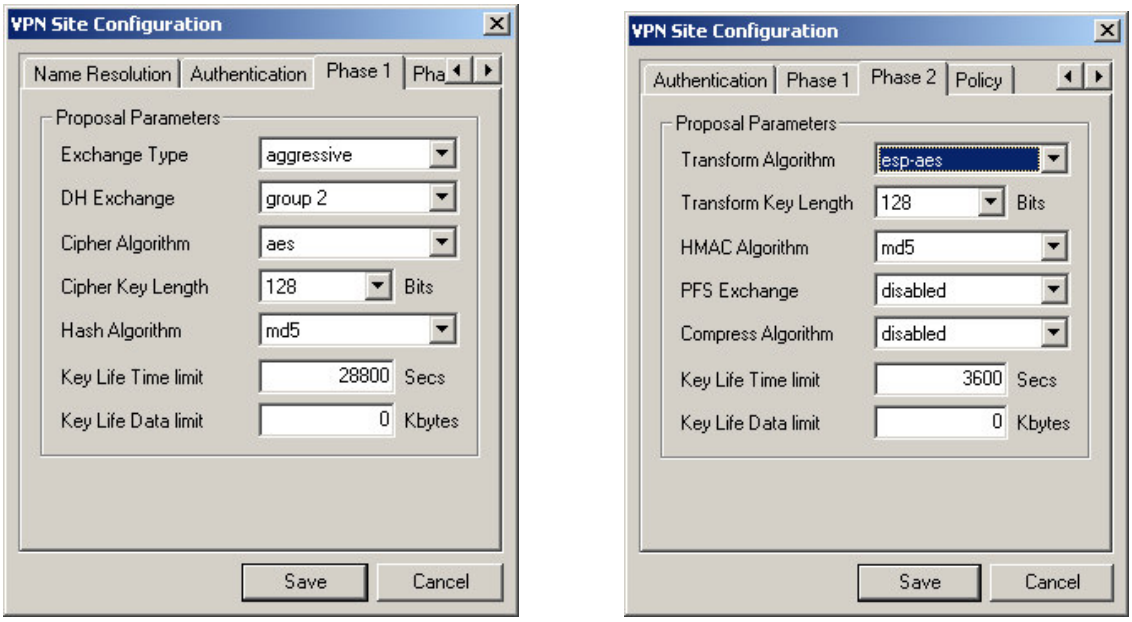
```
# openssl req -new -nodes -out req.pem
# openssl ca -out cert.pem -config -infiles req.pem
```

Vytvořený certifikát (cert.pem) a klíč (key.pem) je nutné přesunout do adresáře uvedeného v souboru racoon.conf čili do /usr/local/etc/racoon. Kořenový certifikát se musí bezpečným způsobem dostat ke klientovi a ten pomocí něj nastaví svoji aplikaci pro připojení k R1.

Nezbývá než nakonfigurovat klientský software na počítači teleworkera. Stanice vzdáleného pracovníka bude mít pravděpodobně nejrozšířenější operační systém MS Windows, pro který existuje řada aplikací poskytovaných pro nekomerční zdarma. V mém případě jsem zvolil Shrew Soft VPN Klient [22], který je nutné nastavit se stejnými parametry jako nástroj racoon (obr. 13a, 13b, 14a, 14b). V této chvíli již možné se připojit do firemní sítě a využívat jejích služeb.

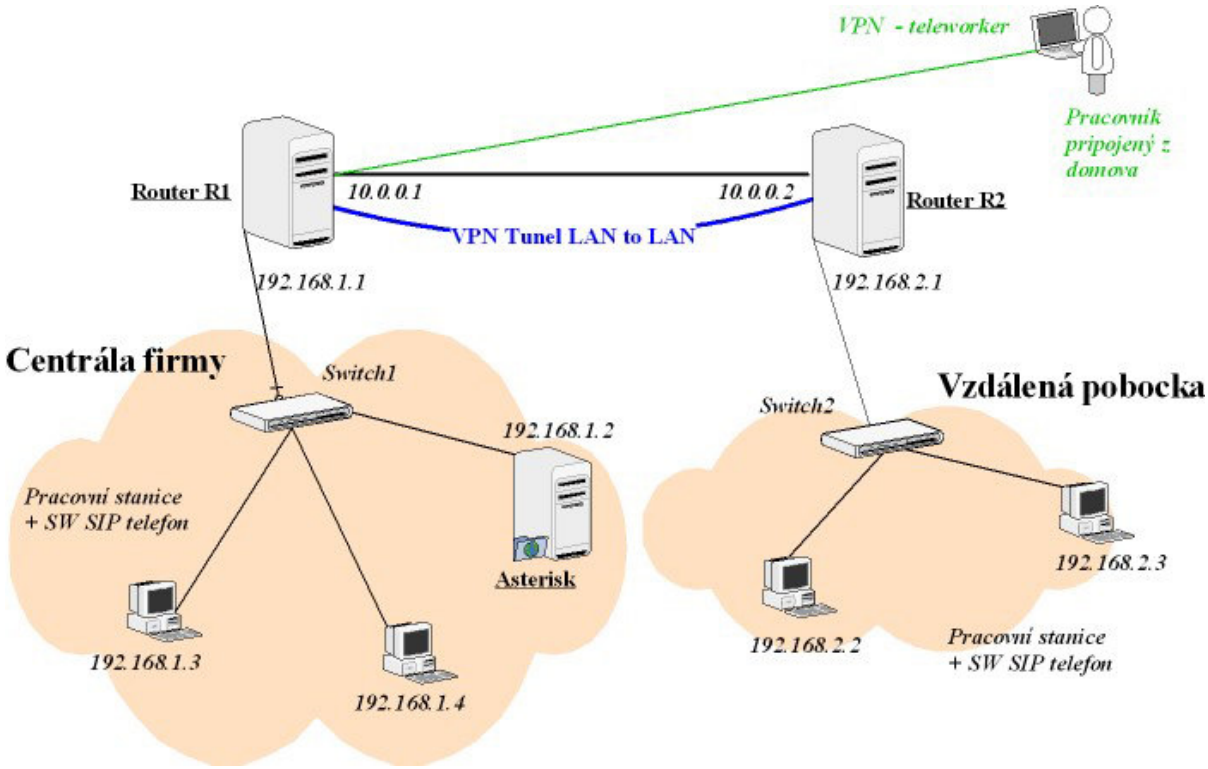



Obr.12 : a) b) Konfigurace klientské VPN aplikace - teleworker




Obr.13 : a) b) Konfigurace šifrování – teleworker

Na obr.14 je znázorněna výsledná podoba sítě, kterou jsem realizoval pomocí výše rozebraných technologií. Konfigurační soubory jednotlivých prvků sítě jsou uvedeny v příloze.




Obr.14 : Výsledné řešení

# 7. Závěr

Ve své práci jsem se zabýval rozborem technologií, které jsem dále využil při návrhu a realizaci komunikační sítě. Speciálně jsem se věnoval technologii Voice over IP, která umožňuje přenos digitalizovaného hlasu prostřednictvím paketů rodiny protokolů TCP/IP a je alternativním řešením ke klasické telekomunikační síti.

Při výběru technologií jsem se řídil především tím, aby výsledné řešení bylo užitečné, efektivní, perspektivní a finančně nenáročné. Těchto požadavků jsem dosáhl použitím nekomerčního open source softwaru a dnes již zastaralého hardware, který však posloužil pro demonstraci moderních technologií. Při vytváření sítě jsem využil technologii virtuální privátní sítě podle návrhu na obr. 7.

Páteřní infrastrukturu sítě jsem realizoval pomocí starších PC s operačním systémem FreeBSD, který se pro tyto účely skvěle hodí díky propracovanému systému síťových služeb. Obě PC jsem vybavil dvěmi síťovými rozhraními a vytvořil jsem z nich směrovací brány R1 (IP adresa 10.0.0.1) a R2 (IP adresa 10.0.0.2), které jsem propojil kříženým kabelem. Toto zapojení simuluje přístupovou síť k Internetu. Ke druhému síťovému rozhraní na každém routeru jsem připojil lokální síť 192.168.1.0 u R1 (simuluje sídlo firmy) a 192.168.2.0 u R2 (simuluje vzdálenou pobočku). Mezi směrovači jsem vytvořil statický IPSec tunel, který umožňuje šifrovaný přístup obou lokálních sítí ke sdíleným prostředkům firmy. Sdílené prostředky jsou v mém případě reprezentovány SIP serverem, ke kterému se registrují uživatelé z jednotlivých sítí. Ten je tvořen také starším PC s operačním systémem FreeBSD a telekomunikačním softwarem Asterisk, který je považován za špičku ve své třídě. Asterisk je nakonfigurován tak, aby spolu mohli komunikovat uživatelé z obou sítí a v případě nedostupnosti volaného mu můžou zanechat hlasovou zprávu, kterou si volaný později vyzvedne.

V poslední řadě bylo nutné umožnit přístup do sítě vzdáleným uživatelům, kteří nejsou připojeni pomocí statického tunelu a připojují se z různých míst např. zaměstnanci na cestách tzv. teleworkeři (mají pokaždé jinou IP adresu a není tedy možné dopředu nadefinovat tunel). Tento problém jsem vyřešil pomocí autentizace s použitím certifikátů. Tímto způsobem se mohou připojit k firemní síti i k SIP serveru a využívat služeb Voice over IP. To přináší nesporné výhody, jelikož všichni zaměstnanci mohou společně využívat síťové prostředky a komunikovat spolu neomezeně bez poplatků. Podoba výsledné funkční sítě je znázorněna na obr. 14.

# 8. Seznam použité literatury

**[1]** Herman, I.: Pokročilé komunikační technologie . Skriptum. Brno: 2005

**[2]** Lucas M.: Síťový operační systém FreeBSD – Podrobný průvodce. Computer Press. Brno : 2003

**[3]** Mock J.: FreeBSD. Neocortex. Praha : 2001

**[4]** Šilhavý P.: Spojovací a informační systémy. Skriptum. Brno : 2007

**[5]** Wallace K.:VoIP bez předchozích znalostí . Cisco Systems. Brno: 2007

**[6]** Wija T., Zukal D., Vozňák M.: Asterisk a jeho použití. Technická zpráva. CESNET. Praha : 2005

**[7]** Český statistický úřad

http://www.czso.cz/csu/2007edicniplan.nsf/engt/70002633B1/$File/97010704.pdf

**[8]** IPsec-Tools

http://ipsec-tools.sourceforge.net/

**[9]** IP telefonie

http://www.vda.cz/studenti/prace/hajek/iptelefonie.htm#nahoru

**[10]** IP telephony

http://www.iptelephony.org/

**[11]** Jak na OpenSSL

http://www.root.cz/clanky/jak-na-openssl-2/

**[12]** Meet OpenVPN

http://www.linuxjournal.com/article/7949

**[13]** OpenVPN

http://openvpn.net/index.php/documentation/howto.html

**[14]** OpenVPN - VPN jednoduše

http://www.root.cz/clanky/openvpn-vpn-jednoduse/

**[15]** Principy IP telefonie

http://www.cesnet.cz/doc/2004/zprava/voip.html

**[16]** Protokol RTP

http://www.javvin.com/protocolRTP.html

**[17]** Protokol SIP

http://atm.felk.cvut.cz/mps/referaty/2004/valat1/#impl

**[18]** Protokoly pro multimedia

http://atm.felk.cvut.cz/mps/referaty/RTP/MG/mps.html

**[19]** QoS - zabezpečení kvality služeb

http://www.svetsiti.cz/tutorialy.asp

**[20]** Quality of Service (QoS)

http://www.ten.cz/qos

**[21]** Racoon

http://netbsd.gw.com/cgi-bin/man-cgi?racoon.conf+5+NetBSD-current

**[22]** Shrew Soft VPN Client

http://www.shrew.net/?page=software

**[23]** Secure Socket Layer

http://cs.wikipedia.org/wiki/SSL

**[24]** Technické principy IP telefonie

http://www.telefonie.cz/zprava.asp?id=4892

**[25]** The FreeBSD Handbook

http://www.freebsd.org/doc/en/books/handbook/

**[26]** Úvod do VoIP

http://atm.felk.cvut.cz/mps/referaty/2004/valat1/#impl

**[27]** Virtual Private Network

http://en.wikipedia.org/wiki/Virtual_private_network

**[28]** Voice over IP

http://artax.karlin.mff.cuni.cz/~brain/voip.html

**[29]** Voice over IP Basics

http://www.voiptroubleshooter.com/basics/index.html

**[30]** VPN

http://www.czela.net/wiki/index.php/OpenVPN

**[31]** Využití IP sítí pro přenos hlasu

http://www.elektrorevue.cz/clanky/01015/.utf-8#propojeni

**[32]** World of Asterisk

http://www.asterisk.org/

# 9. Seznam příloh

**Příloha A – konfigurační soubory routeru R1**

**Příloha B – konfigurační soubory routeru R2**

**Příloha C – konfigurační soubory Asterisk**

## Příloha A

**/usr/local/etc/racoon/racoon.conf**

```
# R1
path pre_shared_key "/usr/local/etc/racoon/psk.txt" ;

path certificate "/usr/local/etc/racoon";
remote anonymous {
        exchange_mode aggressive,main;
        certificate_type x509 "cert.pem" "key.pem";
        generate_policy on;
        dpd_delay 20;
        ike_frag on;
        proposal {
                encryption_algorithm aes;
                hash_algorithm md5;
                authentication_method hybrid_rsa_server;
                dh_group 2;
                }
        }

mode_cfg {
        network 192.168.1.100;
        pool_size 10;
        netmask 255.255.255.0;
        }

sainfo anonymous {
        pfs_group 2;
        lifetime time 1 hour;
        encryption_algorithm aes;
        authentication_algorithm hmac_md5;
```

```
        }

remote 10.0.0.2 {
        exchange_mode main;
        proposal {
                encryption_algorithm aes;
                hash_algorithm md5;
                authentication_method pre_shared_key;
                dh_group 2;
        }


sainfo 10.0.0.2 {
        encryption_algorithm aes;
        authentication_algorithm hmac_md5;

        }
```

**/etc/rc.conf**

```
#---Konfiguracni soubor routeru 1---

hostname="Router1"
# external interface
ifconfig_rl0="inet 10.0.0.1  netmask 255.0.0.0"
# internal interface
ifconfig_rl1="inet 192.168.1.1  netmask 255.255.255.0"
defaultrouter="10.0.0.2"
gateway_enable="YES"
natd_enable="YES"
natd_interface="rl0"
firewall_enable=”yes”
```

## Příloha B

**/usr/local/etc/racoon/racoon.conf**

```
# R2
path pre_shared_key "/usr/local/etc/racoon/psk.txt" ;

path certificate "/usr/local/etc/racoon";

remote 10.0.0.1 {
        exchange_mode main;
        proposal {
                encryption_algorithm aes;
                hash_algorithm md5;
                authentication_method pre_shared_key;
                dh_group 2;
        }
      sainfo 10.0.0.1 {
        encryption_algorithm aes;
        authentication_algorithm hmac_md5;
                }
```

**/etc/rc.conf**

```
#---Konfiguracni soubor routeru 2---

hostname="Router2"
# external interface
ifconfig_rl0="inet 10.0.0.2  netmask 255.0.0.0"
# internal interface
ifconfig_rl1="inet 192.168.2.1  netmask 255.255.255.0"
defaultrouter="10.0.0.1"
gateway_enable="YES"
natd_enable="YES"
natd_interface="rl0"
```

firewall_enable=”yes”

## Příloha C

**sip.conf**

```
[general]
disallow=all
allow=ulaw
nat=yes
insecure=port,invite


[100]
type=friend
secret=100100
qualify=yes
host=dynamic
context=headquarters
mailbox=100@default



[101]
type=friend
secret=101101
qualify=yes
host=dynamic
context=headquarters
mailbox=101@default

[102]
type=friend
secret=102102
qualify=yes
host=dynamic
```

```
context=headquarters
mailbox=102@default

[103]
type=friend
secret=103103
qualify=yes
host=dynamic
context=headquarters
mailbox=103@default

[200]
type=friend
secret=200200
qualify=yes
host=dynamic
context=branch
mailbox=200@default

[201]
type=friend
secret=201201
qualify=yes
host=dynamic
context=branch
mailbox=201@default

[202]
type=friend
secret=202202
qualify=yes
host=dynamic
context=branch
mailbox=202@default
```

[203]
type=friend
secret=203203
qualify=yes
host=dynamic
context=branch
mailbox=203@default

**extensions.conf**

;
; Static extension configuration file. This is where you
; configure all your inbound and outbound calls in Asterisk.

[headquarters]

exten => _1XX,1,Dial(SIP/${EXTEN},20)
exten => _1XX,2,VoiceMail(${EXTEN}@default)
exten => _2XX,1,Goto(branch,${EXTEN:1},20)
exten => _2XX,2,VoiceMail(${EXTEN}@default)
exten => 9999,1,VoiceMailMain(${CALLERID(num)}@default)

[branch]

exten => _2XX,1,Dial(SIP/${EXTEN},20)
exten => _2XX,2,VoiceMail(${EXTEN}@default)
exten => _1XX,1,Goto(headquarters,${EXTEN:1},20)
exten => _1XX,1,VoiceMail(${EXTEN}@default)
exten => 9999,1,VoiceMailMain(${CALLERID(num)}@default)

**voicemail.conf**

```
[default]
maxmsg=20     ;maximalni pocet zprav pro 1 ucet

100 => 100100,100,,,
101 => 101101,101,,,
102 => 102102,102,,,
103 => 103103,102,,,
200 => 200200,200,,,
201 => 201201,201,,,
202 => 202202,202,,,
203 => 203203,203,,,
```