VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

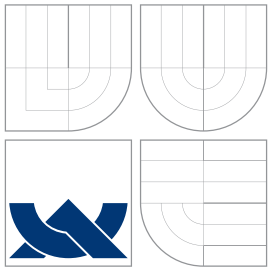
# PORTACE PROXY PRO ZABEZPEČENÍ ELEKTRONICKÉ POŠTY NA EMBEDDED ZAŘÍZENÍ

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

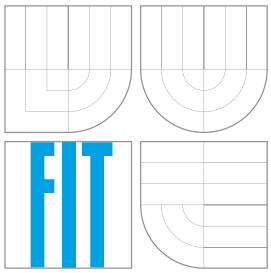
AUTOR PRÁCE
AUTHOR

JAN RICHTER

BRNO 2008



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

# PORTACE PROXY PRO ZABEZPEČENÍ ELEKTRONICKÉ POŠTY NA EMBEDDED ZAŘÍZENÍ

TRANSFER OF SECURITY EMAIL PROXY INTO EMBEDDED DEVICE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — JAN RICHTER
AUTHOR

VEDOUCÍ PRÁCE — Ing. JIŘÍ SCHÄFER
SUPERVISOR

BRNO 2008

# Zadání

## Portace proxy pro zabezpečení elektronické pošty na embedded zařízeni

1. Seznamte se s předchozími projekty implementujícími bezpečnou email proxy a zjistěte možnosti portace této proxy na embedded zařízení.
2. Upravte implementaci existující proxy tak, aby bylo možné ji portovat na embedded zařízení.
3. Navrhněte další možnosti zabezpečení proxy a popište je.
4. Implementujte některá tato zabezpečení a proveďte testování.

**Kategorie:** Bezpečnost

**Implementační jazyk:** C, C++

**Literatura:** Gutmann: Cryptographic Security Architecture, Springer, 2002

**Datum zadání:** 1. listopadu 2007

**Datum odevzdání:** 14. května 2008

# Licenční smlouva

Licenční smlouva je uložena v archivu Fakulty informačních technologií Vysokého učení technického v Brně.

# Abstrakt

Bakalářská práce se zabývá analýzou embedded zařízení a operačních systémů pro tato zařízení za účelem portace proxy pro zabezpečení elektronické pošty na některé z těchto zařízení. Dále se věnuje existujícímu řešení Mailproxy a způsobům zabezpečení elektronické pošty.



# Klíčová slova

elektronická pošta, mebedded zařízení, linux, zabezpečení, kryptografie, proxy, Cryptlib, uClibc, SMTP, POP3, IMAP, WRAP 2C



# Abstract

This bachelor's thesis deals with embedded devices and their operating systems analysis for transfer of security email proxy into one of these devices. It also describes already existing project Mailproxy and techniques of email securing.



# Keywords

electronic mail, single board computer, linux, security, cryptography, proxy, Cryptlib, uClibc, SMTP, POP3, IMAP, WRAP 2C


# Citace

Jan Richter: Portace proxy pro zabezpečení elektronické pošty na embedded zařízení, bakalářská práce, Brno, FIT VUT v Brně, 2008

# Portace proxy pro zabezpečení elektronické pošty na embedded zařízení

## Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Jiřího Schäfera

.....
Jan Richter
9. května 2008

## Poděkování

Rád bych poděkoval svému vedoucímu, Jiřímu Schäferovi, za podporu a rady poskytnuté při vypracovávání práce. Dále bych rovněž chtěl poděkovat Stanislavu Židkovi za vysvětelní struktury konfiguračního souboru projektu Mailproxy, což bylo pro mou práci neocenitelné a v neposlední řadě moje díky patří také Liboru Valentovi za vytvoření linuxové distribuce K240, o kterou se má práce z velké části opírá.

# Obsah

# Kapitola 1

# Úvod

Elektronická pošta je hned po WWW druhou nejpoužívanější službou internetu a v dnešní době je zcela nepostradatelnou součástí našeho života. Ovšem stejně jako u klasické pošty, i zde je třeba počítat s tím, že přenos pošty od odesílatele k adresátovi zprostředkovávají jiné osoby (servery) a vždy hrozí nebezpečí, že se citlivé informace dostanou do nesprávných rukou nebo bude zpráva podvržena, proto vzniklo a stále vzniká velké množství různě účinných metod zabezpečení jak obsahu a neporušenosti zprávy, tak identity odesílatele.

Problém zabezpečení přepravovaných informací lidé řeší již od samotných počátků písemné komunikace a byly používány různé více či méně spolehlivé metody zabezpečení. K popisu některých z nich se vrátíme v následující kapitole společně s úvodem do historie a principů elektronické pošty.

Ve třetí kapitole se budeme zabývat výběrem vhodného embedded zařízení pro portaci proxy pro zabezpečení elektronické pošty. Probereme výhody a nevýhody dvou desek, které jsem měl pro práci k dispozici, srovnáme některé důležité parametry a praktické zkušenosti s oběmi deskami a vybereme tu pravou.

Ve čtvrté kapitole se potom budeme věnovat popisu existujícího řešení aplikace proxy pro zabezpečení elektronické pošty, které vzniklo na naší fakultě a knihovně Cryptlib, se kterou je aplikace úzce svázána.

Pátá kapitola se zabývá samotnou portací aplikace Mailproxy na vybrané embedded zařízení, rozdíly ve standardních knihovnách, které měly na portaci vliv a problémy, které při portaci nastaly. Také budou popsány různé další způsoby zabezpečení aplikace a celého systému.

Poslední dvě kapitoly budou věnovány možnostem využití embedded zařízení s Mailproxy v reálném provozu, bezpečnostním rizikům aplikace a budou navrženy kroky dalšího vývoje na projektu.

# Kapitola 2

# Elektronická pošta a bezpečnost

Počátky elektronické pošty sahají do šedesátých let dvacátého století, kdy vznikala potřeba zasílání zpráv mezi jednotlivými uživateli počítačů se sdíleným časem. Začaly tedy vznikat první programy, které umožňovaly výměnu textových zpráv a také real-time chaty mezi uživateli přihlášenými na různých terminálech, ovšem stejného počítače.

Na začátku sedmdesátých let tým okolo Raye Tomlinsona vyvíjel operační systém TENEX na počítači PDP-10 a na něm konzolové programy `sndmsg` a `readmail` pro práci s lokální elektronickou poštou. V roce 1971 Ray Tomlinson rozšířil program `sndmsg` promocí programu `cpynet`, schopného kopírovat soubory po síti, a odeslal historicky první email po síti Arpanet, který adresoval svým kolegům a v němž popisoval, jak používat rozšířený`sndmsg` k zaslání emailu po síti.

Ray Tomlinson začal pro oddělení jména uživatele a počítače používat znak „zavináč“, což se ujalo a stalo se součástí specifikace adres elektronické pošty v RFC-733 a později RFC-822.

V následujícím roce byly do specifikace protokolu FTP[1] (v RFC-385) přidány příkazy `MAIL` a `MLFL` pro umožnění přenosu elektronické pošty tímto protokolem. FTP protokol byl pro zasílání elektronické pošty používán až do osmdesátých let, kdy vznikl protokol SMTP, který byl pro tento účel mnohem vhodnější.

Pro snazší čtení a správu příchozí pošty později vznikla sada maker pro textový editor TECO[2], pod názvem `rd`, která byla dále rozvíjena přes `nrd`, `wrd` a `bananard` až v první program, který sloužil jak k přijmu, tak odesílání elektronické pošty, s názvem `msg`.

V této době začalo vznikat velké množství různých formátů elektronické pošty a proto v roce 1977 pánové D. Crocker, J. Vittal, K. Pogran a A. Henderson analyzovali většinu těchto formátů a vytvořili jednotnou specifikaci elektronické pošty v RFC-733, později revidovaném a nahrazeném RFC-822 z roku 1982.[2]

## 2.1 Protokoly a standardy

V následujících odstavcích krátce popíšu několik nejzákladnějších protokolů a standardů, využívaných v elektronické poště, které bude nutno znát k dalšímu pokračování.

[1] File Transfer Protocol
[2] Text Editor and COrrector

### 2.1.1 SMTP

Simple Mail Transfer Protocol (SMTP) je protokol definovaný v RFC-821 a je dnes nosným pilířem elektronické pošty, stará se o doručování pošty od odesílatele až k příjemci.

Jednotlivé servery, které se podílejí na doručení emailu pomocí SMTP protokolu, jsou označovány jako Mail Transfer Agent (MTA), na cílovém serveru potom běží Mail Delivery Agent (MDA), který se stará o doručení zprávy do schránky adresáta.

SMTP server naslouchá na portu 25 nebo 465 při šifrování komunikace přes SSL a základními příkazy, potřebnými k odeslání zprávy, jsou `HELO` pro začátek transakce, `MAIL FROM:` pro zadání odesílatele pošty, `RCPT TO:` pro zadání příjemců, `DATA` pro uvození těla emailu, ukončeného tečkou na novém řádku a `QUIT` pro ukončení spojení se serverem. Standardně je SMTP protokol textový ASCII protokol a nepodporuje tak přenos diakritických znaků a binárních souborů, proto vznikl standard MIME, viz. 2.1.4.

**ESMTP** (Extended SMTP) je rozšíření protokolu SMTP, definované v RFC-1869. Při navázání spojení zasílá klient namísto příkazu `HELO`, příkaz `EHLO`, na který mu ESMTP server odpoví mj. seznamem ESMTP rozšíření, které podporuje. Mezi nejpoužívanější ESMTP rozšíření patří například `8BITMIME`, umožňující zasílání 8-bitově kódovaných zpráv, `DSN` pro oznámení o doručení zprávy, `SMTP-AUTH` pro autentifikaci odesílatele, `PIPELINING` pro proudové zpracování příkazů, `STARTTLS` pro zahájení zabezpečeného přenosu a další, kterými se zde dále nebudeme zabývat.

### 2.1.2 POP3

Post Office Protocol v. 3 (POP3) je protokol, který používá aplikace příjemce elektronické pošty, Mail User Agent (MUA), pro kontrolu a stahování pošty z poštovního serveru. Zprávy se z POP3 serverů defaultně stahují do klientovy poštovní aplikace a ze serveru se mažou. Specifikace protokolu POP3 je definována v RFC-1939.

POP3 servery obvykle naslouchají na portu 110, případně na portu 995 při komunikaci šifrované pomocí SSL. Základními příkazy zde jsou `USER` a `PASS` pro přihlášení uživatele k serveru, `STAT` pro zjištění počtu a velikosti zpráv, `LIST` pro seznam zpráv na serveru, `RETR` pro získání konkrétní zprávy ze serveru, `DELE` pro odstranění zprávy a `QUIT` pro ukončení spojení.

### 2.1.3 IMAP

Internet Message Access Protocol (IMAP) je stejně jako POP3 využíván ke komunikaci klientské aplikace (MUA) s poštovním serverem, na kterém má uživatel svou stránku, největším rozdílem oproti POP3 protokolu je to, že IMAP uchovává zprávy na serveru a stahuje je jen, když si je uživatel vyžádá, umožňuje tak flexibilnější přístup ke schránce z různých míst a správu složek a stavů zpráv na serveru. Dnes je používána první revize verze 4 tohoto protokolu, jejíž specifikace je v RFC-3501 a všechny zde popsané vlastnosti se týkají právě této verze.

Základní port, na kterém naslouchají IMAP servery je port číslo 143, respektive 993 při komunikaci přes SSL/TLS.

Klienti, připojující se pomoci protokolu IMAP, zůstávají připojeni po celou dobu běhu programu a umožňují tak například rychlejší zjištění, že došel nový email nebo efektivnější práci s velkým množstvím zpráv. Oproti POP3 je umožněno připojení většího množství klientů ke stejné schránce v jeden okamžik a díky tomu, že informace o stavu zprávy

jako příznak zda byla zpráva přečtena, bylo na ni odpovězeno a pod. jsou uloženy na serveru, můžou takto uživatelé tyto informace sdílet mezi sebou. Velikou výhodou proti POP3 je možnost přijímat odděleně jednotlivé MIME (viz část 2.1.4)části zprávy a klient tak například nemusí stahovat společně se zprávou i všechny její přílohy. Dále lze pomocí IMAP protokolu přímo na serveru vyhledávat zprávy podle různých kritérií a existuje zde mechanismus pro přidávání rozšíření protokolů, podobně jako u ESMTP.

### 2.1.4 MIME

Multipurpose Internet Mail Extensions (MIME) je standard, který rozšiřuje formát emailu definovaný v RFC-822 o možnost přenosu 8-bitově kódovaného obsahu a možnost zasílání jakýchkoliv příloh. Přestože bylo MIME vyvinuto pro rozšíření elektronické pošty, dnes se využívá k popisu obsahu i v dalších oblastech internetu, například službě WWW v protoklu HTTP. Základní popis MIME je rozdělen v RFC-2045 – RFC-2049, ale existuje spousta dodatků a rozšíření v dalších RFC.

MIME přidává do hlaviček emailové zprávy další své hlavičky, kde specifikuje verzi MIME (hlavička `MIME-Version`), typ přenášených dat (`Content-Type`), způsob kódování přenášených dat (`Content-Transfer-Encoding`) a případně další.

**Hlavička Content-Type** definuje typ přenášených dat ve formátu `mediatype/subtype`, například `text/blank` pro čistý text, `image/jpeg` pro obrázek ve formátu jpeg a další. Seznam jednotlivých MIME typů lze nalézt na webových stránkách organizace IANA[3] na `http://www.iana.org/assignments/media-types/`.

**S/MIME** (Secure / Multipurpose Internet Mail Extensions) je rozšíření standardu MIME o možnosti zabezpečení obsahu elektronické pošty spojením s průmyslovým standardem PKCS #7[4].

## 2.2 Úvod do šifrování

V této části si uděláme lehký úvod do různých historických i současných metod zabezpečení přepravovaných informací.

### 2.2.1 Steganografie

Steganografie je jednou z nejstarších metod zabezpečení informace, název je odvozen z řeckého slova *stegenós*, které znamená ukrýt. A jak název napovídá, jde o ukrývání zprávy způsobem, aby si pozorovatel neuvědomil, že komunikace vůbec probíhá, například použitím „neviditelného“ inkoustu, vyrýváním textu do dřevěných destiček pod vrstvu vosku, na kterou se teprve mělo psát nebo později třeba použitím tzv. mikroteček na filmu.

### 2.2.2 Substituční šifry

Podstatou substituční šifry je nahrazování jednotlivých znaků v původním textu jinými podle určitých pravidel. Nejjednodušší substituční šifrou je například Caesarova šifra, kterou používal pro vojenskou komunikaci Julius Caesar. Principem je nahrazení každého písmene

[3] Internet Asigned Numbers Authority
[4] Public Key Cryptography Standards

zprávy písmenem, které je v abecedě o tři znaky později. Caesarova šifra je velice slabá a je snadno luštitelná kryptoanalýzou na základě frekvenční analýzy textu.

### 2.2.3 Vigenèrova šifra

Vigenèrova šifra je velice podobná Caesarově šifře, ovšem nepoužívá posun o tři znaky, ale je definováno heslo, jehož jednotlivé znaky určují posunutí původního textu tak, že se text rozdělí do bloků o stejné délce jako je délka hesla a každý znak se sečte s odpovídajícím znakem hesla. Tato šifra je podstatně silnější, než Caesarova šifra, a jelikož pravděpodobnosti výskytu jednotlivých znaků abecedy jsou heslem změněny, nefunguje zde frekvenční analýza.

### 2.2.4 Vernamova šifra

Vernamova šifra, také nazývaná *one-time pad*, je jediná známá šifra, u které bylo exaktně dokázáno, že je nerozluštitelná[5]. Šifra je velice podobná Vigenèrově šifře, ale heslo musí mít stejnou délku, jako text, který se bude šifrovat a musí být dokonale náhodné, činnost softwarových generátorů pseudonáhodných čísel je možné poměrně snadno předvídat, proto nejsou pro generování klíče vhodné. Stejný klíč rovněž nesmí být použit opakovaně.

Tato šifra i přes svou sílu není příliš využívána především kvůli délce klíče, který je třeba bezpečně doručit příjemci šifrované zprávy. V historii byla používána například za studené války na horké lince mezi Washingtonem a Moskvou, kdy si USA se Sovětský svazem vyměnily pásky s náhodně vygenerovanými klíči, které byly poté používány k šifrování komunikace.

Nevýhody manipulace s klíčem dnes odstraňují metody kvantové kryptografie, využívající poznatků kvantové mechaniky.

## 2.3 Moderní symetrické šifry

Symetrické šifrování je založeno na použití jediného klíče na zašifrování i dešifrování zprávy, což přináší velký problém s bezpečnou distribucí klíče mezi účastníky komunikace. Výhodou je, že symetrické šifrovací algoritmy jsou o několik řádů méně náročné na výpočetní výkon počítače, než asymetrické.

Symetrické šifry se dělí do dvou základních skupin, na šifry blokové a proudové.

### 2.3.1 Blokové šifry

Blokové šifry při šifrovacím procesu zpracovávají šifrovaná data po stejně dlouhých blocích (poslední blok je případně vhodně doplněn) o délce většinou 64 nebo 128 bitů.

Mezi nejznámější symetrické blokové šifry patří:

**AES** (Advanced Security Standard), nazývaná také **Rijndael** podle autorů Joana Daemena a Vincenta Rijmena, je bloková šifra přijatá jako šifrovací standard Americké vlády. Šifra je nástupcem šifry DES, která byla roku 1997 prolomena. AES používá bloky o velikosti 128 bitů a klíče o velikostech 128, 192 nebo 256 bitů.

**Blowfish** je šifra vytvořená roku 1993 Brucem Schneierem, využívá bloků o délce 64 bitů a klíčů o délce 32 – 448 bitů. Nástupcem se v roce 1998 stala šifra **Twofish** stejného autora.

[5] důkaz proveden C. E. Shannonem v roce 1949

**DES** (Data Encryption Standard) je šifra vyvinutá v 70-tých letech v IBM, která byla od roku 1976 používána jako standardní šifra Americké vlády. DES byla prolomena roku 1997 a není dnes považována za bezpečnou. Velikost bloku je 64 bitů a používá 56-ti bitové klíče.

**Triple DES** (Triple Data Encryption Standard) šifra vznikla na konci 70-tých let jako reakce na zjištění, že 56-ti bitový klíč není dostatečná ochrana proti útokům hrubou silou[6]. Zvoleným řešením bylo jednoduché zřetězení stávající šifry DES s použitím různých klíčů pro jednotlivé kroky. Velikost bloků této šifry je stejný jako u DES, tedy 64 bitů, velikost klíče je buď 112 bitů u varianty 2TDES, kdy je použit stejný klíč v prvním a třetím kroku šifrovacího procesu nebo 168 bitů při použití rozdílných klíčů v každém kroku u varianty 3TDES.

**IDEA** (International Data Encryption Algorithm) šifra byla vytvořena v roce 1991 na Eidgenössische Technische Hochschule v Zürichu, kde ji vymysleli Xuejia Lai a James Massey, jako nástupce šifry PES (Proposed Encryption Standard) a původně byla pojmenována IPES (Improved PES). IDEA používá bloky o délce 64 bitů a 128-mi bitové klíče. Šifra je patentována v řadě zemí, ale je volně použitelná pro nekomerční účely, tento patent by měl vypršet v letech 2010 – 2011.

### 2.3.2 Proudové šifry

Tyto šifry jsou méně používané, než šifry blokové a zpracovávají text bit po bitu, zástupci jsou například

**FISH** (Fibonacci SHrinking) šifra byla publikována v roce 1993 firmou Siemens, jde o rychlou softwarovou šifru využívající generátory pseudonáhodných čísel založených na zobecnění fibonacciho posloupnosti a shrinking generator.

**RC4** (Rivest Cipher 4) šifra byla vytvořena Ronem Rivestem v RSA Security roku 1987. Původně šlo o uzavřený kód, ale v roce 1994 byl kód anonymně zveřejněn na mailinglistu Cypherpunks a později bylo ověřeno, že jde o autentický kód srovnáním výstupu s proprietárním softwarem využívajícím licencované RC4. Přestože jde dnes o otevřený algoritmus, RSA Security ho nikdy neuvolnila, proto je často nazýván také ARCFOUR nebo ARC4 (Alleged RC4), aby se vyhnulo případným problémům s licencí.

Algoritmus je dnes součástí běžně používaných šifrovacích protokolů jako WEP, WPA nebo SSL.

## 2.4 Moderní asymetrické šifry

Předností asymetrických šifer je především to, že k zašifrování zprávy se používá zcela odlišný klíč, než k jejímu dešifrování, čímž odpadá odvěký problém s bezpečnou distribucí dešifrovacích klíčů. Jeden z klíčů bývá označován jako privátní a druhý jako veřejný.

V praxi si uživatel za pomocí nástrojů šifrovací aplikace vygeneruje tuto dvojici klíčů, privátní klíč si bezpečně uschová a veřejný může zaslat všem osobám, které mu budou zasílat šifrované zprávy. Odesílatel zprávu zašifruje veřejným klíčem příjemce, dešifrovat

[6] metoda postupného zkoušení všech možných klíčů

pak lze již pouze privátním klíčem příjemce, který má příjemce bezpečně uložen u sebe. Obrazně by se tento princip dal přirovnat k situaci, kdy by příjemce předal odesílateli trezor, který je zamčený, ale má otevřené dvířka, poté, co do něj odesílatel vloží zprávu a dvířka zabouchne, už jej nikdo kromě odesílatele, který jako jediný vlastní klíč, neodemkne.

Mezi nejznámější asymetrické šifry patří například RSA a DSA.

**RSA** je algoritmus široce využívaný v protokolech pro elektronický finanční styk a jde o první algoritmus, který je použitelný jak pro šifrování, tak pro elektronické podepisování. RSA je považováno za bezpečné při použití dostatečně dlouhých klíčů a aktualizovaných implementací.

Algoritmus vznikl na konci sedmdesátých let na Massachusetts Institute of Technology v Cambridge a autory byli Ron Rivest, Adi Shamir a Leonard Adleman (a podle nich byl pojmenován). RSA podléhalo v USA do roku 2000 patentu. Jelikož je tato šifra výrazně pomalejší, než například DES, bývá kombinována se symetrickými šiframi, viz 2.5.

**DSA** (Digital Signature Algorithm) je algoritmus, určený k digitálnímu podepisování elektronické pošty. Autorem je David W. Kravitz, který je také od roku 1991 držitelem patentu na tento algoritmus.

## 2.5 Hybridní šifrování

Hybridní šifrování spojuje výhodu bezpečnosti asymetrických šifer společně s rychlostí symetrického šifrování. Při této metodě se využívá jednorázového náhodně generovaného klíče, kterým se symetricky zašifruje zpráva (u které se předpokládá, že bude delší, než klíč) a poté se již asymetrickými metodami zašifruje klíč, který se pošle příjemci společně se zprávou. Příjemce nejdříve dešifruje pomocí svého privátního klíče přiložený klíč a pomoci něj zbytek zprávy.

**PGP** (Prety Good Privacy) je počítačový program, který umožňuje šifrování/dešifrování a autentizaci, často je používán pro podepisování a šifrování elektronické pošty. Jeho první verze byla vytvořena v roce 1991 Philem Zimmermannem.

Původní verze používala k distribuci veřejných klíčů systém tzv. pavučiny důvěry v kontrastu s tehdejšími systémy založenými na hierarchickém systému X.509, který využívá certifikační autority, ten byl do PGP přidán později. Dnes PGP a všechny podobné produkty pracují podle OpenPGP standardu, definovaného v RFC-4880.

Více o problematice šifrování je možno nalézt na webu a v literatuře [10], [8].

# Kapitola 3

# Analýza dostupných embedded zařízení

Prvním krokem byl výběr vhodné embedded desky, které se dále přizpůsobovaly požadavky při výběru operačního systému, knihoven a aplikací pro bezpečnou emailovou proxy.

Z nabízených SBC[1] desek nejlépe vycházely Routerboard RB-112 a Wireless Router Application Platform (WRAP) 2C, které jsem důkladně otestoval, porovnal a vybral tu vhodnější.

## 3.1 Routerboard RB-112

Routerboard RB-112 je Single Board Computer, plně kompatibilní se standardní architekturou MIPS32 s PCI sběrnicí. Instalovaný procesor MIPS 4Kc používá *little-endian* kódování bytů, obsahuje TLB Memory Management Unit, ale neobsahuje Floating Point Unit. Deska RB-112 je dodávana s operačním systémem MikroTik RouterOS, který je stavěný pro snadnou konfiguraci směrování, ovšem není operačním systémem v tom pravém slova smyslu a neumožňuje spouštění dalšího software. Jiné operační systémy nejsou deskou podporovány, což byl zásadní a rozhodující nedostatek při výběru desky pro portaci proxy pro zabezpečení elektronické pošty.

### 3.1.1 MikroTik RouterOS v. 3.6

MikroTik RouterOS je vynikající operační systém pro routery, který velice snadno umožňuje konfiguraci síťových rozhraní, obsahuje telnet a ssh server pro konzolový přístup,webový server s grafickým konfiguračním rozhraním a propracovanou grafickou konfigurační utilitu RouterOS WinBox pro operační systém Windows.[2] Lze zde snadno nakonfigurovat cokoliv, co by se od routeru dalo očekávat, je zde plná podpora IPv6, traffic shapingu, směrovacích protokolů PIM, RIP, RIPng, OSPF, BGP a MME nebo třeba využívání RADIUS serverů, dále lze vykreslovat grafy jakékoliv činnosti routeru, sledovat všechna navázaná spojení a to vše z obrazu disku o velikosti pouhých 20MB.

Jak grafické, tak konzolové rozhraní je velice intuitivní a ovládání z konzole je velice podobné operačnímu systému Cisco IOS. Kdybych si vybíral operační systém čistě pro router, neváhal bych a volil MikroTik RouterOS, pro požadavky emailové proxy je ovšem

[1] Single Board Computer

[2] RouterOS WinBox lze bez problému spustit i na Linuxu pod windows emulátorem wine

z důvodu nemožnosti instalace uživatelských aplikací tento operační systém, jak již bylo zmíněno dříve, nepoužitelný.

### 3.1.2 Parametry RB-112[9]

**Procesor:** MIPS 4Kc 175MHz (little-endian)

**Operační paměť:** vestavěný 16MB SDRAM paměťový čip

**Bootloader:** RouterBOOT, 1Mbit Flash

**Permanentní paměť:** vestavěný 64 nebo 128MB paměťový čip NAND

**Ethernet:** 1× 10/100 Mbit/s port s podporou Auto-MDI/X a IEEE 802.3af (PoE)

**Sloty:** 2× MiniPCI Typ IIIA/IIIB

**Sériové rozhraní:** asynchronní DB9 RS232C

**LED:** 1× napájení, 2×2 pro miniPCI sloty a 1× uživatelská LED

**Repro:** 1× Mini PC-Speaker

**Napájení:** stejnosměrné 11–60V přes napájecí konektor nebo 12V přes POE

**Rozměry:** 14,0cm × 8,5cm

**Hmotnost:** 92g (bez MiniPCI karet)

**Spotřeba:** 3–4W bez MiniPCI karet, max. 10W

## 3.2 WRAP 2C

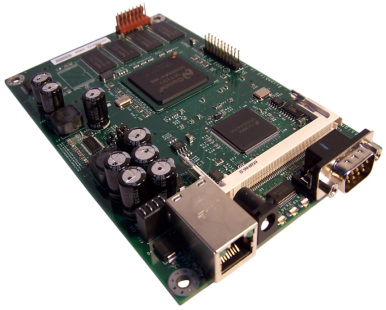
WRAP (Wireless Router Application Platform) 2C je SBC deska na bázi i586, optimalizovaná pro použití jako směrovač paketů s nízkou spotřebou elektrické energie. Výrobcem je švýcarská firma PC Engines GmbH, která se specializuje na výrobu hardware do embedded zařízení. Deska podporuje standard IEEE 802.3[3] Power over Ethernet, čímž odpadá nutnost napájení samostatným kabelem. Stejně jako RouterBoard RB-112 je deska WRAP 2C rozšiřitelná pomocí dvou miniPCI slotů například o WiFi adaptéry.

Je zde dobrá podpora operačních systémů založených na FreeBSD, Linuxu, NetBSD, OpenBSD a také specializovaných operačních systémů jako například Mikrotik RouterOS.

WRAP obsahuje tinyBIOS™, vyvinutý firmou PC Engines GmbH. TinyBIOS je napsaný v A386 assembleru speciálně pro použití v embedded zařízeních, na WRAPu umožňuje nastavení frekvence sériového portu, geometrie CF karty a aktualizace po sériovém portu pomocí protokolu Xmodem.

O dalších praktických zkušenostech s deskou WRAP 2C je možno se dočíst na webu [5].

[3] http://standards.ieee.org/getieee802/802.3.html



Obrázek 3.1: SBC deska WRAP 2C

### 3.2.1 Parametry WRAP 2C[6]

**Procesor:** 266MHz AMD Geode SC1100 CPU

**Operační paměť:** 64MB SDRAM

**Rozšiřitelnost:** 2× MiniPCI slot, možnost připojení USB redukce

**Ethernet:** 1× National DP83816 Ethernet port s podporou IEEE 802.3af (PoE)

**Permanentní paměť:** 1× CompactFlash slot

**Sériové rozhraní:** DB9 RS232 port

**Watchdog:** Watchdog timer vestavěný v CPU

**Monitor:** teplotní čidlo LM77 s nastaveným resetem při 80°C

**Napájení a spotřeba:** 7–18V DC, 2–5W podle instalovaných MiniPCI karet

**LED:** 3× programovatelné LED na čelním panelu, 1× programovatelné tlačítko

**Další sběrnice:** $I^2C$ bus (dvouvodičová synchronní, sdílená), LPC bus (nástupce ISA)

**BIOS:** tinyBIOS$^{TM}$

**Rozměry:** 10,0cm × 16,0cm

## 3.3 Srovnání embedded zařízení

V tabulce 3.1 je uvedeno porovnání obou desek z hlediska požadavků, které jsem na SBC pro portaci proxy pro zabezpečení elektronické pošty měl. Tučně je vždy zvýrazněn parametr u té desky, která daný požadavek splňovala lépe.

Podpora operačního systému Linux byla zásadním požadavkem na embedded zařízení, stejně jako ethernetový port, který ovšem bývá u desek, určených k použití jako routery, samozřejmostí, Routerboard zde má mírně navrch v tom, že jeho ethernetový port podporuje Auto-MDI/X a tedy nevyžaduje použití správného typu kabelu (přímého/kříženého) pro připojení do sítě.

Nízká spotřeba elektrické energie je jeden z důvodů k portaci na embedded zařízení, proto i tento parametr hrál při výběru roli, ovšem rozdíl těchto hodnot mezi deskami zde

není nikterak zásadní. Výkon je poměrně důležitý parametr, jelikož kryptografické algoritmy jsou výpočetně poměrně náročné. Menší rozměry jsou výhodou, ale nejde zde již o nikterak zásadní parametr.

| **Můj požadavek** | **Routerboard RB112** | **WRAP 2C** |
|---|---|---|
| Podpora OS Linux | NE | **ANO** |
| Ethernet port | **ANO, Auto-MDI/X** | **ANO** |
| Podpora Wi-Fi | **ANO, miniPCI** | **ANO, miniPCI** |
| Nízká spotřeba el. energie | 3–10W | **2–5W** |
| Rozšířitelná paměť | NE, interních 128MB | **ANO, CF slot** |
| Výkon | 175MHz CPU, 16MB RAM | **266MHz CPU, 64MB RAM** |
| Malé rozměry | **140mm × 85mm** | 180mm × 100mm |

Tabulka 3.1: Srovnání dostupných SBC desek

Jak je vidět z tabulky, po naprosté většině stránek v našem srovnání, vychází jako vhodnější kandidát deska WRAP 2C, která proto byla také pro další řešení projektu vybrána.

# Kapitola 4

# Existující řešení Mailproxy

Jako základ své práce jsem si vybral projekt Mailproxy Bc. Stanislava Židka, který jej zpracovával jako svou bakalářskou práci na FIT VUT v Brně v akademickém roce 2006/2007.

## 4.1 Co je to emailová proxy

Na úvod bych mírně odbočil od popisu konkrétního řešení a pokusil se blíže popsat, co pojem „emailová proxy“ vlastně znamená. Proxy servery se na internetu používají především pro službu www nad protokolem HTTP, a to za účelem ochrany soukromí uživatele, zvýšení výkonu komunikace, zvýšení bezpečnosti, filtrování provozu a nebo připojení více klientů k internetu.

Typický webový proxy server přijímá požadavky klientů na webové stránky z konkrétních webserverů v Internetu, ty zpracuje podle předem stanovených pravidel a vrací klientům odpověď stejně, jako by klient komunikoval přímo s cílovým webserverem.

Ochrana soukromí uživatele je zde zajištěna tím, že cílovému serveru příjde požadavek z IP adresy proxy serveru a nikoli z adresy uživatele, který takto zůstává utajen. Výkon komunikace proxy server zvyšuje tím, že přeposílané webové stránky si ukládá lokálně a při stejném dotazu jiného klienta již nemusí stránku vyžadovat od vzdáleného serveru, ale v případě, že ještě nevypršela platnost stránky, pošle klientovi její lokální kopii. Bezpečnost uživatelů proxy server může zvyšovat například integrací antiviru a filtrováním nebezpečného obsahu. Pro připojení více klientů se dnes proxy servery příliš nepoužívají, protože lépe tento úkol plní NAT. Proxy server tedy hraje roli prostředníka mezi klientem a serverem.

Z předchozího popisu je zjevné, že emailová proxy bude muset pracovat trochu jinak než webová, nebude si rozhodně uchovávat žádná data pro poskytnutí jiným uživatelům, ale stále bude plnit roli prostředníka mezi klientem a serverem. Data přicházející od uživatele bude před odesláním do internetu šifrovat a podepisovat a data z internetu se bude pokoušet dešifrovat a bude u nich ověřovat případné podpisy. Předností Mailproxy bude tedy především transparentní zabezpečení elektronické pošty a přemístění konfigurace všech stanic, potažmo uživatelů, na jedno místo.

## 4.2 Struktura Mailproxy

Aplikace Mailproxy je implementována v jazyce C, a to především kvůli rychlosti, jednoduchosti a přenositelnosti kódu. Pro kryptografické funkce je zde využíváno knihovny

Cryptlib,[1] která bude blíže představena v sekci 4.4.

Mailproxy se skládá ze tří nezávislých konkurentních serverů, u kterých je možno nastavit, zda budou nebo nebudou spouštěny a jednoho volitelného daemona. První tři servery korespondují se službami, které Mailproxy kontroluje, tedy SMTP, POP3 a IMAP. Konfigurační daemon, v originální práci [11] nazýván konfiguračním serverem, což dle mého názoru není zcela přesné (viz. popis v části 4.2.4), slouží k podpoře vzdálené konfigurace Mailproxy. Dále pak obsahuje utilitu `keymgr` pro generování a správu uživatelských certifikátů.

### 4.2.1 SMTP server

SMTP server Mailproxy přijímá požadavky klientů na předem definovaném portu podle specifikace SMTP protokolu, nejprve přečte všechny hlavičky a tělo zprávy, poté projde konfigurační soubor a podle hlaviček v něm dohledá odpovídající politiku, která stanoví, zda se má zpráva šifrovat a/nebo podepisovat, jaké klíče k tomu mají být použity a skutečný SMTP server, na který bude upravená zpráva předána.

### 4.2.2 POP3 server

POP3 server je dalším modulem Mailproxy, který naslouchá na svém portu a požadavky přeposílá na správný POP3 server, dohledaný v konfigurační databázi podle autentizačních příkazů, problémem zde může být rozluštění správného cílového POP3 serveru, jelikož se může teoreticky stát, že různí uživatelé na stejné síti budou mít stejný login na dvou různých POP3 serverech. Doporučeným řešením nastavení klienta aby použil celou emailovou adresu jako přihlašovací jméno k Mailproxy.

Odpovědi od skutečného POP3 serveru jsou před zasláním klientovi podle potřeby dešifrovány a je ověřen elektronický podpis.

### 4.2.3 IMAP server

IMAP server Mailproxy je ve své podstatě velice podobný POP3 serveru, pouze komunikuje pomoci jiného, o něco náročnějšího protokolu IMAP.

### 4.2.4 Konfigurační daemon

Konfigurační daemon umožňuje vzdálenou konfiguraci Mailproxy pomocí konfiguračních emailů. Daemon v nastavitelných intervalech kontroluje zadanou poštovní schránku, zda neobsahuje emaily s konfigurací, podepsané některým z autorizovaných administrátorů, v případě, že je tomu tak, email stáhne a adekvátně podle něj upraví konfiguraci Mailproxy.

### 4.2.5 Keymgr

Mailproxy dále obsahuje konzolovou utilitu `keymgr` pro generování a správu certifikátů ve formátu X.509,které jsou v Mailproxy primárně používány pro šifrování, dešifrování a podepisování elektronické pošty.

[1] http://www.cs.auckland.ac.nz/~pgut001/cryptlib/

**Použití utility keymgr**

Keymgr lze volat s několika různými parametry:

`--create label [file]` generuje nový certifikát s názvem *label* do souboru *file*, defaultně do souboru s názvem *keyset.p15*.

`--remove label file` odstraní certifikát *label* ze souboru *file*.

`--copy label file1 [file2]` kopíruje certifikát *label* ze souboru *file1* do souboru *file2*, případně z defaultního souboru do *file1*.

`--move label file1 [file2]` přesune certifikát *label* ze souboru *file1* do souboru *file2*, případně z defaultního souboru do *file1*.

`--copy label file1 [file2]` kopíruje certifikát *label* ze souboru *file1* do souboru *file2*, případně z defaultního souboru do *file1*.

`--export label [file1 [file2]]` exportuje veřejný klíč pro certifikát *label* ze souboru *file1* do souboru *file2*, případně z defaultního souboru do souboru `label`.

`--importpgp label [file] [secring pubring]` importuje veřejný a privátní klíč s názvem *label* ve formátu PGP do souboru *file*

## 4.3 Konfigurace Mailproxy

Konfigurace Mailproxy je uložena v XML souboru `settings.xml` umístěném v kořenovém adresáři aplikace. Konfigurace je velice komplexní a díky konfiguračních politik a priorit umožňuje ošetření takřka jakéhokoliv požadavku na nastavení chování proxy. Lze zde konfigurovat jak budou zasílány a přijímány emaily konkrétního uživatele konkrétnímu uživateli, uživatele do domény, domény do domény, kohokoliv konkrétnímu uživateli nebo doméně a všechny ostatní kombinace uživatelů, domén a obecných uživatelů a obecných domén. Mimoto se zde pak samozřejmě také nastavují porty, ne kterých poběží jednotlivé servery a jejich chování.

Samotnou konfiguraci lze provádět dvěma způsoby:

- Přímou editací konfiguračního souboru `settings.xml`
- Zasíláním emailů konfiguračnímu daemonovi

V následujících sekcích si oba způsoby blíže popíšeme.

### 4.3.1 Editace souboru `settings.xml`

Přímá editace konfiguračního souboru je asi nejefektivnějším způsobem konfigurace a nabízí stoprocentní kontrolu nad celým systémem Mailproxy. Konfigurační soubor `settings.xml` obsahuje následující kořenové uzly: `<users>`, `<domains>`, `<proxy_setting>` a `<public_keys>`.

#### uzel <users>

Tento uzel obsahuje jednotlivé uzly <user_setting>, ve kterých je možno v dalších poduzlech nastavit jednotlivé uživatele Mailproxy. K nastavení zde existují položky jako skutečné a přihlašovací jméno uživatele, emailová adresa a adresy s porty uživatelova skutečného SMTP, POP3 a IMAP serveru. Další položkou je uzel <policy>, který definuje výchozí politiku daného uživatele, která zahrnuje algoritmus pro šifrování pošty, specifikaci, zda se má pošta šifrovat a/nebo podepisovat, případně jestli se má nejprve podepsat nebo zašifrovat. Nakonec je pak ještě definováno, jak se má Mailproxy zachovat v případě, že se některý z úkonů podepisování/šifrování nezdaří. Kromě výchozí politiky je pak možno definovat neomezené množství dalších politik, které se můžou vztahovat jen ke konkrétním adresátům a/nebo doménám adresátů uživatelových emailů.

Nakonec je zde ještě poduzel <keys>, ve kterém je uveden nejdříve výchozí uživatelův veřejný klíč a odkaz do souboru s jeho privátním klíčem. Stejně jako u politik, je zde poté možno definovat jiné klíče pro jednotlivé adresáty nebo celé domény.

#### uzel <domains>

Uzel domény je svou strukturou velice podobný předešlému uzlu, ale jednotlivé konfigurace v poduzlech <domain_setting> se nevztahují k jednotlivým uživatelům, ale k celým doménám.

#### uzel <proxy_setting>

Tento uzel obsahuje konfiguraci samotné Mailproxy, nastavuje se zde například název, elektronická adresa Mailproxy a porty jednotlivých serverů. Dále jsou zde uvedeni administrátoři, kteří jsou oprávněni měnit konfiguraci pomocí konfiguračního daemona a nastavuje se zde, zda má daemon pro vzdálenou konfiguraci běžet a v jakých intervalech má kontrolovat svou emailovou schránku.

Pomocí poduzlů <log> je zde možno nastavit logování různých událostí do souboru a/nebo tyto informace zasílat elektronickou poštou.

Sekce <proxy_policy> potom nastavuje, podobě jako <policy> v uzlu s jednotlivými uživateli, politiky s nejvyšší prioritou, je zde možno vytvářet adresáře kombinované z elektronických adres a domén, u kterých je dále definován směr, ve kterém se daná politika aplikuje, tedy zda při odesílání daným uživatelem, přijímání, či vždycky.

Mailproxy kvůli komunikaci s uživateli rovněž potřebuje vlastní šifrovací klíč, a ten je definován na konci tohoto uzlu.

#### uzel <public_keys>

Uzel <public_keys>, jak již název napovídá, je uzel obsahující veřejné klíče, a to osob a domén, kteří nejsou uživateli místní Mailproxy a nemají zde své privátní klíče. Tyto klíče zde jsou kvůli šifrování odchozí pošty majitelům daných klíčů.

### 4.3.2 Konfigurace pomocí emailu

V případě, že je povolena vzdálená konfigurace a je definován alespoň jeden administrátor, bude možno Mailproxy konfigurovat i vzdáleně pomocí podepsaných konfiguračních emailů od některého z administrátorů. Kromě elektronického podpisu tyto konfigurační emaily

musí také obsahovat přesně specifikovanou XML strukturu, musí být šifrovány veřejným klíčem Mailproxy a v předmětu musí být vyplněno `CONFIG EMAIL`.

Tělo emailu potom může mít tři různé formáty podle toho, zda nějaké informace z konfigurace chci zjistit, vytvořit nebo změnit, použijeme tak příkazy `read`, `create` nebo `modify`. Pomocí emailu nelze konfigurovat úplně vše, co je možno provést editací XML souboru a nelze konfigurovat více než jednu položku najednou. Uzly, které je možno spravovat pomocí emailů, jsou `<user_setting>`, `<domain_setting>`, `<proxy_setting>`, `<user_keys>` a `<domain_keys>`.

#### Získání konfigurace

Pro získání částí konfigurace se používá párového tagu `<read>`, do kterého se uzavře specifikace položky, kterou chceme získat, například `<read><user>jr@ktknet.cz</user></read>`.

#### Vytvoření nové položky

Nová položka vytváří pomocí tagu `<create>`, ve kterém bude uzavřena kompletní definice daného uživatele, domény nebo klíče.

#### Změna konfigurace

Pro změnu konfigurace je zapotřebí znát původní konfiguraci, ideálně získanou pomoci příkazu `<read>` a zaslat ji serveru společně s novou konfigurací ve tvaru

```
<modify>
      <user>uživatel</user>
      <old>stará konfigurace</old>
      <new>nová konfigurace</new>
</modify>
```

Zasílání původní konfigurace se může zdát zbytečné, ale je to opatření pro situace, kdy se rozhodnou dva administrátoři ve stejnou chvíli editovat stejnou položku konfiguračního souboru. [11]

## 4.4 Knihovna Cryptlib

Cryptlib je open source multiplatformní bezpečnostní toolkit distribuovaný pod Sleepycat licencí,[2] která je kompatibilní s GNU General Public Licencí. Jeho autorem je Peter Gutmann, uznávaný odborník na počítačovou bezpečnost z Aucklandské univerzity na Novém Zélandu. Mezi jeho další známé výtvory patří například tzv. Gutmannova metoda pro bezpečné odstraňování souborů z pevných disků.

### 4.4.1 Trojvrstvá architektura aplikačního rozhraní

Knihovna Cryptlib sestává z několika vrstev bezpečnostních služeb a sdružených programových rozhraní, které poskytují integrovanou sadu informačních a komunikačních funkcí. Jednotlivé vrstvy v knihovně Cryptlib, podobně jako vrstvy v referenčním síťovém modelu,

[2] plné znění licence na http://opensource.org/licenses/sleepycat.php

poskytují různé úrovně abstrakce s vyššími vrstvami těžícími ze služeb poskytovaných vrstvami nižšími. S možností libovolně využívat funkcí a služeb na kterékoliv z těchto vrstev.

Na nejnižší vrstvě jsou základní komponenty jako šifrovací jádro, autentizační rutiny a rozhraní pro bezpečnostní nástroje třetích stran, které jsou obvykle implementovány softwarově, ale mohou být implementovány rovněž hardwarově, Cryptlib umožňuje využití velkého množství externích zařízení jako jsou hardwarové krypto-akcelerátory, bezpečnostní karty, USB klíče a další.

Na následující vrstvě jsou komponenty, které obalují specializované a často poměrně komplexní nízkoúrovňové funkce pro další abstraktnější funkce, které tímto mj. poskytují dokonalou meziplatformní přenositelnost. Tyto funkce typicky pokrývají činnosti jako „vytvoř digitální podpis“ nebo „přenes šifrovací klíč“.

Na nejvyšší vrstvě pak již jsou funkce jako „zašifruj zprávu“, „podepiš zprávu“, „navaž zabezpečené spojení“ nebo „vytvoř digitální certifikát“, které již nepotřebují téměř žádné podrobné znalosti kryptografických metod a přesto dovolují vytváření vysoce zabezpečených aplikací postavených na tomto toolkitu.

Knihovna obsahuje výkonné rozhraní pro správu jednotlivých svých objektů a poskytuje tak možnost přidání podpory šifrování a autentizace do aplikací bez potřeby znát všechny detaily těchto procesů. Tímto Cryptlib nabízí značnou výhodu oproti jiným šifrovacím toolkitům, které často vyžadují velké množství kódu a manipulaci s rozsáhlými datovými strukturami k docílení stejných výsledků. [4]

### 4.4.2 Podpora platforem

Knihovna Cryptlib je napsána v jazyce C a je tedy vysoce přenositelná, v současné době podporuje operační systémy BeOS, DOS, IBM MVS, Macintosh/OS X, OS/2, Tandem, Unixové systémy (zahrnuje AIX, Digital Unix, DGUX, FreeBSD/NetBSD/OpenBSD, HP-UX, IRIX, Linux, MP-RAS, OSF/1, QNX, SCO/UnixWare, Solaris, SunOS, Ultrix a UTS4), VM/CMS, Windows 3.x, Windows 95/98/ME, Windows CE/PocketPC/SmartPhone a Windows NT/2000/XP.

Aplikační rozhraní Cryptlibu je dostupné pro jazyky C, C++, C#, Delphi, Java, Python, Visual Basic a také platformu .NET.

Přes tento velice rozsáhlý seznam použitelných platforem ovšem není knihovna zcela přenositelná na systémy založené na uClibc (viz. sekce 5.1.2), což by se ale v příštím vydání knihovny mělo změnit.

# Kapitola 5

# Portace Mailproxy na WRAP 2C

V této kapitole budou nejdříve popsány rozdíly mezi knihovnami používanými systémy na osobních počítačích a embedded zařízeních. Dále se pak budu věnovat výběru vhodného operačního systému pro portaci a nakonec samotné portaci Mailproxy na desku WRAP 2C.

## 5.1 GNU C Library vs. uClibc

Základním problémem při portaci Mailproxy byly rozdíly v systémových knihovnách jazyka C. Mailproxy byla napsána pro GNU C Library, zatímco na naprosté většině embedded zařízení se z důvodů omezené kapacity paměti a/nebo absenci MMU[1] používá knihovna uClibc. Obě tyto knihovny jsou standardní knihovny jazyka C, které sestávají ze množiny hlavičkových souborů a knihovních rutin používaných k implementaci běžných činností jako jsou vstupně-výstupní operace a zpracování řetězců v jazyce C.

### 5.1.1 glibc

Knihovna GNU C Library, obvykle nazývána zkráceně **glibc**, je standardní knihovna jazyka C vydaná projektem GNU. Původně byla napsána Free Software Foundation (FSF) pro GNU operační systém. Knihovna glibc je uvolněna pod GNU Lesser General Public Licencí jako svobodný software.

Původně byla knihovna glibc napsána z velké části Rolandem McGrathem v 80-tých letech. V 90-tých letech se vývoj glibc rozdělil na dvě větve a vývojáři Linuxového jádra pracovali na vlastní verzi pod názvem Linux libc. Když FSF v roce 1996 vydalo glibc ve verzi 2.0, mělo mnohem lepší podporu POSIX standardů, lepší multijazyčnou podporu, podporu pro IPv6, podporu 64-bitového adresování paměti, přenositelnější kód a mnoho dalších výhod, proto bylo od dalšího vývoje Linux libc upuštěno a začala se znovu používat mnohem dokonalejší glibc.

Glibc je dnes používána na velkém množství systémů s různými jádry a na různých hardwarových architekturách, poskytuje funkce podle standardu Single UNIX Specification a POSIX 1c, 1d a 1j, dále ISO C99, BSD interfaces, System V Interface Description a X/Open Portability Guide. Knihovna je často kritizována za svou velkou mohutnost a z toho důvodu začaly vznikat alternativní knihovny především s důrazem na malou velikost kódu. Patří mezi ně již zmiňovaná knihovna uClibc nebo dále například dietlibc, Newlib, Klibc a EGLIBC.[3]

[1] Memory Management Unit

### 5.1.2 uClibc

uClibc je implementace standardní knihovny jazyka C určená pro vestavěná zařízení s operačními systémy založenými na Linuxovém jádře s důrazem na co nejmenší velikost kódu. Malé „u“ v názvu je přepisem řeckého znaku $\mu$ (mikro) a často se takto i čte, velké „C“, které následuje, je potom zkratkou pro Controller, název tedy lze volně přeložit jako „knihovna jazyka C pro mikro-kontroléry“.

Vedoucím vývojářem projektu je Erik Andersen, který je rovněž autorem projektu BusyBox, malé aplikace, kombinující velké množství základních linuxových utilit do jednoho spustitelného souboru, která je rovněž hojně využívána na embedded zařízeních. Knihovna uClibc je stejně jako glibc uvolněna pod GNU Lesser General Public License jako svobodný software, vývoj knihovny začal okolo roku 1999.

Rozsahem kódu je uClibc mnohem menší, než glibc. Zatímco glibc podporuje všechny standardy jazyka C na velkém množství hardware a platforem, uClibc je určena především pro podporu vestavěných zařízení. Knihovna uClibc je použitelná na procesorech řady i386, x86 64, ARM, AVR32, Blackfin, h8300, m68k, MIPS, PowerPC, SuperH, SPARC a v850. [1]

## 5.2 Linuxové distribuce pro embedded systémy

Poté, co jsem zvolil WRAP 2C jako embedded zařízení pro portaci bezpečné proxy, bylo třeba zvolit vhodný operační systém, na kterém proxy poběží. Do analýzy jsem zařadil několik linuxových distribucí upravených pro použití na embedded zařízeních, zaměřoval jsem se pouze na distribuce, které jsou k dispozici zadarmo, proto zde nejsou uvedeny komerční distribuce jako například StarOS.

### 5.2.1 Voyage Linux

Voyage Linux[2] je distribuce, která je standardně dodávána s deskami WRAP, je odvozena od distribuce Debian Etch r4.0 a redukována na velikost okolo 64MB. Jelikož jde o distribuci přímo vyvíjenou pro provoz na bezdrátových směrovačích, jsou zde obsaženy ovladače pro všechny dostupné miniPCI Wi-Fi karty, podporuje WPA[3] a také watchdog procesoru SC1100, také je zde podpora pro čtení teploty z čidla ML77. Systém je rozšiřitelný pomocí Apt balíčkového manageru, ovšem ne všechny Debian balíčky je možno nainstalovat z důvodů hardwarové nekompatibility s WRAP deskou.

### 5.2.2 OpenWRT

OpenWRT[4] je distribuce vyvinutá původně pro řadu bezdrátových routerů Linksys WRT54G a také je optimalizovaná pro využití k routování paketů. Distribuce je více uživatelskou distribucí, než ostatní, což také naznačuje přítomnost webového GUI administračního rozhraní, to je dále rozšiřitelné projektem X-Wrt, který přináší spoustu nových modulů, především pro získávání statistik o provozu na síti.

Distribuce OpenWRT přináší tzv. univerzální konfigurační rozhraní (UCI), které spojuje konfiguraci všech součástí systému do jednotného formátu se společenou strukturou `config.section.key=value`, což zajisté zjednoduší práci lidem, kteří nejsou zvyklí na OS

[2] http://linux.voyage.hk/
[3] Wi-Fi Protected Access
[4] http://openwrt.org/

Linux, ovšem naopak zřejmě trochu zmate linuxové administrátory, kteří budou konkrétní konfigurační soubory hledat na jiných místech.

OpenWRT obsahuje jednoduchý, ale pro účely distribuce plně dostačující balíčkový manager `ipkg` a distribuce i jednotlivé balíčky se neustále vyvíjí s možností stahování aktuálních verzí z svn. Celý systém je postaven na BusyBoxu a uClibc, viz 5.1.2.

### 5.2.3 Linuxová distribuce K240

K240[5] je linuxová distribuce pro embedded zařízení (především WRAP) založena na distribuci Gentoo, kompilovaná s knihovnou uClibc. Tato distribuce byla vytvořena Liborem Valentou jako operační sytém pro levné bezdrátové routery v síti Humlnet.

Standardně K240 obsahuje služby především pro podporu směrování paketů a typické služby užitečné na routerech. Jsou jimi například DHCP server a DHCP-relay, firewall iptables, OpenVPN, démoni pro směrovací protokoly ospfd a ripd, démon pro analýzu sítě snmpd, zabezpečený shell sshd nebo služba rdate pro nastavení času, všechny tyto služby jsou kompilovány pro protokoly jak IPv4, tak také IPv6. Kromě těchto služeb je zde potom například minimalistický http server monkeyd s podporou PHP5 nebo gphoto ovladače pro fotoaparáty a webkamery.

Systém souborů distribuce K240 je poměrně složitý a připojuje několik virtuálních disků, ze samotné CF karty nelze přečíst jinak, než po nabootování do systému na embedded zařízení, což je veliká výhoda z hlediska bezpečnosti, protože na CF kartě budou muset být uloženy privátní klíče uživatelů. Drobnou nevýhodou je ovšem použití SquashFS filesystému, který je read-only, pro oddíly mountované do `/bin/`, `/boot/`, `/home/`, `/lib/`, `/opt/`, `/sbin/`, `/sys/` a `/usr/` a tedy zdánlivá nemožnost přidání software, který není v distribuci přímo obsažen.

## 5.3 Srovnání distribucí

Stejně jako při výběru vhodné desky v předešlé kapitole, je v tabulce 5.1 seznam požadavků, které hrály hlavní roli při výběru operačního systému a porovnání operačních systémů z hlediska splnění těchto požadavků. Parametry, kterými je požadavek nejlépe splněn, jsou opět vyznačeny tučným písmem.

| **Můj požadavek** | **Voyage Linux** | **OpenWRT** | **K240** |
|---|---|---|---|
| Vlastní balíčkový systém | **ANO (apt)** | **ANO (ipkg)** | NE |
| Read + write filesystém | **ANO** | částečně | částečně |
| Uživatelská přívětivost (1-3) | ** | ** | *** |
| Překladač jazyka C | NE | NE | **ANO, gcc 3.4.6** |
| Editor VIM | NE | NE | **ANO** |
| Interpret jazyka Python | NE | NE | **ANO** |
| HW podpora WRAP 2C (1-3) | *** | * | ** |

Tabulka 5.1: Srovnání vybraných operačních systémů

Ne všechny požadavky na operační systém jsou zde asi zcela objektivní a nezbytné (například editor vim) a také mají různou váhu. Požadavek „Uživatelská přívětivost“ je můj zcela subjektivní dojem z krátkého používání systému, hodnocený jednou až třemi

[5] http://k240.humlak.cz

hvězdičkami. Požadavek na balíčkový systém byl nakonec shledán jako přeceněný, protože přes přítomnost balíčkového systému nebyla možnost (alespoň z oficiálních repozitářů) získat potřebný software, který na systém K240 - jako jediný bez balíčkového systému - dostat nebylo až tak velkým problémem. U hardwarové podpory vycházím především z informací a diskuzí na webu, Distribuci OpenWRT jsem na WRAP 2C neinstaloval, ale chvíli jsem ji provozoval na routeru Asus WL-500, oficiálně deska WRAP ani není podporována, upravené verze by ovšem měly být použitelné, což ovšem nic nemění na vítězství distribuce K240, která v mém srovnávacím testu, sice nesplnila všechny důležité požadavky, je ale poměrně dobře přizpůsobitelná a vychází z mé oblíbené distribuce - Gentoo Linuxu.

## 5.4 Samotná portace Mailproxy

V této sekci se konečně vrhneme na samotnou portaci Mailproxy, probereme zde některé problémy, se kterými jsem se v průběhu portace potýkal a přiblížíme si úpravy provedené na jednotlivých částech projektu.

### 5.4.1 Knihovna Cryptlib a uClibc

Knihovna Cryptlib verze 3.3.1 sice sama o sobě je s knihovnou uClibc přeložitelná, ovšem nefunguje zde bez drobných úprav úplně správně a nelze jí slinkovat s Mailproxy kvůli absenci některých systémových funkcí v uClibc.

Mezi chybějící funkce, které knihovna využívá, patří například funkce `futimes()` pro nastavení časů přístupu a modifikace i-uzlu, tato funkce lze s úspěchem nahradit funkcí `utimes()`, která je v uClibc implementována. Další problémovou funkcí, kterou Cryptlib používá, je funkce `thread_yield()`, kterou volající vlákno předává řízení plánovači procesů a vzdává se procesoru. Tato funkce je na uClibc systému nahraditelná podobnou funkcí `sched_yield()`. Funkcí, pro kterou v uClibc neexistuje adekvátní náhrada, je funkce `dn_skipname()` pro přeskakování po komprimovaných hlavičkách v DNS dotazech a odpovědích, absenci této funkce jsem ošetřil přidáním adekvátní části kódu ze zdrojových kódů knihoven glibc.

Veškeré své úpravy jsem po analýze systémem definovaných maker preprocesoru *cpp*[6] překladače *gcc* [7] začlenil formou podmíněného překladu do zdrojových kódu knihovny Cryptlib a vytvořil patch pro tuto knihovnu, který jsem zaslal autorovi Cryptlibu, Peterovi Gutmannovi z University of Auckland. Ten mé úpravy začlenil do svého kódu a budou součástí příští verze Cryptlibu.

Zmíněný patch, který jsem pro Cryptlib vytvořil, jsem umístil na svůj školní web na `http://www.stud.fit.vutbr.cz/ xricht14/ibp/cryptlib-uclibc.patch`.

### 5.4.2 Distribuce K240

Vybraná linuxová distribuce K240 mi sice ze srovnávaných distribucí vyhovovala nejlépe, ale přecejen nesplňovala všechny požadavky, které byly pro portaci Mailproxy na desku WRAP 2C zapotřebí, proto bylo třeba provést modifikace i zde.

[6] výstupy příkazu `cpp -dM empty.c`

#### NTP a timezone-data

Z důvodů použití časových známek a omezování platnosti bezpečnostních certifikátů na určité časové období, je nutností udržovat na systému s Mailproxy aktuální systémový čas. Distribuce sice obsahuje službu `rdate` pro synchronizaci času, nicméně tato služba dnes nemá příliš velkou podporu na straně time-serverů a používá se výrazně komplexnější služby `ntp`, která při synchronizaci zohledňuje také časová pásma a zpoždění synchronizačních paketů na síti. Službu `rdate` jsem proto nahradil službou `ntp` a doplnil databázi timezone-data pro podporu časových pásem, která je vyžadována `ntp` klientem.

#### Integrace Cryptlibu

Pro bezproblémový provoz Mailproxy na systému K24 bylo třeba do systému zaintegrovat již dříve probíranou knihovnu Cryptlib zkompilovanou přímo na zařízení oproti knihovnám uClibc. Toto bylo jednak velice časově náročné na poměrně slabý procesor a pomalý řadič CF karty a na původním systému nemožné díky read-only filesystému pro kořenovou složku `/usr/`.

#### Zapisovatelný oddíl pro nové aplikace

Nejen kvůli Cryptlibu (viz. předchozí odstavec) bylo třeba vytvořit nový zapisovatelný ext2 oddíl, který se mountuje přes původní read-only oddíl do /usr.

#### Zabezpečení fail2ban

Pro celkové zvýšení zabezpečení Mailproxy jsem do systému K240 přidal pythonového daemona `fail2ban`, který hlídá logy ssh daemona a po třech neúspěšných pokusech o přihlášení zakazuje dané IP adrese pomocí firewallu `iptables` přístup na dobu, kterou jsem ve výchozím nastavení stanovil na 12 hodin. Stejně jako pro ssh je služba dále konfigurovatelná pro jakýkoliv jiný protokol.

#### Systémové služby

Pro dokonalou integraci Mailproxy a ostatních přidaných služeb jsem pro ně vytvořil inicializační skripty v Gentoo formátu, aby bylo možno s nimi zacházet jako s každou jinou systémovou službou. Mail proxy je tedy možno spustit, zastavit, restartovat nebo zjistit zda běží pomocí příkazů

```
/etc/init.d/mailproxy [start|stop|restart|status]
```

### 5.4.3 Konfigurační utilita Proxysetup

Pro usnadnění konfigurace Mailproxy jsem naprogramoval jednoduchou konzolovou konfigurační utilitu v pythonu s využitím knihovny pycurses. Díky této utility je možno pohodlněji spravovat uživatelské účty, klíče, skupiny a jednotlivé politiky v konfiguračním xml souboru aplikace Mailproxy.

Utilita `proxysetup` očekává v pracovním adresáři nebo na cestě zadané jako parametr, konfigurační xml soubor Mailproxy, který zpracuje a v jednoduchém textovém GUI je poté možné konfiguraci Mailproxy procházet a upravovat přidáváním a mazáním uzlů metodou procházení jednotlivých menu s hierarchickou strukturou podobnou struktuře souboru `settings.xml`.

Utilita `proxysetup` poskytuje intuitivní ovládání, v základním menu aplikace obsahuje následující položky:

**nastavení proxy** zde je možno nakonfigurovat porty jednotlivých serverů, název a elektronickou adresu Mailproxy stejně jako přihlašovací informace ke konfigurační schránce, interval mezi kontrolami této schránky, seznam administrátorů oprávněných měnit vzdáleně konfiguraci, veřejný a privátní klíč Mailproxy a politiky, které budou použity jako tzv. fall-back politiky v případě nenalezení vhodné politiky na vyšších úrovních konfigurace.

**nastavení uživatelů** poskytuje možnost přidávání, odebírání a konfigurace jednotlivých uživatelů Mailproxy, nastavují se zde základní údaje jako jméno, elektronická adresa, adresy SMTP, POP3 a IMAP serverů, výchozí politika a politiky pro jednotlivé uživatele a domény.

Například je zde možno nastavit zda se má odchozí pošta pro vybraného uživatele podepisovat, šifrovat, co z těchto úkonů se má provést dříve a jak se má Mailproxy zachovat v případě, že se z nějakého důvodu podesání a/nebo zašifrování nezdaří.

**nastavení domén** je takřka shodné s nastavením uživatelů, ovšem jednotlivé nastavení se aplikují na celé domény namísto konkrétních uživatelů.

**veřejné klíče uživatelů** umožňuje přidávání, odebírání a správu veřejných klíčů osob, které nemají v systému veden svůj privátní klíč, ale bývají například adresáty uživatelů Mailproxy a proto je potřeba jejich klíče uchovávat.

**veřejné klíče domén** poskytuje správu veřejných klíčů pro celé domény.

Utilita zjednodušuje konfiguraci především uživatelům neseznámeným se strukturou konfiguračního souboru a poskytuje rychlejší, pohodlnější a také komplexnější způsob konfigurace, než stávající metoda vzdálené konfigurace pomocí emailu (4.3.2), která je obsažena v Mailproxy.

# Kapitola 6

# Využití

V této kapitole si shrneme nejpodstatnější body, ve kterých měla tato práce přínos a ukážeme si několik možností zapojení desky v síti.

## 6.1 Globální konfigurace

Mailproxy na desce WRAP 2C je určena k nasazení především ve firmách, kde je zapotřebí globálně zabezpečit emailovou komunikaci a nelze spoléhat, že vše zvládnou uživatelé sami. Konfiguraci tedy bude provádět síťový administrátor, kterému se použitím Mailproxy zjednoduší a zautomatizuje velké množství práce, při které by musel normálně obcházet všechny stanice v síti a ručně generovat a konfigurovat uživatelské klíče. Nastavení proxy jako SMTP a POP3 klienta pro jednotlivé stanice lze dle RFC-2132[1] provést pomocí DHCP protokolu, otázkou ovšem zůstává, zda dnes existuje nějaký klient, který je pomocí DHCP konfigurovatelný, doposud se mi nepodařilo takového nalézt.

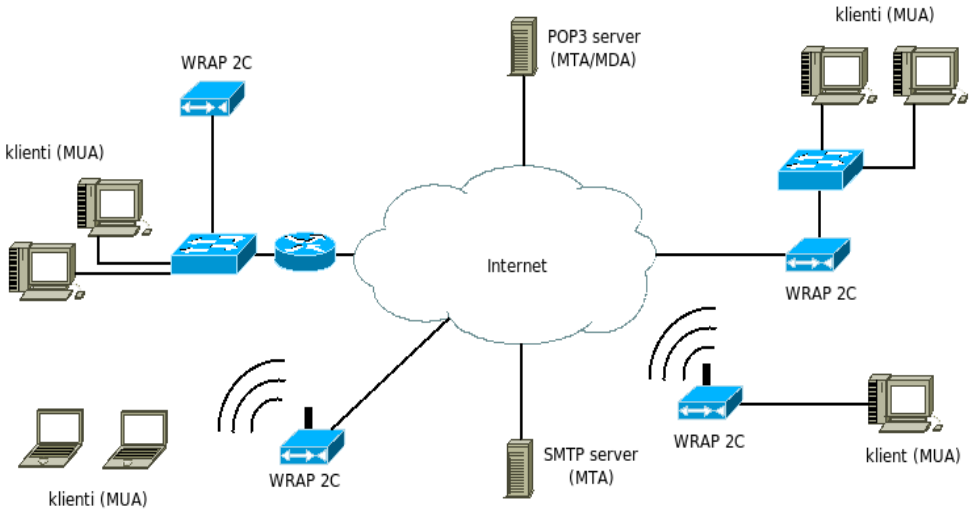
## 6.2 Minimalizace nákladů na provoz

Přínosem portace na embedded zařízení je především minimalizace nákladů na pořízení hardware a na jeho provoz. Provoz mailproxy na vyhrazeném serveru by byl zbytečně drahý a rovněž i pořízení celého serveru by bylo nesrovnatelně dražší a zbytečné.

## 6.3 Zvýšení bezpečnosti Mailproxy

Další výhodou je bezpečnost, zařízení bude určeno k provozu čistě jako Mailproxy, případně ještě jako směrovač a nebude obsahovat další software a uživatelské účty, které by mohly být zneužity k získání přístupu k citlivým údajům. Potenciálním bezpečnostním rizikem mailproxy je získání konfiguračního souboru a souborů s privátními klíči uživatelů, toto riziko je portací na embedded zařízení rovněž sníženo možností celý systém díky jeho velikosti snadněji někde fyzicky uzamknout.

[1] DHCP Options and BOOTP Vendor Extensions, http://tools.ietf.org/html/rfc2132




Obrázek 6.1: Ukázkové zapojední desek WRAP 2C s Mailproxy

## 6.4 Možnosti připojení

Nakonec si ještě představíme několik typických možností zapojení Mailproxy do sítě. Některé z nich jsou naznačeny na obrázku 6.1.

Naše deska může být připojena do vnitrofiremní sítě pomocí Ethernetu jako koncové zařízení, na které budou směrovány požadavky na poštovní servery a ty bude Mailproxy po zpracování přes firemní bránu zasílat do internetu správným serverům (na obrázku vlevo nahoře). Další možností připojení je využití desky společně s Mailproxy také ve funkci přístupového bodu do internetu a firewallu s překladem adres a to třeba jako ethernetový směrovač (obrázek vpravo nahoře), případně může WRAP 2C připojit vnitřní síť k internetu bezdrátově (na obrázku vpravo dole). Další, ale v současném stádiu vývoje nedoporučenou možností (viz. sekce 7.4) je využití desky také jako bezdrátového přístupového bodu pro klienty (na obrázku situace vlevo dole).

Ve všech případech pak Mailproxy komunikuje se SMTP, POP3 a/nebo IMAP servery v internetu.

# Kapitola 7

# Další vývoj

Celý návrh Mailproxy je velice komplexní a rozsáhlý systém, prozatím z něj bylo vytvořeno jádro aplikace, se základními funkcemi a jeho portace na embedded zařízení, desku WRAP 2C, společně s přípravou operačního systému a utilit pro bezpečný provoz. Při analýze aplikace a práci na portaci jsem odhalil různé nedostatky a přišel na několik dalších návrhů na další vývoj na projektu.

## 7.1 Migrace na desku ALIX

V průběhu vývoje vyšla informace, že deska WRAP 2C se přestala vyrábět kvůli ukončení výroby procesoru National Geode SC1100 firmou AMD, proto jedním z dalších kroků by měla být migrace na desku ALIX[1], která je - dle informací na webu firmy PC Engines GmbH - kompatibilním nástupcem desky WRAP a proto by tento krok neměl činit žádný větší problém.

## 7.2 Aplikace pro pohodlnou správu

Dalším krokem ve vývoji by mohla být implementace aplikace s GUI, která by se spouštěla na počítači administrátora a pomocí speciálního zabezpečeného protokolu by komunikovala s Mailproxy a umožňovala takto její pohodlnou konfiguraci a správu účtů, výpis a filtrování logů, případně také vykreslování grafů zatížení Mailproxy. Inspirací by měla být aplikace WinBox pro správu routerboardů, dodávaná se systémem MikroTik RouterOS.

## 7.3 Distribuce veřejných klíčů

Úpravou, která by zajisté velice usnadnila práci administrátorům, je vytvoření zabezpečeného protokolu, pomocí kterého by si jednotlivé vzájemně autorizované Mailproxy v síti mohly vyměňovat veřejné klíče uživatelů, inspirací by mohl být některý ze standardních směrovacích protokolů.

[1] http://www.pcengines.ch/alix.htm

## 7.4 Zabezpečení komunikace s mailproxy

Poměrně důležitou částí zabezpečení Mailproxy je bezpečná komunikace mezi klientem a proxy, což není při komunikaci po sdíleném médiu, jako Ethernet nebo dokonce Wi-Fi, snadno zajistitelné. V případě Ethernetu a použití switchů na druhé vrstvě je komunikace poměrně bezpečná, dokud útočník neužije tzv. Arp-poisoning metody, kdy zahltí vyrovnávací paměť přepínače a ten se pak začne chovat jako hub, případně podvrhne ARP záznamy tak, aby byla komunikace směrována přes něj. Bránit se tomu lze částečně použitím kvalitních switchů, které podporují vytváření Virtuálních LAN (VLAN) sítí, což je umožňují například produkty firmy Cisco.

V případě bezdrátových sítí Wi-Fi je riziko odposlechu komunikace s proxy značné a v současném stavu vývoje použití na bezdrátových sítích silně nedoporučuji.

Řešením této situace by byla implementace zabezpečení komunikace s Mailproxy na transportní vrstvě pomocí protokolů SSL/TLS.

## 7.5 Spamový a virový filtr

Obrovským problémem elektronické pošty je v dnešní době šíření nevyžádané pošty a virů, existuje spousta softwaru, který se snaží se spamem a viry v elektronické poště bojovat, ovšem mají určité nevýhody, pokud se tento software nachází na poštovním serveru, není schopen kontrolovat šifrované zprávy, pokud je naopak na jednotlivých stanicích uživatelů, přibývá spousta práce síťovému administrátorovi, který se o stanice stará. Mailproxy ve svém principu nabízí řešení, kdy stačí spamový filtr a antivir konfigurovat jen na jednom místě a přitom může být účinný i na zašifrované zprávy.

Pro operační systém Linux existuje nepřeberné množství daemonů pro boj se spamem a viry v elektronické poště, jedním z dalších kroků na vývoji Mailproxy by proto mohla být integrace jednoho z těchto systémů.

# Kapitola 8

# Závěr

Ve své bakalářské práci jsem se věnoval problematice zabezpečení elektronické pošty a metodám a historii kryptografie. Testoval jsem dostupná embedded zařízení a operační systémy vhodné k použití na těchto zařízeních, provedl jsem jejich analýzu a vybraný systém přizpůsobil k použití jako emailovou proxy.

Jako stávající implementaci proxy pro zabezpečení elektronické pošty jsem si vybral projekt Mailproxy (4) Bc. Stanislava Židka a svou prací učinil další krok ve vývoji tohoto pozoruhodného projektu. Mailproxy je možno po mém zásahu provozovat na embedded desce WRAP 2C, která má téměř zanedbatelnou spotřebu elektrické energie a minimalizují se tak náklady na pořízení, provoz a údržbu, desku je taky možno díky malým rozměrům snadno fyzicky uzamknout a tím zvýšit zabezpečení uživatelských dat, uložených na CF kartě.

Další pokrok byl učiněn implementací jednoduché konzolové curses aplikace pro snadnější konfiguraci systému a integrací Mailproxy do systému K240 jako systémové služby. Systém K240 jsem dále doplnil několika dalšími bezpečnostními mechanismy a celkově jsem se snažil systém nakonfigurovat s ohledem na maximální zabezpečení uživatelských dat.

Vedlejším produktem mé práce byl vznik patche pro Cryptlib, který dále rozšiřuje přenositelnost této knihovny na systémy založené na uClibc, viz. kapitola 5.4.1.

# Dodatek A

# Obraz CF karty se systémem a Mailproxy

Na přiloženém DVD je obraz celého systému, připravený k nahrání na 1GB CF kartu pro zařízení WRAP 2C. Nahrání je možno provést například pomocí příkazu

```
dd if=k240_mailproxy.iso of=/dev/sdx
```

kde *sdx* je CF karta.

# Dodatek B

# uClibc patch pro cryptlib

Patch, který jsem napsal pro knihovnu Cryptlib (5.4.1) je rovněž na přiloženém médiu, a to v adresáři `uclibc_cryptlib_patch`.

# Literatura


[1] Erik Andersen. Uclibc frequently asked questions. http://www.uclibc.org/FAQ.html, naposledy navštíveno 29. 4. 2008.

[2] Dave Crocker. Email history, how email was invented. http://www.livinginternet.com/e/ei.htm, Naposledy navštíveno 13. 4. 2008.

[3] Free Software Foundation, Inc. Gnu c library. http://www.gnu.org/software/libc/libc.html, Naposledy navštíveno 29.4.2008.

[4] Peter Gutmann. *Cryptographic Security Architecture*. Springer-Verlag New York, Inc., 2002. ISBN 0-387-95387-6.

[5] Petr Laščák. Linux na platformě wrap. http://www.abclinuxu.cz/clanky/hardware/linux-na-platforme-wrap-1, 2005.

[6] PC Engines GmbH. Pc engines wireless router application platform. http://www.pcengines.ch/wrap.htm, Naposledy navštíveno 29. 4. 2008.

[7] Red Hat, Inc. Using cpp, the c preprocessor. http://www.redhat.com/docs/manuals/enterprise/RHEL-3-Manual/cpp/predefined-macros.html, Naposledy navštíveno 13. 4. 2008.

[8] Shafi Goldwasser, Mihir Bellare. *Lecture Notes on Cryptography*. 2001. studijní materál na Massachusetts Institute of Technology http://www-cse.ucsd.edu/~mihir/papers/gb.html.

[9] MikroTikls SIA. Routerboard 111/112 series user's manual. http://www.34t.com/PDF/rb110ugA.pdf, 2006.

[10] Wikimedia Foundation, Inc. Cryptography. http://en.wikipedia.org/wiki/Cryptography, naposledy navštíveno 29. 4. 2008.

[11] Bc. Stanislav Židek. *Konfigurovatelná proxy pro zabezpečení elektronické pošty, bakalářská práce*. Brno, FIT VUT v Brně, 2007.