VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

KONFIGUROVATELNÁ PROXY PRO ZABEZPEČENÍ ELEKTRONICKÉ POŠTY

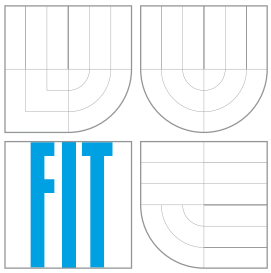
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

STANISLAV ŽIDEK

BRNO 2007

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

KONFIGUROVATELNÁ PROXY PRO ZABEZPEČENÍ ELEKTRONICKÉ POŠTY

CONFIGURABLE SECURITY EMAIL PROXY

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

STANISLAV ŽIDEK

VEDOUCÍ PRÁCE
SUPERVISOR

doc. Ing. DANIEL CVRČEK, Ph.D.

BRNO 2007

Zadání

Konfigurovatelná proxy pro zabezpečení elektronické pošty

1. Proveďte analýzu současných implementací systémů pro zabezpečení elektronické pošty a identifikujte slabá místa z hlediska použitelnosti a bezpečnosti.
2. Seznamte se s předchozími projekty implementujícími bezpečnou email proxy a identifikujte oblasti, které nejsou řešeny.
3. Upravte implementaci existující proxy tak, aby bylo možné je spolehlivě používat.
4. Navrhněte systém pro vzdálenou konfiguraci proxy, příp. sítě těchto proxy.
5. Implementujte systém pro vzdálenou konfiguraci proxy.

Kategorie: Bezpečnost

Implementační jazyk: C, C++

Literatura: Gutmann: Cryptographic Security Architecture, Springer, 2002

Datum zadání: 1. listopadu 2006

Datum odevzdání: 15. května 2007

Licenční smlouva

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

Abstrakt

Tato bakalářská práce se zabývá návrhem a implementací víceuživatelského centrálně konfigurovatelného systému pro zabepečení komunikace elektronickou poštou. Velký důraz je kladen na možnost vzdálené konfigurace. Popisuje současně používané protokoly pro komunikaci elektronickou poštou a některé mechanismy sloužící k zabezpečení různých aspektů této komunikace.

Klíčová slova

elektronická pošta, zabezpečení, vzdálená konfigurace, proxy, SMTP, POP3, IMAP, cryptlib, XML, S/MIME

Abstract

This bachelor thesis deals with design and implementation of multiuser configurable system capable of securing communication via electronic mail. Emphasis is put especially on remote configuration. It also describes existing protocols used for this communication and some of the mechanisms that can secure this communication from different points of view.

Keywords

electronic mail, security, remote configuration, proxy, SMTP, POP3, IMAP, cryptlib, XML, S/MIME

Citace

Stanislav Židek: Konfigurovatelná proxy pro zabezpečení elektronické pošty, bakalářská práce, Brno, FIT VUT v Brně, 2007

Konfigurovatelná proxy pro zabezpečení elektronické pošty

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Daniela Cvrčka. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Stanislav Židek
14. května 2007

Poděkování

Rád bych poděkoval mému vedoucímu, Danielu Cvrčkovi, za dlouhodobou podporu, trpělivost a neocenitelné rady poskytnuté při vývoji systému a psaní bakalářské práce. Rovněž bych rád poděkoval Jakubovi Kubalíkovi, který se do vývoje zapojil implementací IMAP serveru a jehož práce měla význam i pro zbytek systému.

Obsah

1 Úvod **3**

2 Elektronická pošta **5**
2.1 Historie . . . 5
2.2 Princip . . . 5
2.3 Formát zprávy . . . 6
2.4 Používané protokoly . . . 7
2.4.1 SMTP – Simple Mail Transfer Protocol . . . 7
2.4.2 POP3 – Post Office Protocol (version 3) . . . 9
2.4.3 IMAP – Internet Mail Access Protocol . . . 12

3 Zabezpečení elektronické pošty **14**
3.1 Zabezpečení přihlašování k serveru . . . 14
3.1.1 Příkaz AUTH(ENTICATE) . . . 14
3.1.2 Příkaz APOP protokolu POP3 . . . 15
3.2 Zabezpečení celého průběhu komunikace . . . 15
3.2.1 Zabezpečení na úrovni transportní vrstvy, TLS/SSL . . . 15
3.3 Zabezpečení obsahu emailových zpráv . . . 16
3.3.1 S/MIME – Secure / Multipurpose Internet Mail Extensions . . . 17
3.3.2 PGP – Pretty Good Privacy . . . 18

4 Návrh **19**
4.1 Celková filosofie . . . 19
4.1.1 Výhody . . . 19
4.1.2 Nevýhody . . . 20
4.2 Cílové prostředí . . . 20
4.3 Rozšiřitelnost a modularita . . . 20
4.4 Bezpečnostní politiky . . . 21

5 Implementace **22**
5.1 Použité technologie . . . 22
5.2 Struktura . . . 22
5.2.1 Server . . . 23
5.2.2 Databáze nastavení (`xmlhandle`) . . . 24
5.2.3 Kryptografie (`cryptography`) . . . 25
5.2.4 Zpracování emailu (`mailhandle`) . . . 26
5.2.5 Síťový podsystém (`network`) . . . 26
5.2.6 Logování (`logger`) . . . 26

5.2.7 Zpracování chyb (`error`) . . . 27
5.3 Vzdálená konfigurace . . . 27
5.3.1 Konfigurační zpráva . . . 27
5.3.2 Příkazy . . . 27

6 Další vývoj **30**
6.1 Rozvíjení možností zabezpečení . . . 30
6.1.1 Zabezpečení protokolem TLS/SSL . . . 30
6.1.2 Doplnění rozšiřujících funkcí poštovních protokolů . . . 30
6.2 Zabezpečení konfiguračního souboru . . . 31
6.3 Portování na vestavěný systém . . . 31
6.4 Výměna klíčů . . . 31
6.5 Zabezpečení komunikce klient – proxy . . . 31
6.6 Automatizace některých činností . . . 32
6.7 Grafická konfigurační aplikace . . . 32

7 Závěr **33**

Kapitola 1

Úvod

Elektronická pošta patří mezi nejstarší a nejvyužívanější služby internetu. Právě kvůli své „dlouhověkosti“ s sebou přináší některé neduhy, jež si tehdy první uživatelé a návrháři protokolů nedokázali ani představit.

S narůstajícím významem a objemem komunikace elektronickou poštou vznikla a stále existuje potřeba zabezpečit takto přenášená data. K tomuto účelem existuje mnoho různých přístupů, které umožňují jednotlivým uživatelům zabezpečit komunikaci mezi sebou navzájem (případně další aspekty komunikace, například vyzrazení hesla).

Tato bakalářská práci si klade za úkol vytvořit základ sytému pro zabezpečení komunikace pomocí elektronické pošty s možností vzdálené konfigurace starající se o toto zabezpečení. Jednou z nejdůležitějších funkcí je možnost centrální definice bezpečnostních politik pro neomezený počet uživatelů, což lze využít například ve velké firmě, kde s minimálním aktivním zapojením většiny zaměstnanců kompletně obstaráme zabezpečení komunikace s okolím. Centrální definice je vhodná zejména z toho důvodu, že odstíní uživatele od věcí, kterým nerozumí, a systém od chyb z toho vyplývajících – to přináší obrovskou výhodu v přenesení téměř veškeré zodpovědnosti z laiků (uživatelů) na odborníky (administrátory). Uživatel se prostě o zabezpečení nemusí vůbec starat, stačí, že administrátor správně nastaví bezpečnostní politiky pro komunikaci. Jednou z nejdůležitějšich funkcí aplikace je právě tato možnost vynutit určitou bezpečnostní politiku pro daný cíl emailové komunikace a tím eliminovat potenciální záměrné i neúmyslné chyby uživatelů při zabezpečování emailových zpráv, které mohou nastat při použití jiných způsobů zabezpečení.

Druhá kapitola práce se zabývá elektronickou poštou, její historií, principem fungování a popisem používanách poštovních protokolů.

Kapitola tři zkoumá z různých pohledů existující mechanismy určené k zabezpečení elektronické pošty.

Ve čtvrté kapitole se dostáváme k návrhu vytvářeného systému, jsou zde zhodnoceny výhody a nevýhody použitého přístupu a obecně popsány důležité části návrhu systému.

Samotná implementace je konkrétně popsána v kapitole páté, jež se podrobněji zabývá jednotlivými podsystémy a také vzdálenou konfigurací.

Šestá kapitola ukazuje směry, kterými se může ubírat budoucí vývoj.

Práce navazuje na semestrální projekt, jehož cílem byla analýza problematiky související s implementací proxy výsledného proxy serveru.

Ve své práci jsem také částečně využil poznatků z diplomové práce Dalimila Hrabovského *Grafické rozhraní pro konfiguraci bezpečné pošty*. Bohužel systém zabezpečení v ní vytvořený měl natolik chudé možnosti definice bezpečnostních politik, až byl nepoužitelný. [3]

Poznámka Součástí vytvořeného systému je i proxy server pro protokol IMAP. Ten není mým dílem, proto se mu a teorii s ním související budu věnovat jen okrajově. Za těžiště práce považuji zbytek systému – SMTP proxy, POP3 proxy, konfigurační server a ostatní moduly.

Kapitola 2

Elektronická pošta

Elektronická pošta patří v současné době spolu se službou WWW (World Wide Web) k nejvyužívanějším službám internetu. Má dlouhou historii sahající až do poloviny šedesátých let dvacátého století.

2.1 Historie

Vznik elektronická pošty je přirozeně spjat s vývojem síťových možností objevujících se během vývoje internetu. Možnost výměny zpráv existovala již od raných počátků počítačů schopných věnovat výpočetní čas více uživatelům. Email tak, jak jej známe dnes, byl vyvinut pro ARPANET krátce po jeho vzniku a rozrostl se do netušených rozměrů.

V období prvních počítačů schopných spouštět více procesů zároveň vznikaly první programy schopné výměny textových zpráv, a dokonce i chatu v reálném čase. Takováto komunikace byla ale omezena na uživatele jednoho počítače.

Na počátku sedmdesátých let dvacátého století vznikla první aplikace k přenosu elektronická pošty po ARPANETU. Rovněž je do tohoto období zasazeno vytvoření formátu emailové adresy v podobné formě, v jaké ji známe dnes (user@host).

V roce 1972 byly do protokolu FTP[1] přidány příkazy určené k přenosu emailů (`MAIL` a `MLFL`). Spočívaly v kopírování souborů přes síť. Tento model se udržel až do počátku osmdesátých let, kdy vznikl protokol SMTP (viz část 2.4.1).

Sjednocení formátu emailových zpráv přineslo RFC 733 (rok 1977) později nahrazené RFC 822 (rok 1982). [1]

2.2 Princip

Celou službu elektronické pošty si můžeme představit jako paralelu k obyčenjné „papírové“ poště. Princip je velmi podobný, pouze se liší používané nástroje a prostředky. Stejně jako při využívání normální pošty obecně chceme někomu sdělit nějakou informaci.

Prvním krokem je vytvoření zprávy. Ovšem místo pera, papíru a obálky použijeme počítač a na něm příslušný software, který se nazývá emailový klient (někdy ve zkratce MUA – Mail User Agent).

Potom musíme „zajít s dopisem na poštu“. Uděláme to tak, že zkontaktujeme vhodný server pro odesílání elektronické pošty (ve zkratce MTA – message transfer agent). Tím, že

[1] File Transfer Protocol

poskytneme informaci od koho zpráva je a komu je určena, může být při přenosu vytvořena jakási obálka, ve které je uveden odesilatel a příjemce.

Server supluje funkci pošty, dokáže zprávu předat až do cílové schránky. Při tomto průchodu elektronickými kanály může zpráva projít přes několik MTA, až je nakonec uložena na požadované místo.

Odtud si „dojdeme do schránky“ a použijeme k tomu nám známý emailový klient. Ten umí zprávu vyzvednout a posléze nám ji předloží.

2.3 Formát zprávy

Zpráva (email) je základní jednotkou komunikace elektronickou poštou. Jako cokoliv určeného k přenosu mezi počítači má přesně vymezenou strukturu, kterou musí dodržet, aby ji bylo možno bez problému předávat elektronickými komunikačními kanály. Formát je specifikován v RFC 2822 [7].

Zpráva může obsahovat pouze znaky dolní poloviny[2] znakové sady US-ASCII (konkrétně z rozsahu 1–127). Skládá se ze dvou částí, hlavičky a těla. Tyto části jsou při přenosu internetem odděleny prázdným řádkem (podřetězcem CR[3] LF[4] CR LF).

Hlavička v sobě nese různé informace o zprávě. Skládá se z libovolného počtu polí, které jsou odděleny konci řádků (CR LF). Tyto pole se skládají z názvu a obsahu, jež jsou oddělené dvojtečkou.

Omezení pro obsah hlavičky:

1. Pole nesmí obsahovat netisknutelné znaky[5].
2. Název pole nesmí obsahovat znak dvojtečka.
3. Délka řádku[6] nesmí přesáhnout 998 znaků (doporučuje se nepřesáhnout 78 znaků) bez CR LF.

Pole specifikující předmět zprávy může vypadat například takto:

```
Subject: This is testing email
```

Hlavička musí povinně obsahovat pole *Date* a *From*.

Omezení pro obsah těla:

1. Znaky CR a LF se nesmí vyskytnout odděleně, pouze spolu za sebou v pořadí CR LF.
2. Délka řádku nesmí přesáhnout 998 znaků (doporučuje se nepřesáhnout 78 znaků) bez CR LF.

Velmi krátká zpráva schopna přenosu elektronickou poštou pak může vypadat například takto:

[2] Toto lze odstranit použitím formátu MIME – Multipurpose Internet Mail Extensions **??**.

[3] Carriage return (návrat vozíku) je znak sady US-ASCII s ordinální hodnotou 13.

[4] Line feed (nový řádek) je znak sady US-ASCII s ordinální hodnotou 10.

[5] Tisknutelné znaky sady US-ASCII jsou od znaku s ordinální hodnotou 33 do znaku s hodnotou 126 včetně.

[6] Řádky jsou úseky zprávy oddělené dvojicí znaků CR LF.

```
Date: Wed, 02 May 2007 23:50:46 +0200 (CEST)
From: Blacklanner <Blacklanner@seznam.cz>

Ahoj Stando, posilam Ti seznam, ktery jsi po mne chtel. Pavel
```

2.4 Používané protokoly

Při přenosu elektronické pošty internetem dochází k předávání zpráv mezi jednotlivými počítači. Stejně tak vybírání schránky na serveru je záležitost komunikace počítačových programů. Proto musí existovat protokoly, které tuto komunikaci přesně specifikují. V této kapitole popíšu dnes používané protokoly určené pro využití služby elektronické pošty.

Pro odesílání (komunikace MUA s MTA) a přeposílání (komunikace MTA s jiným MTA) se v dnešní době takřka výhradně používá protokol SMTP nebo jeho rozšíření (ESMTP). K vybírání poštovní schránky slouží protokoly POP3 (zatím stále ještě více používaný) a novější protokol IMAP.

2.4.1 SMTP – Simple Mail Transfer Protocol

Úkolem tohoto protokolu je spolehlivě a efektivně přenášet emailové zprávy. Zajišťuje prakticky veškerou komunikaci elektronickou poštou kromě posledního kroku – vyzvednutí zpráv ze schránky.

Historie

Definici protokolu napsal v roce 1982 Jonathan Postel jako RFC 821. Už předtím sice prodělal určitý vývoj, ale toto byl velký mezník v jeho historii. Protokol vznikl jako náhrada protokolu MTP (Mail Transfer Protocol), což byl do té doby používaný protokol pro přenos elektronické pošty založený na FTP (File Transfer Protocol). SMTP byl mnohem elegantnější, navržený jen pro přenos elektronické pošty – díky tomu mohl obsahovat potřebné specifické vlastnosti, které MTP z podstaty nepodporoval a podporovat nemohl.

Během osmdesátých let dvacátého století se stával stále oblíbenějším a používanějším, nicméně neprocházel žádným dalším vývojem.

To se změnilo v únoru 1993, kdy bylo vydáno RFC 1425 – SMTP Service Extensions. V něm byl definován způsob přidávání rozšíření, aniž by tím utrpěla zpětná kompatibilita. SMTP s tímto rozšířením se někdy nazývá ESMTP (Extended/Enhanced Simple Mail Transfer Protocol – Rozšířený SMTP).

Současný standard je definován v RFC 2821 (vydáno v dubnu 2001). Toto RFC sdružuje základ protokolu popsaný v RFC 821 a veškerá do té doby definována rozšíření (do té doby v RFC 1869). [5]

Popis komunikace

Komunikace mezi klientem a serverem probíhá jako výměna řádků s příkazy. Jednotlivé řádky jsou ukončeny znaky CR LF a jejich délka nesmí přesáhnout 998 znaků.

Během komunikace se mění množina použitelných příkazů. Typický průběh komunikace vypadá asi takto:

1. `EHLO` – klient se identifikuje, zjistí podporovaná rozšíření
2. `MAIL` – začne přenos emailu

3. `RCPT` – identifikuje příjemce zprávy
4. `DATA` – pošle obsah emailu
5. `QUIT` – vyžádá ukončení spojení

Při odeslílání více zpráv se vícekrát opakuje smyčka `MAIL` → `RCPT` → `DATA`. Pokud má email více příjemců, je vícekrát zadán příkaz `RCPT`.

Jednoduchý příklad odeslání emailové zprávy je naznačen v tabulce 2.1.

```
S:  220 Welcome to SMTP server
C:  HELO pcbuslab9.fit.vutbr.cz
S:  250 Hello pcbuslab9.fit.vutbr.cz
C:  MAIL FROM:<xzidek05@stud.fit.vutbr.cz>
S:  250 Sender OK
C:  RCPT TO:<Blacklanner@seznam.cz>
S:  250 Recipient OK
C:  DATA
S:  354 Send mail data, end with CRLF.CRLF
C:  <Samotný email>
C:  .
S:  250 Mail OK
C:  QUIT
S:  250 SMTP server signing off
```

Tabulka 2.1: Příklad komunikace pomocí protokolu SMTP

Základní množina příkazů

Minimální množinu příkazů tvoří příkazy, které musí podporovat každý správně naimplementovaný SMTP server, aby byl kompatibilní s jakýmkoliv správně naimplementovaným klientem.

Tyto příkazy jsou přehledně popsány v tabulce 2.2.

Příkaz	**Význam**
`EHLO`	začne komunikaci, identifikuje klienta, signalizuje podporu ESMTP na straně klienta
`HELO`	začne komunikaci, identifikuje klienta
`MAIL`	započne doručování emailu
`RCPT`	určí příjemce emailu
`DATA`	odstartuje posílání obsahu emailu
`RSET`	zruší prováděný přenos emailu
`NOOP`	nedělá nic
`QUIT`	ukončí spojení
`VRFY`	vyžádá potvrzení, že jeho argument identifikuje uživatele či poštovní schránku

Tabulka 2.2: Minimální množina příkazů protokolu SMTP

Rozšíření protokolu

Původní protokol SMTP byl schopen poskytnout pouze omezené množství přesně definovaných služeb a funkcí. Elektronická pošta a tím pádem i způsoby a možnosti jejího doručování se však neustále vyvíjí a mění, a proto vznikla potřeba vytvořit obecný mechanismus, kterým by se dal protokol rozšiřovat o nové funkce bez ztráty funkčnosti starších klientských programů.

Řešení tohoto problému přineslo RFC 1425 a jeho následníci (1651, 1869, 2821[7], které definují:

- příkaz `EHLO`, náhradu dřívějšího `HELO`,
- registry rozšíření protokolu SMTP
- další parametry příkazů `MAIL` a `RCPT`
- volitelné náhrady příkazů SMTP, například `DATA` pro přenos znaků jiných než US-ASCII

Takto „vylepšený“ protkol SMTP se nazývá ESMTP (Extended/Enhanced Simple Mail Transfer Protocol – Rozšířený SMTP).

Je navržen tak, aby byl zpětně kompatibilní s původním, nerozšířeným SMTP. Zavádí nový příkaz, `EHLO` (tj. Extended HELO), jímž klient oznamuje, že by rád komunikoval právě tímto rozšířeným protokolem. Pokud ESMTP podporuje i server, po tomto pozdravu zašle víceřádkovou odpověď, ve které sdělí, které služby podporuje. Pokud server ESMTP nezná, potom nerozpozná ani příkaz `EHLO`, vydá hlášení o chybě a klient může zkusit obyčejný pozdrav `HELO`. Pro příklad viz tabulky 2.3 a 2.4.

```
S:  220 Welcome to ESMTP server
C:  EHLO pcbuslab9.fit.vutbr.cz
S:  250-Hello pcbuslab9.fit.vutbr.cz
S:  250-8BITMIME
S:  250-SIZE
S:  250-AUTH CRAM-MD5 DIGEST-MD5
S:  250-DSN
S:  250 HELP
C:  <klient může v další komunikaci využít deklarované rozšíření
    (8BITMIME, SIZE, AUTH, DSN a HELP)>
```

Tabulka 2.3: Příklad úspěšného pozdravu EHLO

Server oznámí, která rozšíření podporuje, pomocí víceřádkové odpovědi na příkaz `EHLO` (viz tabulku 2.3). Každý řádek této odpovědi kromě prvního obsahuje název `jednoho` podporovaného rozšíření s případnými parametry.

Přehled některých známých rozšíření je v tabulce 2.5 [8].

2.4.2 POP3 – Post Office Protocol (version 3)

Protokol POP3 je v současné době nejrozšířenější protokol sloužící k přenosu zpráv z poštovní schránky na vzdáleném serveru do počítače klienta. Používá se k offline přístupu

[7] Sloučeno s definicí SMTP.

```
S:  220 Welcome to SMTP server
C:  EHLO pcbuslab9.fit.vutbr.cz
S:  500 Error, unrecognized command
C:  HELO pcbuslab9.fit.vutbr.cz
S:  250 Hello pcbuslab9.fit.vutbr.cz
C:  <komunikace pokračuje jako obyčejně protokolem SMTP>
```

Tabulka 2.4: Příklad neúspěšného pozdravu EHLO a následný pozdrav obyčejný

8BITMIME	Přenos osmibitových znaků	RFC 1652
ATRN	Přeposílání emailových zpráv na požádání	RFC 2645
AUTH	Autentizace	RFC 2554
CHUNKING	Přenos velkých a binárních MIME zpráv	RFC 3030
DSN	Oznámení o stavu doručení	RFC 1891
ETRN	Řazení zpráv do front	RFC 1985
HELP	Poskytování pomocných informací	RFC 821
PIPELINING	Zřetězení příkazů	RFC 2920
SIZE	Deklarace velikosti zprávy	RFC 1870
STARTTLS	Bezpečnost na úrovni transportní vrstvy	RFC 3207

Tabulka 2.5: Seznam nejznámějších rozšíření SMTP. [8]

ke zprávám, který je vhodný především proto, že každý příjemce pošty nemusí na své pracovní stanici nechávat běžet proces poskytující služby SMTP serveru s funkcí lokálního doručování (ukládání) zpráv.

Historie

Offline přístup k emailům začíná svůj vývoj na začátku osmdesátých let dvacátého století. Vývojaři si už tehdy uvědomili výhody stahování pošty do lokálního počítače (oproti přístupu na serveru pomocí telnetu nebo NFS[8]) a vytvořili specifikaci protokolu POP (Post Office Protocol) v RFC 981 (rok 1984). Důraz byl kladen především na jednoduchost. Celé RFC 981 je 5 stránek dlouhé a v podstatě říká to, že uživatel pošle své jméno, heslo a následně mu server odešle *celý* obsah jeho schránky.

Tento přístup měl ovšem onu zásadní nevýhodu neustálého stahování veškerého obsahu, a proto v roce 1985 byl v RFC 987 definován protokol POP2 (Post Office Protocol – Version 2), který přinesl možnost stahovat pouze konkrétní zprávu.

Tyto dva protokoly byly používány během osmdesátých let, avšak ne příliš široce. Offline přístup k emailům nebyl ještě tehdy tak nutný, internet používal zanedbatelný počet uživatelů.

V roce 1988 vzniklo RFC 1081 definující protokol POP3 (Post Office Protocol – Version 3). V této době docházelo k velkému nárůstu významu osobních počítačů (PC – Personal Computer). POP3 zdokonalil a rozšířil možnosti POP2 a poskytl osobním počítačům i jiným klientům dobrý způsob přísupu a stahování emailových zpráv.

Vývoj pokračoval dále v devadesátých letech, co pár let bylo vydáno nové RFC (postupně to byly 1225, 1460, 1725 a 1939). I přes tento poměrně velký počet revizí se protokol samotný od roku 1988 v podstatě nezměnil a od roku 1996, kdy byl vydán poslední

[8]Network File System

jmenovaný dokument, RFC 1939, nebyl měněn vůbec. Vznikaly pouze definice volitelných rozšíření, jako například alternativní autentizační mechanismus.

Celý vývoj se nesl v duchu myšlenky jednoduchosti, přímočarosti a rychlosti. [4]

Popis komunikace

POP3 je přímočarý třístavový protokol. Obsahuje relativně malý počet příkazů určených k jednoduchým úkonům.

Komunikace se může nacházet ve třech stavech: autorizačním (authorization state), transakčním (transaction state) a aktualizačním (update state). Postupně si tyto stavy rozebereme.

Authorization state V tomto stavu je po klientovi požadováno, aby se autentizoval. K tomu slouží kombinace příkazů `USER` a `PASS` nebo příkaz `APOP`. Server musí podporovat alespoň jeden z těchto způsobů. Navíc může být v tomto stavu vydán jen příkaz `QUIT` k ukončení komunikace. Pokud se klient úspěšně autentizuje, sezení přechází do transakčního stavu.

Transaction state V tuto chvíli má klient otevřenu svou schránku s exkluzivním přístupem. Nejdůležitější příkazy v tomto stavu jsou bezesporu `LIST`, vracející seznam zpráv a případně jejich velikosti, a `RETR`, sloužící ke stažení určené zprávy. Rovněž lze použít (mimo jiné) příkaz `DELE` (označí zprávu jako určenou ke smazání) či `RSET` (zruší označení zpráv určených ke smazání). Tento stav se ukončí příkazem `QUIT` a sezení přejde do aktualizačního stavu.

Update state V tomto stavu jsou všechny zprávy určené ke smazání skutečně smazány. Tím je zajištěno, že pokud komunikace neskončí korektně (posláním `QUIT`), žádné emaily odstraněny nebudou.

Jednoduchý příklad stažení zprávy ze serveru je uveden v tabulce 2.6.

Základní množina příkazů

V tabulce 2.7 jsou shrnuty základní příkazy protokolu POP3.

Rozšíření protokolu

Protokol POP3 obsahuje příkazy, které nejsou naprosto nutné ke komunikaci, ale mohou ji usnadnit, popřípadě pomoci zabezpečit. Viz tabulku **??**.

Jako nevyhnutelný krok ve vývoji protokolu (stejně jako u SMTP) přišla možnost definovat různá rozšíření. V protokolu POP3 je toto zajištěno pomocí příkazu `CAPA` a registrovaného seznamu rozšíření.

Příkaz `CAPA` Tento příkaz je definován v RFC 2449 (POP3 Extension Mechanism). Může být poslán v autentizačním i transakčním stavu. Jako odpověď server pošle seznam „schopností“ (capabilities), kterými disponuje. Klient poté může tyto využívat.

Základní množina těchto capabilities je definována přímo ve stejném RFC. Jejich přehled se nachází v tabulce 2.9.

Definice dalších rozšíření není doporučována z důvodu zachování jednoduchosti protokolu.

```
C:  <navázání spojení>
S:  +OK POP3 server ready
C:  USER john
S:  +OK Enter your password please
C:  PASS heslo
S:  +OK You can work now
C:  STAT
S:  +OK 1 130
C:  LIST
S:  +OK 1 message (130 octets)
S:  1 130
S:  .
C:  RETR 1
S:  +OK 130 octets
S:  <Samotný email>
S:  .
C:  QUIT
S:  +OK POP3 server signing off
C:  <odpojení>
```

Tabulka 2.6: Příklad komunikace pomocí protokolu POP3

2.4.3 IMAP – Internet Mail Access Protocol

Tento protokol má stejný účel jako předchozí. Velký rozdíl ovšem je v celkovém pojetí.

Zatímco POP3 je po celou dobu svého vývoje směrován k pokud možno co největší jednoduchosti, někdy na úkor možnosti využití pokročilých funkcí, IMAP se vydal právě druhou cestou.

Přináší mnoho výhod, jmenujme za všechny jen hierarchickou strukturu poštovní schránky, možnost pracovat s poštou přímo na serveru (netřeba ji stahovat do lokálního počítače) nebo možnost stahovat jen vybrané části zprávy.

Jako důsledek je tento protokol o dost složitější a tím pádem náročnější k implementaci.

Příkaz	**Význam**	Stav
USER name	zadá uživatelské jméno	autorizační
PASS string	zadá uživatelovo heslo	autorizační
STAT	vrátí celkový počet a celkovou velikost zpráv	transakční
LIST [msg]	vypíše velikost zadané zprávy nebo postupně všech zpráv	transakční
RETR msg	pošle obsah zprávy s číslem msg	transakční
DELE msg	označí zprávu s číslem msg jako určenou ke smazání	transakční
NOOP	nedělá nic	transakční
RSET	zruší označení zpráv určených ke smazání	transakční
QUIT	ukončí spojení	kterýkoliv

Tabulka 2.7: Minimální množina příkazů protokolu POP3

APOP	zabezpečené přihlašování, viz 3.1.2	autentizační
TOP msg n	pošle hlavičku a n řádků těla zprávy s číslem msg	transakční
UIDL [msg]	vrátí unikátní identifikátor zprávy s číslem msg nebo všech zpráv	transakční

Tabulka 2.8: Další příkazy protokolu POP3

CAPA štítek	**Přidané příkazy**	**Význam**
TOP	TOP	podpora příkazu TOP (viz tabulku 2.8)
USER	USER, PASS	podpora autentizačního mechanismu USER/PASS
SASL	AUTH	podpora různých autentizačních mechanismů
RESP-CODES		rozšířené možnosti návratových kódů
LOGIN-DELAY		zavádí minimální čas mezi pokusy o přihlášení
PIPELINING		zřetězení příkazů
EXPIRE		umožňuje serveru oznámit, jak dlouho uchová zprávy před automatickým smazáním
UIDL	UIDL	podpora příkazu UIDL (viz tabulku 2.8)
IMPLEMENTATION		umožňuje serveru sdělit svou implementaci

Tabulka 2.9: Rozšíření protokolu POP3

Kapitola 3

Zabezpečení elektronické pošty

Elektronická pošta je jednou z nejstarších služeb internetu. Během svého předlouhého vývoje, ve kterém dospěla až k dnešnímu velmi masovému rozšíření, se vyskytlo mnoho nových požadavků a jedním z nich bylo i zabezpečení. Mnoho se změnilo od dob, kdy elektronickou poštu používalo pár nadšenců a několik univerzitních nebo armádních zaměstnanců. Dnes ji používají téměř všichni uživatelé internetu a posílají přes ní široké spektrum informací, od řetězových dopisů až po citlivé firemní údaje. Díky tomu je pro některé uživatele potřeba zabezpečení velmi kritická.

V této kapitole se podíváme na existující způsoby zabezpečení elektronické pošty.

3.1 Zabezpečení přihlašování k serveru

Poštovní protokoly na tom zpočátku byly velmi špatně z hlediska autentizačních mechanismů – buď neměly vůbec žádné (SMTP), nebo měly, ale nevhodné (POP3 a heslo posílané v otevřeném textu). Kvůli tomu existovalo a existuje nebezpečí, že se k serveru přihlašuje někdo jiný než má, například útočník, kterému se podařilo odposlechnout heslo při komunikaci protokolem POP3.

Tyto chabé možnosti autentizace s sebou přinášejí bezpečnostní rizika, a to například:

- Útočník se může vydávat za někoho, kým není (například použije heslo pro přístup k WWW rozhraní nebo k zabezpečenému SMTP serveru).
- Útočník si může číst veškeré zprávy uložené na serveru.
- Útočník může zprávy na serveru smazat, nedostanou se k zamýšlenému příjemci.
- Útočník může využít heslo ke změně na jiné a tím odříznout právoplatného uživatele od schránky.

První dva body můžeme vyřešit použitím zabezpečení na aplikační úrovni (viz 3.3), ale tím stále nevyřešíme vše. Jako obrana před těmito útoky byly do protokolů přidány různé (většinou ovšem nepovinné) autentizační mechanismy.

3.1.1 Příkaz AUTH(ENTICATE)

Jde o rozšíření vytvořené pro protokoly SMTP, POP3 i IMAP, sloužící k zabezpečnému přihlašování k serveru.

Server ohlásí, které autentizační algoritmy podporuje (jak, to záleží na protokolu, pro princip to je nepodstatné). Klient si poté některý z nich vybere a sdělí ho serveru jako parametr příkazu AUTH (nebo AUTHENTICATE v případě protokolu IMAP) – například `AUTH CRAM-MD5`[1]. Poté následuje výměna řetězců kódovaných pomocí base64, kterými se navzájem klient a server autentizují.

Po úspěšném ověření komunikace probíha komunikace dále normálním způsobem (srovnejte s 3.2.1).

Příkaz AUTH protokolu ESMTP

Extended/Enhanced Simple Mail Transfer Protocol (rozšířený SMTP protokol) nabízí možnost autentizace odesilatele pomocí příkazu AUTH.

Bohužel má tento příkaz pouze částečné využití. Protože příjemce zprávy neví, zda byl odesilatel autentizován, a protože nijak neřeší, zda se autentizoval uživatel, který je u dopisu uveden v poli hlavičky *From:*, slouží v podstatě jen pro boj s nevyžádanou poštou. [9]

3.1.2 Příkaz APOP protokolu POP3

Velký problém připojování k poštovní schránce protokolem POP3 je posílání hesla v prostém textu. Toto může být poměrně uspokojivě vyřešeno rozšířením protokolu o příkaz APOP.

V praxi to vypadá tak, že se uživatel neautentizuje pomocí přikazů USER a PASS, ale použije místo toho jen jeden, APOP. Server v tomto případě musí do svého pozdravu zahrnout pokud možno jedinečný řetězec. Ten klient využije tak, že jej konkatenuje s heslem a ze vzniklého řetězce vytvoří hash algoritmem MD5 (viz příklad 3.1).

```
S:  +OK POP3 server ready <unique_string>
C:  APOP username md5(<unique_string>password)
```

Tabulka 3.1: Schéma autentizace příkazem APOP

Nevýhodou tohoto řešení je nutnost uchovávat na serveru hesla v otevřeném textu (při autentizaci příkazy USER/PASS bohatě stačí otisky vytvořené vhodnou rozptylovací funkcí (MD5, SHA1)).

3.2 Zabezpečení celého průběhu komunikace

V tomto případě provádíme šifrování veškeré komunikace mezi klientem a serverem. Jednoznačnou volbou v této oblasti je protkol TLS/SSL.

3.2.1 Zabezpečení na úrovni transportní vrstvy, TLS/SSL

Protokol TLS (Transport Layer Security, nástupce protokolu SSL – Secure Sockets Layer) slouží k obecně k zabezpečení jakékoliv komunikace (tedy nejen elektronické pošty) na úrovni transportní vrstvy.

Takto by bylo možno vyřešit problém zatím odsunutý do pozadí, a to je zabezpečení přihlašovacích údajů. Jejich vyzrazení sice není úplně klíčové, útočník, který by se k nim

[1] Například tento algoritmus je svým principem velmi podobný průběhu autentizace pomocí příkazu APOP, navíc ale nevyžaduje uložení hesel v otevřeném textu.

dostal, si může přečíst (v ideálním případě) pouze zašifrované zprávy uložené ve schránce, ale i tak je schopen způsobit jisté potíže.

Rád bych ještě podotknul, že rozhodně nejde o náhradu zabezpečení obsahu emailových zpráv a nenahrazuje systémy k tomu určené (PGP, S/MIME...).

Nevýhody použití TLS [2]:

- potenciální výpočetní náročnost při větším počtu příchozích spojení, která může vyústit v nutnost použití hardwaru urychlujícího šifrovací operace či celkově velmi výkonného
- pro použití se všemi klienty nutnost podporovat jak samotné TLS na daném portu, tak příkaz rozšiřující aplikační protokol (STARTTLS)
- nutnost spravovat databázi certifikátů, případně je kupovat
- falešný pocit bezpečí uživatelů – opravdu *nejde* o náhradu šifrování obsahu emailů

Existují dva způsoby použití TLS, které nyní rozeberu.

TLS jako zabezpečení na úrovni transportní vrstvy

Prvním je klasické šifrování na úrovni transportní vrstvy. V tomto případě komunikace probíhá na jiném portu než obyčejně (viz tabulku 3.2 [6]) a jde v podstatě o zapouzdření celé komunikace poštovními protokoly.

Protokol	Port (bez TLS)	Port (TLS)	
SMTP	25	465	(odvoláno[2])
POP3	110	995	
IMAP	143	993	

Tabulka 3.2: Čísla protokolů při zabezpečení komunikace pomocí TLS

Nevýhodou tohoto řešení je právě nutnost dvou portů, s tím související problematičnost a dlouhodobost zavedení příslušných standardů.

TLS jako rozšíření aplikačních protokolů

Jako alternativa k předchozímu bylo zavedeno začlenění TLS přímo do aplikačních protokolů jako rozšíření. Toto je novější způsob, prosazovný například i IETF především kvůli nedostatku čísel portů.

Vypadá to tak, že se klient připojí pomocí SMTP, respektive POP3 protokolu k serveru a v určitou (poměrně přesně danou) chvíli požádá o změnu způsobu komunikace na zabezpečenou TLS pomocí příkazu STARTTLS, respektive STLS.

3.3 Zabezpečení obsahu emailových zpráv

I když používáme nezabezpečené poštovní protokoly, jsme schopni obsah zpráv zabezpečit. V podstatě jde o to, že použijeme šifrování nebo podepisování na tělo emailové zprávy a tím ji ochráníme před nežádoucími čtenáři nebo potvrdíme své autorství.

Tato oblast zabezpečení přísluší aplikační úrovni. Úloha zabezpečení je zde většinou dána do rukou koncovým uživatelům, kteří spoléhají na vlastní možnosti zabezpečení obsahu zprávy.

Výhody:

- Nevadí nám nezabezpečené poštovní protokoly.
- Nepotřebujeme žádnou speciální podporu ze strany serverů, kterou bychom si eventuálně neměli možnost u jiných způsobů vynutit.
- Nikdo nepovolaný si naši citlivou informaci nepřečte na serveru, kde je při použití jiných způsobů zabezpečení vždy uchovávána nezašifrovaně.

Nevýhody:

- Nijak neřeší možnost vyzrazení hesla pro přístup ke schránce.
- Toto řešení je spíše individuální než aplikovatelné na širší skupinu lidí. Pokud bychom chtěli donutit celou firmu používat něco takového, přineslo by to velmi obtížně řešitelné problémy.
- Zabezpečením celého obsahu můžeme znemožnit nalezení potencionálního malware[3], pokud je skenování prováděno kdekoliv jinde než na koncových stanicích.

Teoreticky bychom tedy mohli vzít obsah každé zprávy, kterou budeme odesílat, zašifrovat ji domluveným algoritmem a klíčem (podle příjemce) a poslat ji. V praxi se ale používají určité standardizované metody, jež ušetří hodně práce a starostí s nekompatibilitou. Zmiňme si tedy nejpoužívanější systémy.

3.3.1 S/MIME – Secure / Multipurpose Internet Mail Extensions

S/MIME je standard pro zabezpečení emailových zpráv založený na kryptografii pomocí veřejného klíče; používá/rozšiřuje formát zpráv MIME.

Původně byl vyvinut společností RSA Data Security Inc. jako systém založený na formátu MIME a PKCS #7 (Public Key Cryptography Standards). Posléze byl převzat IETF a učiněn součástí CMS (Cryptografic Message Syntax), což je de facto ve všech podstatných ohledech dvojník PKCS #7.

Systém funguje tak, že vezme celou MIME entitu[4], aplikuje na ní požadovanou operaci (šifrování, digitální podepsání) a vzniklá data vloží do nové MIME entity s obsahem deklarovaným jako `application/pkcs7-mime`.

Jde ve svém návrhu o centralizovaný systém, který ke svému chodu potřebuje certifikační autoritu[5] (CA – Certificate Authority). Bezpečnost komunikace je dána tím, že obě strany věří certifikační autoritě, od které si vyžádají veřejný klíč druhé strany. Nevýhodou tohoto přístupu je, že pokud je CA kompromitována, má to dopad na všechny uživatele, kteří jí věří.

[3] Malware – MALicious softWARE, škodlivý sowtware, jako například viry, trojští koně, červi, spyware, adware...

[4] MIME entita je emailová zpráva, případně její tělo.

[5] instituce schopná vydat digitální certifikát

3.3.2 PGP – Pretty Good Privacy

PGP je počítačový program určený k šifrování a podepisování obecně jakýchkoliv dat. Byl vytvořen roku 1991 Philipem Zimmermannem. Díky svému velkému rozšíření byly pro potřebu komunikace elektornickou poštou vytvořeny standardy RFC 2015 (MIME Security with Pretty Good Privacy) a RFC 2440 (OpenPGP Message Format).

K ověření, zda veřejný klíč patří skutečně zmýšlenému příjemci, má PGP dva mechanismy.

Web of trust Starší způsob ověření používá takzvanou „pavučinu důvěry“. Vypadá to tak, že jednotliví uživatelé se mohou rozhodnout věřit tomu, že daný veřejný klíč patří tomu, komu hlásá. Pak tento klíč s informacemi, komu patří, podepíší svým podpisem a takto dají svou důvěru najevo ostatním uživatelům. Tím vznikne jakási distribuovaná síť, ve které si jednotlivci navzájem více či méně věří. Míra důvěry v určitý klíč se pak přepočítáva při každé změně v sadě klíčů, kterým důvěřuji.

Certifikační autority Ve specifikaci OpenPGP existuje možnost vytvořit takzvané *trust signatures*, které umožňují tvorbu certifikačních autorit. Pokud klíč podepíšu pomocí *trust signature*, značí to mou důvěru v to, že klíč patří udávanému vlastníkovi a zároveň že tomuto vlastníkovi věřím v otázce podpisu jiných klíčů. Podpis nulté úrovně je podobný už zmiňované „pavučině důvěry“, indikuje, že věřím pouze identitě vlastníka. Podpis první úrovně je analogický důvěře v certifikační autoritu, takto podepsanému klíči věřím každý klíč nulté úrovně, který podepíše. Existuje ještě podpis druhé úrovně – klíč, který takto podepíšu, pro mě vlastně může vytvářet certifikační autority.

Kapitola 4

Návrh

Cílem této bakalářské práce bylo vylepšit existující implementaci emailové proxy a rozšířit ji o možnost vzdálené konfigurace. V této kapitole popíšu, jak byl celý systém navržen, proč zrovna tak, jeho výhody a nevýhody, cílové prostředí pro nasazení a podobně.

4.1 Celková filosofie

Celý systém funguje jako *proxy server*[1]. Má za úkol zabezpečit veškerou komunikaci, která přes něj prochází, podle definovaných politik.

Zabezpečení se děje na úrovni obsahu zpráv, tedy na aplikační úrovni. Systém je schopen odchytit zprávu, jež je přes něj odesílána serveru nebo doručována klientovi, a její obsah patřičně pozměnit. Pokud jde o odesílání, pak zpráva může být (podle nastavení příslušných politik mezi odesilatelem a příjemcem) zašifrována a podepsána. Při doručování proběhne opačný proces, je ověřen podpis, zpráva je rozšifrována a v otevřeném textu předána příjemci.

4.1.1 Výhody

Asi největší výhodou takovéhoto systému je požadované zapojení uživatele a nároky na jeho znalosti, které jsou prakticky nulové. O vše podstatné se stará proxy server a jeho administrátor.

Dále je systém pro koncového uživatele prakticky neviditelný. V podstatě jedinou věcí, jak na sebe proxy upozorňuje, jsou potenciální varování o nesedícím podpisu nebo nemožnosti rozšifrování.

Velmi důležitá je možnost vytváření poměrně komplexních politik zabezpečení na několika různě prioritních úrovních (například celkové nastavení proxy může přebít nastavení uživatelovo).

Výhodou je také možnost vzdálené konfigurace, která umožní proxy serveru běžet na jakkoli nedosažitelném místě, k němuž administrátor vůbec nemusí mít fyzický přístup.

Systém je také poměrně modulární. Většinu komponent lze bez větších problémů nahradit jejich jinak řešenými ekvivalenty, jediný požadavek je dodržení rozhraní. Například pokud se nám nelíbí, že se nastavení ukládá do XML, můžeme si ho uložit kamkoliv, od textového souboru po SQL databázi; jedinou podmínkou je už zmíněné zachování rozhraní.

Celkové pojetí také přináší velkou výhodu v možnosti přidat k proxy serveru další dnes žádané funkce, jako například testování na obsah malware nebo filtrování nevyžádané pošty.

[1] Proxy server je server obsluhující své klienty posíláním žádostí a přijímáním odpovědí od jiných serverů. V podstatě se v komunikaci nachází mezi klientem a serverem, veškerá data procházejí přes něj.

Pokud by se zabezpečení dělo už na klientských počítačích, toto by v žádném případě nebylo možné.

4.1.2 Nevýhody

Proxy zatím nijak neřeší zabezpečení konfiguračního souboru. Pokud by se útočník dostal k zapnutému počítači, na kterém proxy běží, mohl by poměrně snadno narušit celou komunikaci, například zakázáním šifrování. Nebo může změnit veřejné klíče uživatelů, což s sebou přinese tu svízel, že místo zamýšleného přijemce si šifrované emaily bude číst právě útočník, případně nám bude posílat zprávy, které budou naoko podepsány někým jiným.

Podobným problémem je i uložení soukromých klíčů. Jsou uloženy v souboru a chráněny heslem, bohužel ale k nim proxy musí mít kdykoliv přístup a proto si toto heslo musí někde ukládat. Pokud by se tento soubor dostal do rukou útočníkovi (i kdyby bez potřebných hesel), znamenalo by to doslova katastrofu.

Další oblastí, na které by se dalo zapracovat, je využití lepších autentizačních mechanismů, které by zabránily útočníkovi získat hesla uživatelů.

Proxy server se také zatím nedá použít spolu s SMTP serverem vyžadujícím autentizaci, což souvisí s předchozím bodem. Existuje tedy v podstatě nutnost mít ve vnitřní síti vlastní SMTP server, který bude proxy věřit bez autentizace, servery v internetu toto nedělají kvůli boji s nevyžádanou poštou (spammem).

Systém potřebuje neustále běžící počítač, na kterém může fungovat. Není příliš vhodné používat jej na počítači, na kterém se děje i jiná uživatelská práce, ať už z hlediska nutnosti spouštění jako uživatele root, kvůli občasným nutným restartům (kdo nemusel při práci nikdy restartovat počítač...) nebo například pro omezení výkonu. V podstatě si takto vyžaduje jeden počítač sám pro sebe, což může být další finanční zátěž. Řešení navrhuje podkapitola 6.3.

4.2 Cílové prostředí

Systém je obecně navržen k použití více uživateli sdílejícími společnou *bezpečnou síť*, která má přístup k internetu (jinak bychom nemohli nic posílat).

Jako ideální případ se jeví například velká firma používající elektronickou poštu k posílání citlivých informací. Zde může systém ušetřit mnoho starostí souvisejících s nepoučeností uživatelů, viz podkapitola 4.1.1.

Nic nebrání ani tomu mít proxy server spuštěný na lokálním počítači a využívat ho jen pro sebe. Tím ale nebudeme schopni využít mnoho služeb, které poskytuje, celý mechanismus definice bezpečnostních politik se degraduje na využívání pouze jeho zlomku. Nicméně využít systém takto je také možné.

4.3 Rozšiřitelnost a modularita

Celý systém je navržen pokud možno co nejvíce modulárně, především z důvodů snadné rozšiřitelnosti, možnosti změny implementace podsystémů, přehlednosti a udržovatelnosti.

Rozšiřitelnost Jádro systému poskytuje poměrně obecné funkce, jež mohou být využity pro zabezpečení emailové komunikace probíhající téměř jakýmkoliv protokolem. Proto například není velkým problémem přidat další modul zabezpečující komunikaci dosud nepoužívaným protokolem.

Modularita Systém byl navržen tak, aby se pokud možno co nejvíce zjednodušila komunikace mezi jednotlivými moduly. Ne vždy to je sice možné stoprocentně dodržet (například v databázi je velmi mnoho druhů údajů a proto je potřeba více funkcí pro přístup k nim), nese to s sebou určité komplikace (například z hlediska výkonu), ale celkově jde o dle mého názoru dobrý přístup zvyšující čitelnost a znovupoužitelnost kódu, potenciální budoucí vývojáři se díky tomu nemusí starat o jiné moduly než ty, na kterých opravdu chtějí pracovat.

4.4 Bezpečnostní politiky

Dost klíčovou částí je možnost nadefinovat si různé bezpečnostní politiky pro různé trasy dopisů. Celkový návrh o tomto říká pouze to, že je možné tuto politku definovat dvakrát, jednou pro jednotlivé uživatele a podruhé pro celé domény. Více o tomto v popisu konkrétní implementace v části 5.2.2.

Kapitola 5

Implementace

V této kapitole popíšu skutečnou implementaci systému, použité technologie, strukturu aplikace. Velmi důležitá je i vzdálená konfigurace.

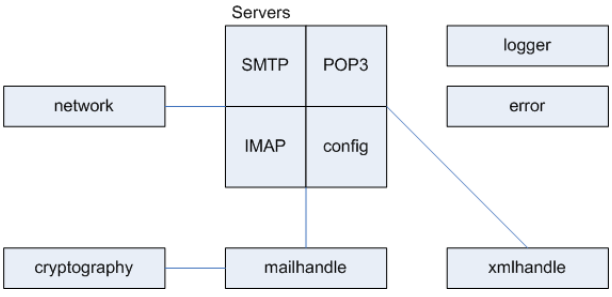
5.1 Použité technologie

Pro implementaci byl zvolen programovací jazyk C a (primárně) prostředí operačního systému GNU/Linux. Jazyk C především kvůli svému velkému rozšíření, jednoduchosti, rychlosti a přenositelnosti. Operační sytém GNU/Linux je s jazykem C velmi úzce svázán už od počátků svého vývoje, poskytuje skvělé nástroje vhodné pro ladění a kontrolu chyb (gdb, ddd, valgrind) a jsou v něm jednoduše použitelné zvolené pomocné knihovny.

Z důležitých nestandardních knihoven bych rád jmenoval hlavně cryptlib (velmi mocná kryptografická knihovna s poměrně jednoduchým použitím) a libxml2 (knihovna určená pro práci s XML).

5.2 Struktura

V této kapitole se konečně dostáváme k popisu jednotlivých podsystémů/modulů. Pro základní přehled viz obrázek 5.1, podrobnější popis jednotlivých částí následuje. Při tomto popisu se snažím i trochu naznačit rozhraní, které používají jednotlivé moduly pro vzájemnou domluvu.



Obrázek 5.1: Schéma systému

5.2.1 Server

Toto je jedna z nejdůležitějších částí systému. Obecně jde o jakýkoliv server, který bude vuyžívat možností poskytované zbytkem systému. V našem konkrétním případě na místě obecného serveru běží SMTP, POP3 a IMAP proxy server plus server konfigurační.

SMTP proxy (`smtp`)

V tomto modulu je naimplementován klasický konkurentní server[1]. Obsluhuje připojení od různých klientů, vlastně jim předstírá jednoduchý SMTP server, a zprávy, které mu pošlou, upravuje podle definovaných politik a posílá nyní už skutečným SMTP serverům.

Využívá především funkci `ProcessOutgoingEmail` z modulu `mailhandle` (viz část 5.2.4), která je schopná provést veškeré nutné transformace.

POP3 proxy (`pop3`)

Tento modul je myšlenkou velmi podobný SMTP proxy. Rozdíl je v tom, že tolik nemění komunikaci klienta se serverem, velkou část nechává projít beze změny narozdíl od SMTP proxy.

Do komunikace se vloží jen ve dvou místech.

1. Při autentizaci (příkazy `USER` a `PASS`) musí zjistit, ke kterému POP3 serveru daný uživatel náleží. To není tak jednoduché, protože se uživatel identifikuje jen uživatelským jménem. Řeším to tak, že buď se uživatelův MUA nastaví tak, aby se identifikoval celou emailovou adresou, nebo se v databázi hledá jen přihlašovací jméno použitelné jen v případě, že nedojde ke kolizi (dva uživatelé mohou mít stejné jméno, pokud používaji různé POP3 servery).

2. Pokud je poslán příkaz `RETR msg`, odpovědí na něj je požadovaná zpráva. Tu musíme odchytit a případně rozšifrovat a ověřit podpis. K tomu slouží funkce `ProcessIncomingEmail` z modulu `mailhandle` (viz část 5.2.4).

IMAP proxy (`imap`)

Svým principem je tato část podobná POP3 proxy (viz podkapitolu 5.2.1). Protokol IMAP má pouze mnohem víc funkcí, což s sebou přináší větší náročnost. Ke zpracování zprávy slouží opět funkce `ProcessIncomingEmail` z modulu `mailhandle` (viz část 5.2.4). Více se touto částí zabývat nebudu.

Konfigurační server (`config`)

Konfigurační server má poměrně jednoduchou úlohu. V zadaných časových intervalech kontroluje určenou emailovou schránku na přítomnost nového konfiguračního emailu. Pokud takový objeví, stáhne jej (pomocí protokolu POP3) a zavolá funkci `ProcessConfigurationEmail`. Pokud tato funkce vrátí ukazatel různý od `NULL`, pak jako odpověď na kofigurační email pošle právě zprávu, na kterou zmíněný ukazatel míří. Více o vzdálené konfiguraci v podkapitole 5.3.

[1] Konkurentní server je schopen v jednu chvíli obsluhovat více klientů

5.2.2 Databáze nastavení (`xmlhandle`)

Pro formát databáze byl zvolen XML soubor, a to především z důvodu kompromisu mezi čitelností lidským okem a rychlostí strojového zpracování. Toto se ukázalo jako dobrá volba i z pohledu jednoduchého řešení vzdálené konfigurace (viz část 5.3).

Databáze je rozdělena na čtyři části, které postupně rozeberu.

Uživatelé

Tato část obsahuje nastavení pro jednotlivé uživatele. Z údajů uchovávaných u každého uživatele stojí za zmínky především tyto:

- přihlašovací jméno na POP3 server
- emailová adresa
- adresa SMTP serveru a číslo portu
- adresa POP3 serveru a číslo portu
- adresa IMAP serveru a číslo portu
- otisk hesla pro výměnu klíčů[2]
- nastavení politik v kompetenci uživatelů
- soubor klíčů – veřejné a soukromé kliče uživatele; jeden je určen jako univerzální, u (případných dalších je určeno, pro komunikaci s kterými uživateli jsou určeny)

Nastavení politik se může skládat z nastavení pro jednotlivé uživatele a ze skupinových politik. U každé politiky je tímto určeno, pro komunikaci s kým je určena. Dále toto nastavení obsahuje požadavky na komunikaci s daným uživatelem nebo skupinou, a to:

- zda se má šifrovat
- zda se má podepisovat
- co z toho se má dělat dříve
- co se má dělat, když se něco nepovede, zda poslat nezabezpečeně, nebo neposlat vůbec

Domény

V této sekci jsou nastavení pro jednotlivé domény. Opět se podívejme na uchovávané údaje:

- adresa domény
- seznam administrátorů – uřivatelů oprávněných měnit nastavení pro celou doménu
- nastavení politik v kompetenci domén
- soubor klíčů – veřejné a soukromé klíče uživatele; jeden je určen jako univerzální, u (případných dalšich je určeno, pro komunikaci s kterými uživateli jsou určeny)

Nastavení politik je podobné tomu u uživatelů. Zásadním rozdílem ale je to, že jde o nastavení ne pro komunikaci s jednotlivými uživateli, ale s celými doménami.

[2] zatím neimplementováno

Proxy

V této části se nachází nastavení pro proxy server a tím pádem i pro celý zde popisovaný systém.

- porty, na kterých běží jednotlivé „falešné“ servery pro komunikci s uživateli
- seznam uživatelů – administrátorů proxy
- informace týkající se vzdálené konfigurace – zda je povolena, údaje o poštovní schránce, informace o serveru odchozí pošty (používaný pro odpovědi a zaznamenávání událostí, logování)
- seznam míst, kam zapisovat logy
- centrální definice politik, mající nejvyšší prioritu
- jediný pár klíčů, sloužící především k zabezpečení konfiguračních zpráv

Definice politik je v tomto případě poněkud složitější. Kromě toho „kam“ si musíme uchovávat i „odkud“ (předtím to bylo jasné, vždy se vycházelo od aktuálního uživatele nebo domény). Toto nastavení je rozděleno na dvě části, uživatelskou a doménovou, v každé části lze definovat i politiky skupinové. Rovněž zde existuje mechanismus určení směru komunikace, na který se má daná politika aplikovat (od prvního k druhému, naopak, v obou směrech). Pro podrobnosti odkazuji čtenáře přímo do konfiguračního souboru, je poměrně názorný a pochopitelný.

Veřejné klíče

V této části se nachází veřejné klíče těch uživatelů nebo domén, kteří nespadají přímo pod daný proxy server a tedy nemáme jejich soukromý klíč. Forma tohoto oddílu je velmi podobná seznamu klíčů u jednotlivých uživatelů nebo domén, až na to, že zde nejsou informace o veřejných klíčích.

5.2.3 Kryptografie (`cryptography`)

Tento modul poskytuje jednoduchá rozhraní pro šifrování (s využitím knihovny `cryptlib`) a potřebné kódování. Pro přehled viz tabulku 5.1

Název	**Funkce**
Encrypt	zašifruje data
Decrypt	rozšifruje data
Sign	podepíše data
Verify	ověří podepsaná data
EncodeBase64	zakóduje data algoritmem base64
DecodeBase64	rozkóduje data zakódovaná algoritmem base64

Tabulka 5.1: Funkce kryptografického modulu

Zajímavý zde je způsob předávání klíčů. Kvůli požadované nezávislosti modulů na sobě navzájem nebylo možné použít vnitřní formát `cryptlib`u. Místo toho se používá v případě veřejných klíčů exportovaný certifikát a v případě soukromých klíčů řetězec obsahující název

keysetu[3] a název klíče (tyto dvě informace jsou odděleny dvojtečkou, případná dvojtečka v názvu se zdvojí).

5.2.4 Zpracování emailu (`mailhandle`)

Tento modul je v podstatě centrem celého systému, na které jsou napojeny skoro všechny ostatní moduly (viz schéma 5.1). Právě zde se děje zpracování emailových zpráv podle zadané politiky.

Toto mají za úkol již zmíněné funkce `ProcessOutgoingEmail` (pro odchozí zprávy), `ProcessIncomingEmail` (pro příchozí) a `ProcessConfigurationEmail` (pro konfigurační zprávy).

`ProcessOutgoingEmail` pracuje tak, že si funkcí `GetPolicy` zjistí politiku pro daného odesilatele a příjemce. Podle toho zprávu může zašifrovat a podepsat. Takto upravenou zprávu vrátí volajícímu.

U funkce `ProcessIncomingEmail` je to trochu složitější, nemůžeme úplně přesně vědět, co se se zprávou dělo. Víme ale, jaký následující krok máme provést, a to díky tomu, že při šifrování a podepisování se k výsledné zprávě přidávají hlavičky udávající obsah zprávy (zašifrovaná/podepsaná data). Dokud tedy ve zprávě nalézáme takovou hlavičku, provádíme v cyklu právě udanou operaci (dešifrování nebo ověření podpisu) – tím vznikne vždy nová zpráva s případně jinou hlavičkou.

`ProcessConfigurationEmail` slouží ověření, zda odesilatel může provádět danou konfigurační akci a případně k vytvoření odpovědi na ni. Více o vzdálené konfiguraci v části 5.3.

5.2.5 Síťový podsystém (`network`)

Toto je v podstatě pomocný modul pro některé operace se sítí. Asi nejvyužívanější zde definované funkce jsou `ReadLine` a `WriteLine`, jež jsou důležité především z důvodu „řádkové filosofie“ poštovních protokolů. Z dalších důležitých funkcní zmiňme ještě snad ty určené k připojení ke klientovi nebo k serveru.

5.2.6 Logování (`logger`)

Tato část systému slouží k zaznamenávání určitých významných událostí na nastavená místa. Konkrétně těmito místy mohou být textový soubor nebo emailová adresa.

U každého takovéhoto místa je určena minimální priorita zpráv, které akceptuje, aby se například kvůli oznámení nepodstatných maličkostí nemusela hned posílat zpráva. Pro přehled priorit viz tabulku 5.2.

	Důležitost	**Příklad**
1.	Kritická	nedostatek paměti
2.	Vysoká	pokus o podvržení podpisu
3.	Nízká	špatný formát konfigurační zprávy
4.	Informace	připojení klienta
5.	Žádná	pro ladící účely

Tabulka 5.2: Přehled priorit zpráv k zaznamenávání (řazeno od nejvyšší)

[3] Keyset je soubor pro ukládání klíčů.

5.2.7 Zpracování chyb (`error`)

V tomto modulu je vytvořen centrální systém pro správu chyb. Každá má svůj jednoznačný identifikátor, který může sloužit ke zkvalitnění hlášení o problémech. Tento systém ještě není zaveden v celé aplikaci, tedy se zatím příliš nepoužívá, proto se jím více nebudu zabývat a odkážu čtenáře na zdrojové texty a případně na další verze aplikace.

5.3 Vzdálená konfigurace

Vzdálená konfigurace je v této implementaci poměrně silně svázána s XML formátem databáze. Přináší to velkou výhodu v jednoduchosti implementace, kdy si stačí exportovat část stromu dokumentu do řetězce a tyto řetězce jednoduše porovnávat. Nevýhoda je taková, že je tímto předepsaný formát konfiguračního emailu a při změně databáze bude nutné přepracovat i celý tento systém.

5.3.1 Konfigurační zpráva

Pro konfigurační email existuje několik přesně daných záležitostí, které musí dodržet:

- Jeho předmět musí mít pevně daný obsah, a to `CONFIG EMAIL`.
- Především musí mít požadované zabezpečení, tj. být podepsán uživatelem oprávněným spravovat dané nastavení a být zašifrován veřejným klíčem proxy.
- Rovněž po odšifrování a ověření podpisu má zpráva předepsanou strukturu, ta už se ale liší podle konkrétního příkazu (viz následující část).

5.3.2 Příkazy

Existují tři příkazy, které lze vzdáleně provést, a sice read, create a modify, které jsou popsány v následujících podkapitolách.

Úroveň XML dokumentu, na které probíhá konfigurace Funkce vzdálené konfigurace z důvodu přílišné složitosti implementace neumožňuje konfigurovat všechny elementy XML dokumentu, ale je určena úroveň, na které změny probíhají. Je možné měnit jednou zprávou změnit jeden element, konkrétně jeden z těchto: `<user_setting>`, `<domain_setting>`, `<proxy_setting>`, `<user_keys>` a `<domain_keys>`.

Read

Tento příkaz slouží k přečtení specifikovaného elementu. Jako odpověď je textově exportována celá část stromu XML dokumentu. Například chceme-li přečíst nastavení určitého uživatele, funkce `ProcessConfigurationEmail` vrátí textově celý tag `<user_setting>`.

Je vhodný zejména pro spolupráci s příkazem modify, který u modifikovaného tagu vyžaduje i starou verzi.

Tělo konfiguračního emailu zde může vypadat například takto:

```
<read>
  <user>emperor.palpatin@seznam.cz</user>
</read>
```

Create

Posláním tohoto příkazu můžeme vytvořit nového uživatele, doménu nebo veřejný klíč. V případě úspěchu vrátí kladnou odpověď informující o úspěšném vytvoření.

Příklad těla následuje:

```
<create>
  <user>emperor.palpatin@seznam.cz</user>
  <user_setting>
    <author>ADMINISTRATOR_EMAIL</author>
    <name>EP</name>
    <login>emperor.palpatin</login>
    <email>emperor.palpatin@seznam.cz</email>
    ...
  </user_setting>
</create>
```

Modify

Tento příkaz je ze všech tří nejsložitější. Obsahuje identifikaci měněného tagu, jeho původní verzi a jeho požadovanou novou verzi. Informace o původní verzi zde je především z toho důvodu, že kdyby se jinak dva administrátoři zároveň pokoušeli změnit ten samý tag, mělo by to velmi nepředvídatelné důsledky – oba by si mysleli, že jejich nastavení bylo aplikováno, ovšem v jednom případě by to nebyla pravda.

Tělo konfiguračního emailu by mohlo vypadat například takto:

```
<modify>
  <user>emperor.palpatin@seznam.cz</user>
  <old>
    <user_setting>
      <author>ADMINISTRATOR_EMAIL</author>
      <name>EP</name>
      <login>emperor.palpatin</login>
      <email>emperor.palpatin@seznam.cz</email>
      ...
    </user_setting>
  </old>
  <new>
    <user_setting>
      <author>ADMINISTRATOR_EMAIL</author>
      <name>EP</name>
      <login>emperor.palpatin</login>
      <email>emperor.palpatin@seznam.cz</email>
      ...
    </user_setting>
  </new>
</modify>
```

Tímto dost dlouhým emailem jsme u tohoto uživatele změnili adresu POP3 serveru.

Tag `<author>` Každý tag nastvení, který může být měněn, ma jako svého potomka tag `<author>`. V něm musí být uvedena emailová adresa administrátora, který provedl poslední změnu.

Zálohování starých verzí konfiguračního souboru Při každé změně se XML soubor pro jistotu zálohuje.

Kapitola 6

Další vývoj

Problematika vývoje v praxi použitelného systému pro zabezpečení elektronické pošty je natolik rozsáhlá, že na některé věci v této bakalářské práci prostě a jednoduše nebyl ani nezbyl čas. Věnuji proto alespoň tuto kapitolu popisu oblastí, na kterých by bylo možné dále pracovat.

6.1 Rozvíjení možností zabezpečení

Vytvářený systém se zatím zabývá pouze zabezpečením obsahu komunikace elektronickou poštou. Jednou z oblastí, na kterých by bylo možné dále pracovat, je implementace už existujících možností zabezpečení.

Například se na to můžeme podívat z pohledu zabezpečení na různých úrovních komunikace – úroveň aplikačních protokolů nebo transportní vrstva. Jiný náhled představuje třeba hledisko zabezpečované vlastnosti – zatím se zabývám pouze obsahem elektronické pošty, ale nijak není řešeno zabezpečení přihlašování a s tím související potenciální vyzrazení uživatelského jména a hesla.

V této podkapitole se tedy pokusím shrnout oblasti, jež by mohly pomoci zvýšit úroveň zabezpečení komunikace především mezi proxy a poštovním serverem.

6.1.1 Zabezpečení protokolem TLS/SSL

Jde o zabezpečení celého přenosu dat, které se děje na transportní vrstvě. Pro více detailů viz podkapitolu 3.2.1.

6.1.2 Doplnění rozšiřujících funkcí poštovních protokolů

Poštovní protokoly za dobu své exitence prošly určitým vývojem, což s sebou přineslo mnoho změn a rozšíření. Pro účely proxy by mohly být zajímavé především rozšíření zabývající se autentizací uživatele.

Nebylo by špatné doimplementovat především podporu příkazu `AUTH(ENTICATE)` protokolů SMTP, POP3 (IMAP) (viz podkapitolu 3.1.1) a příkaz `APOP` protokolu POP3 (viz podkapitolu 3.1.2). Není to ovšem kritická záležitost, neboť podpora těchto příkazů na straně serveru není z nejrozšířenějších.

6.2 Zabezpečení konfiguračního souboru

Konfigurační soubor v tuto chvíli představuje jedno z nejzranitelnějších míst systému. Tím, že jde o obyčejný textový soubor s informacemi uloženými pomocí XML je vystaven útoku kohokoliv, kdo se dostane k počítači s běžícím proxy serverem.

Vhodným řešením by bylo například použít vhodný hardware sloužící k zabezpečení souborového systému.

Tento nedostatek by také mohl odpadnout při aplikaci návrhu v následující podkapitole...

6.3 Portování na vestavěný systém

V současné době je možné proxy provozovat jen pod systémem GNU/Linux. Několik kolegů se pokoušelo naportovat jej na systém Microsoft Windows XP a narazili na velké problémy.

Většího rozšíření a by bylo možné dosáhnout přepsáním aplikace pro vestavěný systém. Vypadalo by to tak, že proxy by nebyla jen spustitelná aplikace, ale kus hardwaru. K okolní síti by byl připojen přes síťové rozhraní, lépe přes dvě (jedno pro komunikaci s klienty a druhé s internetem – nezabezpečenou sítí).

Odpadla by tím nutnost vyhradit proxy celý funkční počítač, což by mohlo nemálo snížit náklady a přispět k dostupnosti. Rovněž by odpadlo mnoho problému s kompatibilitou, aplikaci by prostě nebylo potřeba psát pro různé operační systémy.

Z hlediska bezpečnosti je určitě pohodlnější zamezit fyzickému přístupu nepovolaných osob k malé krabičce než k celému počítači – bylo by možné ji zamknout někam do skříňky. Také by nebylo tak jednoduché ovlivnit software, který běží na vestavěném systému (oproti klasickému osobnímu počítači).

6.4 Výměna klíčů

Bylo by vhodné rozšířit systém o funkce, které by zajišťovaly automatickou výměnu veřejných klíčů mezi navzájem si důvěřujícími proxy servery. Především by se tím hodně ulehčila práce správci proxy.

Například by každá proxy při vytvoření nového uživatele rozeslala patřičně zabezpečený email s jeho veřejným klíčem všem ostatním, které by byly uvedeny v konfiguračním souboru. Také by mohla přijímat požadavky od neuvedených, kteým by klíč poslala na požádání.

6.5 Zabezpečení komunikce klient – proxy

Zranitelné místo může představovat komunikace klienta s proxy, jež s sebou nese všechna bezpečnostní rizika, kterým se chceme nasazením systému vyhnout. Zatím je předpokladem nasazení bezpečná vnitřní síť.

Tohoto může být dosaženo v podstatě jakýmkoliv způsobem, který je aplikovatelný na komunikaci proxy – server, což už bylo popsáno v podkapitole 6.1.

6.6 Automatizace některých činností

Systém je stále ve vývoji a stále zůstává mnoho prostoru, jak zpohodlnit jeho užívání. Například při vzdálené konfiguraci se musí dělat ručně i některé věci, u kterých to není potřeba, jmenujme třeba vyplňování tagu `<author>`. S tím souvisí i další podkapitola.

6.7 Grafická konfigurační aplikace

Proxy už z návrhu aplikace umožňuje definování zabezpečovacích politik na několika úrovních. Nejméně prioritní, avšak rozhodně ne nedůležitou jsou nastavení samotných uživatelů.

Bohužel asi málokterý uživatel bude schopen ručně tvořit a zabezpečovat konfigurační emaily, přesto by ale rád měl možnost definovat si svá jednoduchá pravidla zabezpečení. Z tohoto pohledu by bylo vhodné vytvořit grafickou konfigurační aplikaci, jež by uživatele odstínila od příliš technických podrobností procesu a nechala jej si veškerá nastavení „naklikat“.

Kapitola 7

Závěr

V této práci jsem se věnoval současnému stavu služby elektronické pošty a možnostem jejího zabezpečení. V závislosti na těchto poznatcích jsem navrhl a implementoval základ systému, který představuje velmi rozumnou alternativu pro zamýšlené cílové prostředí. Také jsem popsal nevýhody takového řešení a určité možnosti, jak se jim bránit. Neméně důležitou částí je i popis možností dalšího vývoje.

Povedlo se mi vytvořit základ systému vhodného k centrálnímu zabezpečení emailové komunikace větší skupiny uživatelů. Rád bych vyzdvihl především velmi komplexní systém určování dané politiky, které probíhá v několika krocích s různou úrovní důležitosti. Rovněž stojí za zmínku poměrně elegantní implementace vzdálené konfigurace – v detailech jistě ještě projde změnami, ale její filosofie se už pravděpodobně měnit nebude. Systém je v podstatě použitelný v praxi i přes jisté (prozatímní) nepohodlí a vazbu na operační systém GNU/Linux. Podrobnosti se můžete dozvědět v kapitole 6.

Proxy pro zabezpečení elektronické pošty představuje velmi komplexní systém. Svým rozsahem značně překračuje rozsah této bakalářské práce, jak může být patrné především z kapitoly o budoucím vývoji. Programová část je vyvíjena s přestávkami již několik let a tento vývoj stále trvá, proto se na první pohled může jevit jako nedokončená. I potom, co vývoj dospěje do fáze, kdy bude možné systém šířeji použít, bude vyžadovat průběžnou údržbu a případně drobné či méně drobné modifikace, například v souvislosti s možnými budoucími změnami protokolů či větší podporou některé dosud příliš nepodporované funkce.

Literatura

[1] Dave Crocker. Email history, how email was invented. http://www.livinginternet.com/e/ei.htm, 2007. [Online; navštíveno 11. 05. 2007].

[2] Van Emery. Essential internet protocols – pop3. http://www.vanemery.com/Protocols/POP/pop.html, 2007.

[3] Dalimil Hrabovský. Grafické rozhraní pro konfiguraci bezpečné pošty. Master's thesis, Vysoké učení technické v Brně, 2004.

[4] Charles M. Kozierok. Pop overview, history and standards. http://www.tcpipguide.com/free/t_POPOverviewHistoryandStandards.htm, 2007. [Online; navštíveno 11. 05. 2007].

[5] Charles M. Kozierok. Smtp overview, history and standards. http://www.tcpipguide.com/free/t_SMTPOverviewHistoryandStandards.htm, 2007. [Online; navštíveno 11. 05. 2007].

[6] Joel Snyder. How can tls increase security. http://www.networkworld.com/newsletters/gwm/0329gw1.html, 2007. [Online; navštíveno 11. 05. 2007].

[7] Internet Society. Rfc 2822 internet message format. http://tools.ietf.org/html/rfc2822, 2007. [Online; navštíveno 11. 05. 2007].

[8] Wikipedia. Extended smtp — wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Extended_SMTP&oldid=122940451, 2007. [Online; navštíveno 11. 05. 2007].

[9] Wikipedia. Smtp-auth — wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=SMTP-AUTH&oldid=128049154, 2007. [Online; navštíveno 11. 05. 2007].