Fakulta Matematiky, Fyziky a Informatiky

Univerzita Komenského

Katedra Informatiky

Balancovaný Antispamový Systém

Diplomová Práca

Autor : Marián Devečka

Vedúci : RNDr. Marián Vittek, PhD.

Bratislava

Poďakovanie

Touto cestou vyslovujem poďakovanie RNDr. Mariánovi Vittekovi, PhD. za poskytnutie nápadov, cenných rád a pripomienok pri vypracovaní mojej diplomovej práce.

Čestne vyhlasujem, že diplomovú prácu som vypracoval samostatne s použitím uvedenej literatúry.

.....

V Bratislave, máj 2009 Marián Devečka

Obsah

Kapitola 1

Úvod

Asi nikomu z nás nie je cudzie slovo spam. Každý deň sú mailové schránky po celom svete zaplňované desiatkami spamov. Zvyčajne je to vidieť vo forme narastajúceho čísla pri záložke Spam. Avšak z času na čas sa stane, že sa takýto nechcený mail dostane aj medzi normálnu poštu, alebo dokonca horšie, mail ktorý si chceme prečítať, sa dostane medzi spamy.
Prečo vôbec takéto spamy existujú a ako to, že systém zle odhaľuje takéto správy? Pre nájdenie odpovede na prvú otázku sa musíme pozrieť do minulosti, kedy ešte neexistoval internet a teda ani elektronický mail. Predtým ako vznikol internet sa písané správy šírili výlučne listovou podobou. V úplných začiatkoch túto funkciu vykonávali nositelia správ, no neskôr sa tento proces stále viac a viac rozrastal až do formy centrálnych pôšt a poštových schránok ako to poznáme i dnes. Klasická pošta sa však od elektronickej líši v najmä v jednom – nemôže si ju dovoliť každý bez obmedzenia. Kým elektronickú poštu môže odosielať každý a to v podstate ľubovoľných množstvách, klasická pošta je spoplatnená. Do klasických poštových schránok tiež dostávame reklamné spamy, avšak vieme ich jednoducho odlíšiť od „riadnej“ pošty. Základné ukazovatele sú väčšinou validná poštová známka, či rôzne iné poštové pečiatky. Bežný internetový spam sa ale navonok nijako nelíši od žiadanej elektronickej pošty. Až po hlbšom preskúmaní elektronickej správy - jej odosielateľa a obsahu, to vieme určiť. Za týmto účelom boli vyvinuté aj anti-spamové filtre, ktoré túto úlohu plnia za nás. Napriek tomu, že ich existuje niekoľko, nepostačujú na vyriešenie tohto problému. V dnešnej dobe existuje toľko spamu, že aj pri úspešnosti filtrov nad 95%, budeme dostávať do schránok takýchto správ niekoľko denne. Následkom je plytvanie internetových zdrojov, ktoré by inak mohli byť lepšie využité.

1.1 Cieľ práce

Cieľom tejto práce je načrtnúť prehľad existujúcich riešení problému spamu, ukázať ich nepostačujúcosť a preskúmať alternatívne možnosti riešenia. Jadrom práce je posúdiť nový antispamový systém, ktorý prostredníctvom vyváženej emailovej komunikácie odhaľuje spamové správy a navrhnúť a realizovať jeho experimentálnu implementáciu.

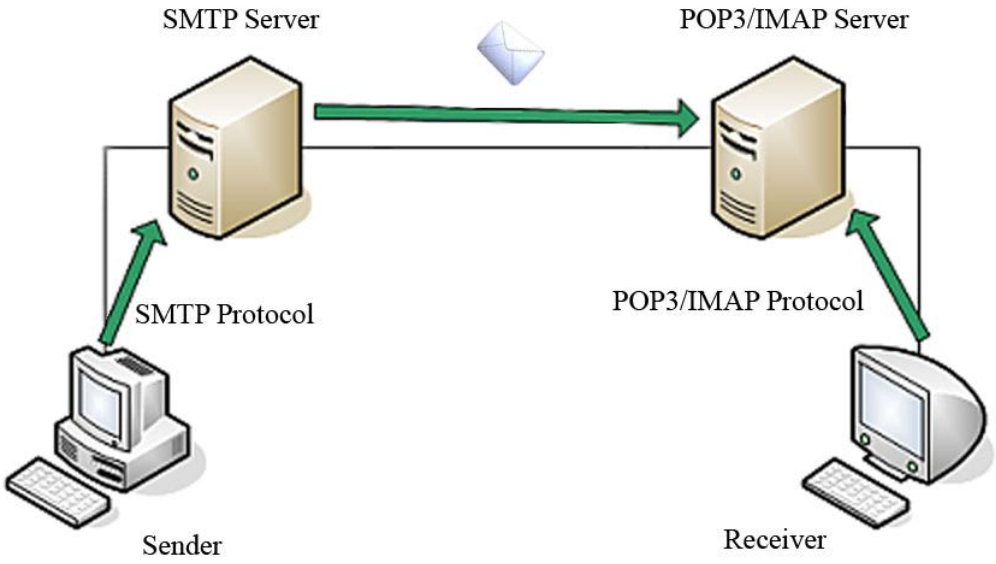
1.2 Členenie práce

Druhá kapitola vysvetľuje ako funguje elektronická pošta. Zadefinujeme tu pojem spam a popíšeme jeho negatívne vlastnosti. Ďalej rozoberieme metódy, ako proti nemu bojovať. Zhrnieme prínosy a nedostatky týchto metód a ukážeme možný priestor pre nový antispamový systém.
Tretia kapitola sa zaoberá návrhom alternatívneho antispamového systému – Balancovaného antispamového systému(BAS). Vysvetlíme tu jeho prínos a popíšeme jeho vlastnosti. Potom navrhneme naivnú implementáciu BAS, ukážeme jej nedostatky a preskúmame možnosti ďalších návrhov.
V štvrtej kapitole vyberieme jeden z týchto návrhov a naimplementujeme jeho konkrétne riešenie. Táto implementácia pozostáva z dvoch hlavných častí BAS Servera a BAS Klienta. Komunikáciu medzi nimi riadi BAS protokol, ktorý tu bude prezentovaný. Ukážeme si ako vyzerá konkrétna aplikácia a v jednoduchosti si popíšeme aj ako sa používa.
Piata kapitola je venovaná bezpečnosti BAS protokolu. Zistíme, že základný BAS protokol nie je postačujúci pre bezpečnosť systému. Využijeme preto asymetrické šifrovanie a poukážeme na bezpečnosť rozšíreného BAS protokolu.
V šiestej kapitole sa dočítame o možnostiach, ako by sa dala takáto implementácia BAS dala vylepšiť.
V záverečnej siedmej kapitole zhrnieme dosiahnuté výsledky a prínos práce.
Na záver práce je pridaný aj návod na kompiláciu zdrojového kódu a spustenie aplikácie.

Kapitola 2

Elektronická pošta, spam a metódy ochrany voči spamu

Pod pojmom elektronická pošta rozumieme posielanie správ cez internet. Laicky to môžeme popísať, ako proces prenesenia správy (spolu s jej prílohami) do schránky príjemcu po stlačení tlačítka Odoslať a proces doručenia správy do mailového klienta pri zisťovaní novej pošty. Prenos správy od odosielateľa ku príjemcovi majú na starosti sieťové protokoly SMTP, IMAP a POP3 (Obr. 2.1).



Obr. 2.1: Cesta elektronickej správy od odosielateľa ku príjemcovi.

SMTP, z anglického Simple Mail Transfer Protocol – Jednoduchý poštový prenosový protokol, je internetovým štandardom na prenos elektronických správ cez IP siete. Štandardne sa využíva sieťový TCP protokol na porte 25 alebo 587. Prostredníctvom SMTP odosielateľ zadá jedného alebo viac príjemcov, text správy, prílohy a iné objekty. Následne je daná správa prenesená do cieľovej emailovej

schránky. Tento proces sa realizuje sériou jednoduchých textových dotazov a odpovedí medzi klientom a serverom.

SMTP protokol nazývame aj „vkladacím“ protokolom, pretože slúži len na odoslanie správ a nie je možné ho použiť na výber správy zo schránky.

Jeho hlavným nedostatkom je neschopnosť autentifikácie užívateľa. Moderné rozšírenie protokolu ESMTP(Extended Simple Mail Transfer Protocol) túto funkciu už zvláda. [9, RFC2821]

```
Príklad SMTP protokolu :
S: 220 smtp.example.com ESMTP Postfix
C: HELO relay.example.org
S: 250 Hello relay.example.org, I am glad to meet you
C: MAIL FROM:<bob@example.org>
S: 250 Ok
C: RCPT TO:<alice@example.com>
S: 250 Ok
C: RCPT TO:<theboss@example.com>
S: 250 Ok
C: DATA
S: 354 End data with <CR><LF>.<CR><LF>
C: From: "Bob Example" <bob@example.org>
C: To: Alice Example <alice@example.com>
C: Cc: theboss@example.com
C: Date: Tue, 15 Jan 2008 16:02:43 -0500
C: Subject: Test message
C:
C: Hello Alice.
C: This is a test message with 5 headers and 4 lines in the body.
C: Your friend,
C: Bob
C: .
S: 250 Ok: queued as 12345
C: QUIT
S: 221 Bye
{The server closes the connection}
```

Príklad ilustruje proces odosielania správy medzi klientom C a serverom S cez protokol SMTP. Najprv sa odosielateľ prihlási na server príkazom HELO, potom označí odosielateľa príkazom MAIL FROM a následne určí dvoch príjemcov cez RCPT TO. Nakoniec pomocou príkazu DATA pošle telo samotnej správy.

Na výber správy z mailovej schránky štandardne slúžia protokoly POP3 a IMAP. Protokol POP3(Post Office Protocol version 3) je poštový protokol využívajúci TCP/IP pripojenie na porte 110. Prostredníctvom POP3 mailový klient získava elektronickú poštu z mailovej schránky. Rovnako ako pri SMTP pozostáva zo série textových dotazov a odpovedí. POP3 v zásade funguje tak, že sa napojí na server,

zistí či sú nové správy, následne ich stiahne a originály zmaže. Problém pri takomto spôsobe vznikne, ak pristupujeme k danej emailovej schránke z viacerých počítačov. [9, RFC1939]

Protokol IMAP(Internet Message Access Protocol) je podobne ako POP3 poštový protokol na získavanie pošty cez TCP/IP na porte 143. Rozdielom je, že prevzaté správy nemaže zo servera a podporuje rovnako nepripojený režim(offline mode) ako aj režim s pripojeným internetom(online mode). IMAP preto nemá problém s viacerými používateľmi tej istej mailovej schránky. Nevýhodou je väčšia zložitosť, ktorá si vyberá daň na výkone servera a rýchlosti komunikácie. IMAP je oproti POP3 aj menej rozšírený. [9, RFC2060]

Na podporu bezpečnej komunikácie prostredníctvom SSL(Secure Sockets Layer) a TLS(Transport Layer Security) [RFC2246], existujú príslušné rozšírenia protokolov SMTP, POP3 a IMAP.

```
Príklad POP3 protokolu :
S: <wait for connection on TCP port 110>
C: <open connection>
S:    +OK POP3 server ready <1896.697170952@dbc.mtview.ca.us>
C:    APOP mrose c4c9334bac560ecc979e58001b3e22fb
S:    +OK mrose's maildrop has 2 messages (320 octets)
C:    STAT
S:    +OK 2 320
C:    LIST
S:    +OK 2 messages (320 octets)
S:    1 120
S:    2 200
S:    .
C:    RETR 1
S:    +OK 120 octets
S:    <the POP3 server sends message 1>
S:    .
C:    DELE 1
S:    +OK message 1 deleted
C:    RETR 2
S:    +OK 200 octets
S:    <the POP3 server sends message 2>
C:    QUIT
S:    +OK dewey POP3 server signing off (maildrop empty)
C:  <close connection>
S:  <wait for next connection>
```

V tomto príklade sa klient C autentifikuje do svojej mailovej schránky na serveri S príkazom APOP a zistí jej stav cez STAT a LIST. Následne vyberie dve správy cez RETR a prvú z nich pritom zmaže príkazom DELE.

2.1 Definícia Spamu

Spam je nevyžiadaná a nechcená hromadne rozosielaná správa prakticky rovnakého obsahu, zväčša používaná ako reklama. Na spamovanie sa využívajú elektronické médiá ako e-mail, instantné zasielanie správ (ICQ), skupiny Usenet, krátke textové správy a podobne.

E-mailový spam definujeme ako posielanie nevyžiadaných e-mailových správ, často s komerčným obsahom, veľkým množstvám príjemcov. Zvykne sa používať aj skratka UBE (unsolicited bulk email), alebo UCE (unsolicited commercial email). Spam posielaný e-mailom začal byť problémom od polovice 90. rokov, keď sa Internet otvoril pre širokú verejnosť. Od tej doby jeho množstvo exponenciálne rastie a dnes tvorí od 82% do 87% všetkých e-mailov na svete. [10] Tlak na postavenie spamu mimo zákon bol v niektorých krajinách úspešný, v iných nie. Toto spameri využívajú a často presúvajú svoje aktivity do krajín, kde sa nedostanú do problémov zo zákonmi. [9]

E-mail je extrémne lacné masové médium a profesionálny spameri svoje procesy do detailov automatizovali. Z tohto dôvodu môže byť spamovanie veľmi ziskové napriek tomu, že pomer množstva poslaných e-mailov k reakciám je považovaný za extrémne nízky. [9]

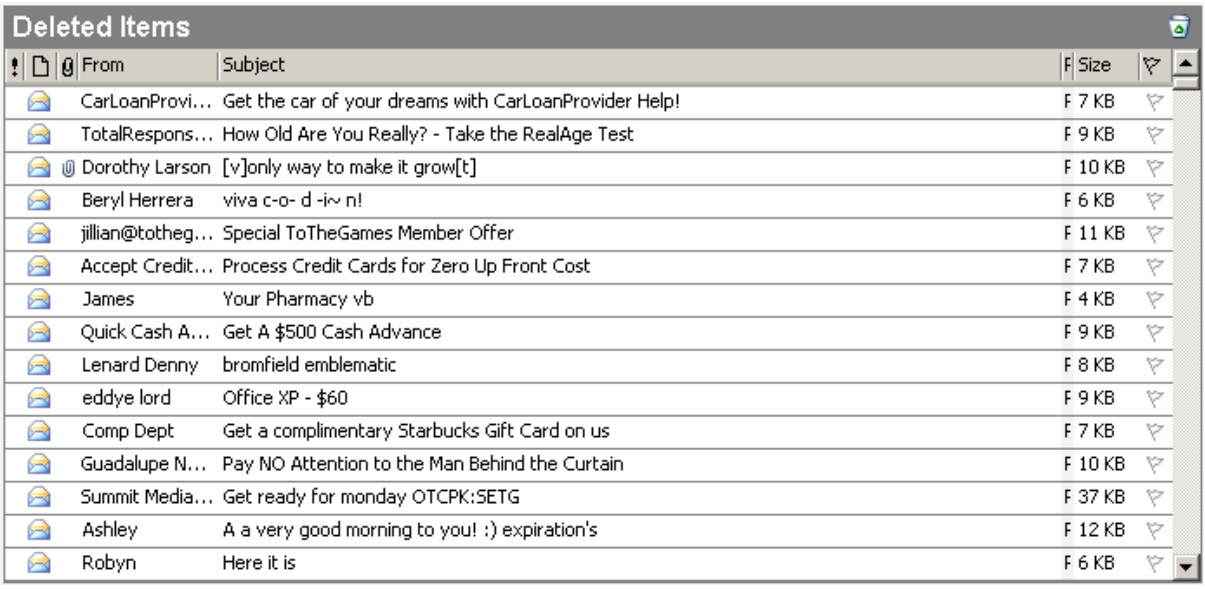
Podľa spôsobu posielania zvykneme rozlišovať dva druhy E-mailového spamu. Prvým je poslanie správy do množstva diskusných skupín - Fórumov. Tieto správy sú väčšinou zamerané na distribúciu materiálu propagujúceho sex alebo komerčné produkty, napr. z oblasti farmácie. Cieľovou skupinou sú ľudia, ktorí často čítajú elektronickú poštu, ale neposkytujú na webe svoje emailové adresy. Druhým je emailový spam, zaslaný konkrétnemu človeku priamo na jeho emailovú adresu. Emailové adresy sa získavajú najmä prehľadávaním diskusných skupín alebo webových stránok. Na tento účel používajú spameri automatizované roboty. Iným spôsobom ako sa získavajú adresy sú webové formuláre, ktoré za ich vyplnenie ponúkajú na prvý pohľad ponúkajú zaujímavú protihodnotu. [5, 6, 7, 9]

Existujú dva podstatné dôvody, prečo bojovať proti spamu. Prvým je dopad na bežných emailových a internetových používateľov. Spam spôsobuje plytvanie internetových zdrojov, zahlcuje diskové kapacity a triedenie legitímnych správ od spamu je pre používateľa časovo-náročný proces. Druhým dôvodom je nebezpečenstvo skryté v obsahu správy. Obsah spamovej správy môže byť často

nevhodný pre mladistvých a odoslanie súkromných údajov alebo zakúpenie produktov propagovaných v spame môže mať nedozierne následky. [7]

2.2 Pôvod slova

Nie je úplne jasné odkiaľ tento pojem pochádza. Mnohý veria, že slovo spam má korene v tzv. MUDoch z 80. rokov. MUD(Multiuser Dungeon) je virtuálna onlineová textovo založená hra pre viacerých hráčov. V dnešnej dobe už prakticky neexistujú lebo ich nahradili graficky vyspelejšie MMORPG (Massive-Multiplayer Online Role-Playing Game). V MUDoch sa používal pojem „spamovanie“ s významom zahlcovania počítača informáciami. [6, 7]
Iný sa domnievajú, že slovo spam bolo prepožičané zo seriálu Monty Python(1970), kde v jednej scéne prehlušuje zbor Vikingov konverzáciu postáv piesňou opakujúcou „SPAM, SPAM, SPAM,...“. Podľa nich si až neskôr MUDy prevzali tento pojem. Jedno zo zábavnejších vysvetlení tvrdí, že spam bol pomenovaný podľa konzervy mäsa, ktorá sa vyrába od 30. rokov 20 storočia dodnes. Pri rozsekaní mäsa sa všetko zašpiní a nadobudne čudný zápach. To ilustruje pocit mnohých ľudí keď do mailovej schránky dostanú spam. [6, 7, 9]



Obr. 2.2: Príklad emailovej schránky obsahujúcej spam

2.3 Ochrana

V tejto časti si popíšeme niekoľko spôsobov ako sa dá chrániť pred spamom. Najväčším problémom ochrany proti spamu je, že neexistuje ostrá hranica medzi spamom a užitočnými emailami a neexistujú ani príznaky, ktoré by jednoznačne indikovali, že sa jedná o spam. Následkom toho nie je možné vytvoriť prostriedok, ktorý by na sto percent likvidoval spam a na sto percent prepúšťal chcené emaily. [5, 6]

Ako teda proti nemu bojovať?

Hlavným prostriedok boja proti spamu sú Anti-spamové filtre. Anti-spamové filtre predstavujú automatizované nástroje, či už na strane samotného mailového klienta alebo mailového servera, ktoré oddeľujú spamové správy od legitímnych. Podľa toho akým spôsobom fungujú ich môžeme rozdeliť do nasledujúcich skupín :

2.3.1 Primitívna analýza jazyka

Primitívna analýza jazyka je procesom pri ktorom sa v elektronickej správe hľadajú určité frázy. Prvé spamové filtre boli skutočne veľmi primitívne a neboli vlastne ani pravými filtrami. V dobe, keď sa používala táto metóda(90. roky), spameri nepoznali dnešné „špinavé“ triky ako obísť takýto filter. Zvyčajne sa teda jednalo o hľadanie fráz typu „Call Now!“ alebo „Free Trial“. Filtrovanie na základe jediného slova mohlo mať vtedy úspešnosť až 80%. Časom však stále pribúdali nové a nové frázy, ktoré bolo treba manuálne pridávať do filtra. S narastajúcim počtom slov zároveň stúpala aj šanca o nesprávnu identifikáciu. Niektoré frázy navyše nejednoznačne určujú povahu správy. Napríklad správa s pozvaním na konferenciu je pre niekoho spamom a pre iného žiadaná pošta. [6]

Pozitíva : *Jediné riešenie v danom období.*

Negatíva : *Vyžaduje veľa údržby, malá presnosť, vysoká chybovosť.*

Použitie : *Nepoužívané*

2.3.2 Blacklisting

Blacklist, v preklade „čierna listina“, je zoznam emailových adries a Internetových domén, z ktorých ľubovoľná odoslaná správa je považovaná

za spam. Koncom 90. rokov vznikali tzv. RBL – Real-time blackhole-lists, bezplatné servere, ktoré plnili funkciu Blacklistu pre komunitu ľudí. Ak dostal takýto server opakovane sťažnosť na určitú doménu alebo server tak ho pridal do svojho Blacklistu. Každý Blackhole-list mal iné kritéria, či už prísnejšie alebo menej, ľubovoľný užívateľ mal k dispozícii tieto informácie a mohol si na ich základe prispôsobiť svoj spam filter podľa ľubovôle. MAPS RBL – Mail-Abuse Prevention System Real-time blackhole lists, je špeciálnou formou RBL, ktorá pre jej užívateľov blokuje správy priamo na sieti. Výhodou takejto metódy je, že sa šetria sieťové zdroje a spamovacia sieť akoby zmizne zo sveta, až dokiaľ sa problém neodstráni. Slabou stránkou Blackhole-listov je doba spracovania, čo znamená, že existuje nemalé omeškanie medzi pridaním siete na čiernu listinu a začiatkom rozosielania spamových správ. Spočiatku to nebol problém, lebo spameri boli dosť nemobilní, ale v dnešnej dobe ukradnutých dial-up účtov, obetných spam-strojov a zle nakonfigurovaných serverov, sa noví spameri môžu objaviť a stratiť skôr než sú pridaní do Blackhole-listu. A to tiež súvisí aj s problémom údržby. Označené siete predsa raz môžu zmeniť majiteľa, ktorý už nemá v úmysle spamovať. [5, 6]

2.3.3 Heuristické Filtrovanie

Heuristické filtrovanie je pokročilou formou primitívnej analýzy jazyka, kedy sada pravidiel určuje, či je mail spam a zároveň iná sada určuje legitimitu správy. Znamená to, že niekedy je správa, ktorá má určité charakteristiky spamu ohodnotená ako legitímna ak má aj požadované vlastnosti nespamovej správy.

Brightmail

Jedným z prvých heuristických filtrov bol komerčný Brightmail. Išlo o server z databázou pravidiel, kde experti alebo „spam majstri“ vyberali charakteristické črty z kolekcie spamových správ.

SpamAssasin

Neskôr vznikli aj prvé nekomerčné heuristické filtre, medzi ktoré patril aj SpamAssasin [11]. SpamAssasin prišiel s novým prvkom ohodnocovania – každé pravidlo bolo ohodnotené bodmi. Tieto body sa potom sčítali a ak

celkový počet bodov prekročil istú hranicu, správa bola označená za spam. SpamAssasin navyše odhaľoval aj niekoľko trikov, ktorými spameri maskovali svoju identitu a využíval aj blacklisty. Hlavnou nevýhodou SpamAssasina bolo, že spameri si ho mohli stiahnuť a prispôsobiť svoje spamové správy tak aby ním boli akceptované. Správcovia síce následne upravili pravidlá, na čo spameri znovu vylepšili svoje správy. V dnešnej dobe má SpamAssasin vyše 900 pravidiel a aby ostali efektívne, treba ich upravovať približne každých 6 mesiacov. Bodové ohodnocovanie tiež nie je ideálne pre každého. Body sú totiž pridelené podľa vôle tvorcu pravidla a nie sú založené na štatistike alebo matematických rovniciach. [6]

Pozitíva : *Oveľa presnejšie ako primitívne filtrovanie, pravidlá jednoducho distribuovateľné*

Negatíva : *Pravidlá stále vyžadujú údržbu, menšia presnosť oproti štatistickým filtrom, spameri môžu zneužiť filter nato, aby ho obišli*

Použitie : *Systémoví administrátori, ktorí si môžu dovoliť tolerovať chybu 5% a viac s premenlivou presnosťou*

2.3.4 Bayesiánske filtrovanie

Je technológia, ktorá umožňuje spamovému filtru „naučiť“ sa rozpoznávať spamové správy na základe štatistickej analýzy dĺžky správy a distribúcie slov nachádzajúcich sa v správe. V zásade treba Bayesiánskemu filtru podsunúť niekoľko správ a určiť povahu jednotlivých správ – či sú spam alebo nie. Filter sa potom sám naučí rozpoznať správy podobného zloženia. Aby však dosahoval dostatočne dobrú presnosť(nad 90%) pre nový druh spamu, treba takýchto správ určiť aspoň 100. V súčasnej dobe mnoho heuristických filtrov využíva práve Bayesiánske filtrovanie. [5, 6]

2.3.5 Whitelisting

Úplný opak Blacklistov je Whitelist. Whitelist, alebo „biela listina“ je zoznam mailových adries a domén z ktorých správy akceptujeme. Všetko ostatné je považované za spam. Zvyčajne si ho každý užívateľ udržuje sám. Mnohí tvrdia, že táto metóda má 100% úspešnosť pri odhaľovaní spamu, avšak

môže sa stať, že správa od neznámeho užívateľa ostane nepovšimnutá[1]. Neexistuje tu žiadne ohodnocovanie a preto považujeme metódu Whitelist za „čierno-bielu“. Takýto systém môžu spameri navyše aj oklamať falšovaním svojej identity. Stačí si len vymyslieť adresu, ktorá je natoľko bežná, že ju má väčšina užívateľov vo whiteliste(napríklad support@mcafee.com). Prijatie správy prebehne bez autentifikácie odosielateľa a spamer môže posielať toľko spamu koľko sa mu len zachce. Ak sa takáto situácia vyskytne, užívateľ sa dostáva do patovej situácie. Takúto adresu potom nemôže odstrániť zo svojho whitelistu, pretože by nedostával ani chcené správy od skutočného odosielateľa. [5, 6]

Pozitíva : *Veľmi presná metóda, nezaložená na obsahu správy*

Negatíva : *Dá sa oklamať, všetci užívatelia sú akceptovaní, vyžaduje ručnú úpravu*

Použitie : *Užívatelia, ktorí požadujú vysokú presnosť a nevadí im ručná úprava, či občasná strata emailu.*

2.3.6 Challenge/Response

Challenge/Response(C/R) – Výzva a Odpoveď je metóda podobná Whitelistu avšak omnoho náročnejšia na správu. C/R pridáva bremeno správy whitelistu na chrbát odosielateľa. V praxi to vyzerá tak, že pri pokuse o odoslanie správy nám príde správa s výzvou na odoslanie. Odpoveď na výzvu odosielanú správu podpíše a odošle. Príjemca správy potom už vie rozpoznať, či sa jedná o podpísanú správu alebo spam. Pre mnohých ľudí je tento proces neakceptovateľný a nebudú ochotní komunikovať s užívateľmi, ktorý vyžadujú odpoveď na mailovú výzvu. Obhajcovia C/R poukazujú na minimálnu mieru nesprávne označených spamov a veľkú presnosť. Podobne ako pri Whitelistoch tu však existuje možnosť sfalšovať identitu správy. Nevýhodou je aj zvýšené množstvo sieťovej komunikácie a užívatelia sa sťažujú, že im C/R značne spomaľuje proces odosielania správ. [6]

Pozitíva : *Veľmi presná metóda, nezaložená na obsahu správy.*

[1] nový kontakt alebo člen newsgroupy

Negatíva : *Môže spôsobiť nevôľu posielať maily užívateľovi. Spomaľuje emailovú komunikáciu a zahlcuje siete. Dá sa oklamať, vyžaduje ručnú úpravu.*

Použitie : *Užívatelia, ktorí požadujú verifikáciu od odosielateľa pred tým, než začnú komunikovať a nevadí im občasná strata emailu, alebo obmedzený počet ľudí ktorí s nimi budú ochotní komunikovať.*

2.3.7 Kolaboračné Filtrovanie

Pod kolaboračným filtrovaním rozumieme ľubovoľný spôsob fitrovania, pri ktorom je využívaná spoločná inteligencia siete užívateľov na odhaľovanie spamu. Kolaboračné siete zvyčajne ťažia z nešťastia jednotlivcov, ktorý dostanú spam nato, aby vybudovali ešte lepší filtrovací systém. V zásade sa jedná o identifikovanie spamu, následné zozbieranie kľúčových informácii o správe(obsah, štruktúra, ...) a jej zdieľanie medzi skupinou užívateľov alebo filtrov. Slabinou kolaboračného filtrovania je práve nutnosť prispievania jednotlivcov komunite. Pre rozsiahle komunity manuálne riadenie znamená zložitú správu a propagáciu. V prípade, ak je tento proces automatizovaný, tu vzniká šanca falošného poplachu či infiltrácie záškodníkom. [6]

Pozitíva : *Proaktívna ochrana proti novým typom spamu.*

Negatíva : *Musí sa dohliadať na spoľahlivosť. Zdĺhavá propagácia.*

Použitie : *Dodatočná vrstva ochrany medzi filtrami.*

Iné metódy boja proti spamu :

2.3.8 Throttling

Predstavuje techniku, ktorá spomaľuje sieťovú komunikáciu pri posielaní. Filozofiou je, že na posielanie legálnych správ stačí určitý limitovaný počet zdrojov pre ľubovoľnú sieť. Oprávnený odosielateľ môže posielať aj väčšie množstvo správ, zvyčajne však smerujú do rôznych sietí. Spamer na druhej strane, často zahlcuje jednu sieť skúšajúc všetky možné kombinácie emailových adries slovníkovým útokom. Takáto metóda má výhodu v tom, že vždy prepustí všetky legitímne správy. Pre spamera sú čas a dĺžka

spracovania natoľko dôležité, že po útoku na takýto „spomalený“ server pravdepodobne stratí záujem. [6]

TaxProxy

Jedným z programov využívajúcich Throttling je aj TaxProxy. TaxProxy dokáže zmiasť spamera-útočníka, zahltiť ho naspäť informáciami a zdržať. [6]

Pozitíva : *Napomáha ku chráneniu zdrojov.*

Negatíva : *Nie je riešením na spam a môže spôsobiť problémy legitímnym odosielateľom.*

Použitie : *Poskytovatelia služieb a veľké spoločnosti ktoré chcú šetriť zdrojmi.*

2.3.9 Nové štandardy

Za posledných pár rokov sa objavilo niekoľko návrhov na zlepšenie alebo pozmenenie klasického SMTP protokolu. SMTP bol totiž navrhnutý tak aby fungoval anonymne a zaručoval súkromie používateľov Internetu. A práve spameri využili túto skutočnosť na odosielanie anonymných správ.

Autentifikované SMTP

Spočiatku sa myslelo, že autentifikované SMTP bude odpoveďou na spam, avšak ukázalo sa, že je užitočné iba na identifikovanie legitímnych odosielateľov v systéme. Autentifikované SMTP vyžaduje od odosielateľa, aby zadal heslo skôr, než môže odoslať email. Mnoho klientov a ani serverov však nepoužíva autentifikáciu. Navyše spamerom nič nebráni, aby odosielali mail z vlastného mailového servera. Centrálna autorita by mohla vynútiť autentifikáciu v globálnom meradle, ale množstvo ľudí by určite nesúhlasilo zo zmenou konceptu anonymity na internete a možnosti vystopovať odosielateľa ľubovoľnej správy. [6]

Sender Policy Framework

Jeden z novších návrhov boja proti spamu je Sender Policy Framework (SPF).

SPF rozširuje DNS[1] a využíva ho na identifikáciu užívateľov ktorí sú oprávnení na odosielanie správ. Ostatní sú vydaní napospas spam filtrom. SPF sám o sebe nezabraňuje spamu, avšak bráni falšovaniu identity. Aby však SPF fungovalo za ľubovoľných okolností, je potreba zmeniť SMTP štandard. V niektorých situáciach by takéto riešenie zatiaľ neuspelo. Predstavme si napríklad, že sme na dovolenke a snažíme sa odoslať mail z tamojšieho hotela. Je možné, že SPF by našu žiadosť zavrhol, pretože internetová doména hotela by sa javila ako nedôveryhodná. [6]

2.3.10 Obfuskácia adries

Jednou z noriem ako bojovať proti spamu sa stala obfuskácia emailovej adresy za účelom skryť ju pred spamermi. Spameri totiž využívajú tzv. zberných robotov na prehľadávanie stránok a zbieranie emailových adries. Predpoklad je, že ak zamaskujeme adresu *petra.fialova@kvet.sk* napríklad na *petra(dot)fialova(at)kvet(dot)sk* bežný človek ju bude schopný previesť do normálnej formy avšak automatizovaný robot zlyhá. Nanešťastie koncept obfuskácie adries nefunguje až tak dobre ako si ľudia myslia. Zberní roboti sú stále viac a viac prefíkaní a dokážu znovu poskladať takto zmenené adresy. Spameri okrem iného získavajú adresy aj z iných miest ako len Web. Napríklad poskytovatelia internetových služieb alebo kredit-kartové spoločnosti predávajú emailové adresy spamerom alebo s nimi uzatvárajú reklamné kontrakty. [6]

Pozitíva : *Napomáha vyhnúť sa niektorým automatizovaným robotom.*

Negatíva : *Je skôr ústupková metóda, nenapomáha riešiť spam.*

Použitie : *Zásadoví bojovníci proti spamu.*

2.3.11 Súdna Žaloba

Niekoľkí sa pokúšajú bojovať proti spamu súdnou cestou. Unspam [13] je konzultačná spoločnosť, ktorá sa zaoberá kreatívnymi riešeniami na problém

[1] Domain Name System – je zdieľaný systém, ktorý umožňuje podľa mena identifikovať IP adresu počítača a naopak

spamu. Nie je dôležité, či spor vyhrajú alebo prehrajú, stačí, že súdne výdavky sú dosť vysoké nato, aby spamerov vyradili z biznisu. Spôsobov ako zažalovať spamerov je nemálo, ako však odhaliť ich identitu? Jedným z nápadov je ustanoviť licenciu priamo na webovej stránke (napríklad v komentároch HTML, ktoré sú pre človeka neviditeľné), ktorá zakazuje zberným robotom a iným automatizovaným pomôckam spracovávať údaje na danej webovej stránke. Ak potom zistíme, že email ktorý máme na takejto stránke bol zneužitý na spam, vieme sa dopátrať ku konkrétnemu robotovi a tým aj často odhaliť konkrétneho spamera. Túto techniku s veľkým úspechom využíva Projekt Medový Hrniec (Project Honey Pot [12]). [6]
Zisťovanie odtlačkov spamerov (Spammer Fingerprinting) je ďalším spôsobom ako sa dopátrať k spamerovi. Štruktúra, použité slová, lexikálne konštrukcie sú charakteristiky, ktoré umožňujú identifikovať správy pochádzajúce z podobného zdroja. SpammerPrinting odchytáva správy z rôznych zdrojov a snaží sa ich na základe tejto metódy roztriediť a následne sa tak dopracovať k skutočnému zdroju. [6]
Habeas Haiku bol jedným z menej úspešných pokusov ako zabrániť spamu. Jednalo sa o niekoľko riadkov copyrightu intelektuálneho vlastníctva, ktoré si mohol autor pridať do odoslanej správy. Na základe týchto riadkov bol potom odosielateľ považovaný za legitímneho. Ak by si však takýto copyright pridal do mailu aj spamer, bolo by ho možné na základe jeho obsahu súdiť. Spamerov to však vôbec neodradilo a tieto riadky smelo pridávali do svojich správ. Paradoxne dnešné antispamové filtre využívajú prítomnosť Habeas Haiku na detekciu spamu. [6]

Zdá sa, že problém väčšiny filtrov je buď v zložitosti údržby alebo neefektivite riešenia. Navyše všetky anti-spamové filtre, ktoré pracujú na základe obsahu správy (s výnimkou Bayesiánskeho filtra), sú náchylné na adaptáciu spamu voči jeho statickým pravidlám. Koniec koncov je to len boj medzi šikovnosťou spamera a anti-spamového filtra. V súčasnosti je možné v elektronickej pošte posielať aj obrázky. Spameri môžu využiť práve obrázky ako médium na utajené spamovanie. Dnešné filtre nemajú prakticky žiadnu šancu rozlíšiť, či je obrázok kresleným vtipom od kamaráta alebo reklamou na faramaceutiká.

Kapitola 3

Balancovaný Antispamový Systém

Balancovaný Antispamový Systém(BAS) prichádza s myšlienkou využiť nevýhodu spamových správ proti spamerom. Pravdepodobne najhoršou vlastnosťou spamu je jeho množstvo. Ako však využiť túto nevýhodu vo vlastný prospech?
BAS nato využíva technológiu ohodnocovania mailov kreditmi. Každý mail v takomto systéme predstavuje 1 Kredit. Keď odosielateľ odošle správu, stráca s ňou aj tento kredit. Príjemca tejto správy ho naopak získava. Kredit je vzácna jednotka, ktorej množstvo je obmedzené. Najjednoduchšie si to asi predstavíme ako posielanie správy so známkou v hodnote 1 cent, ktorú je možné znova použiť. Každý užívateľ by si na začiatku kúpil určité množstvo kreditov. Predpokladom je, že bežný užívateľ neodosiela viacej mailov, ako ich dostáva a preto sa mu kredity časom neminú. Klasický spamer však odosiela omnoho viac správ, ako ich dostáva. V systéme BAS si to však nebude môcť beztrestne dovoliť, pretože sa mu budú míňať kredity. Koniec koncov ak nám do schránky predsa len prídu nejaké spamy, každý z nich bude aj patrične ohodnotený vo forme kreditov.
Aby takýto systém však fungoval musí existovať Centrálna entita BAS server, na ktorom má každý užívateľ konto s kreditmi a realizuje proces ich výmeny. Každé konto sa vzťahuje na konkrétnu mailovú adresu. Za spamovú správu sa považuje ľubovoľná správa, ktorú nepotvrdil BAS server. Server nepotvrdí správu ak k nej nie je priradený kredit, teda v prípade, že užívateľ už nemá kredity, alebo jeho konto v rámci BAS systému neexistuje. Keďže sa jedná o citlivý systém musí tu byť zaručená aj bezpečnosť a nemožnosť oklamania alebo zneužitia.
BAS predstavuje riešenie na minimalizáciu spamu, pričom klasické filtre môžu stále slúžiť svojmu účelu. Ich údržba pri malom počte spamu bude omnoho jednoduchšia. Balancovaný Antispamový Systém je nezávislý od odosiela a obsahu správy a prepúšťa všetky legitímne správy. Odosielateľ legitímnych správ nemôže falšovať svoju identitu a ako si neskôr ukážeme, je možné jednoducho zaručiť aj autentickosť správy(prijatá správa je rovnaká ako odoslaná).

3.1 Naivná Implementácia

Predstavme si naivný spôsob implementácie takéhoto systému. Na uchovanie stavu kreditov a užívateľov je nevyhnutný BAS server. Tento BAS server udržiava informácie o kontách užívateľov. Užívateľské konto tvorí dvojica (Email, Kredit). Podstatnou informáciou je aj zoznam odoslaných správ. Správa je trojicou (Odosielateľ, Príjemca, Známka).
BAS klientom by mohla byť napríklad webová stránka. Po prihlásení sa by užívateľ zadal príjemcu, potom by si nechal systémom vygenerovať známku a odoslal by to BAS serveru. Pod známkou si môžeme predstaviť náhodný reťazec zložený z písmen a číslic, povedzme o dĺžke 16 znakov. Túto istú známku by potom skopíroval aj do mailu ktorý chce odoslať.
Príjemca správy by sa po jej obdržaní tiež prihlásil do webového BAS klienta. Tu by sa mu zobrazil odosielateľ správy a textové pole pre vyplnenie známky. Z prijatého mailu by sem potom skopíroval známku na overenie. Prijatie je úspešné ak sa známka zhoduje z odoslanou. Klient je oboznámený z validitou známky, podľa čoho určí či je správa spamom.
Presúvanie kreditov by sa realizovalo v dvoch krokoch. V prvom kroku by sa po vygenerovaní známky a označení správy na odoslanie v BAS klientovi odpočítal 1 kredit. V prípade, že by mal na konte 0 kreditov, nemohol by označiť správu na odoslanie. V druhom kroku po úspešnom potvrdení správy klientom by sa príjemcovi pripočítal 1 kredit. Ako je vidieť, počet kreditov v systéme po potvrdení ostáva nezmenený. Čo by sa stalo ak by klient správu nepotvrdil alebo „čo sa stane s visiacim kreditom“ rieši Kapitola 6.
Je zrejmé, že takýto spôsob implementácie má dve hlavné nevýhody :

- Je nepraktický pre používateľa – nutnosť duplicitne zadávať odosielateľa v BAS klientovi, generovanie známky a jej kopírovanie do odoslanej správy, či test známky odosielanej správy sú zdĺhavé operácie, znepríjemňujú a predlžujú odosielanie aj prijímanie správy.
- Nemusí byť zachovaná autentickosť správy - ak sa po ceste poruší obsah správy(napr. cieleným útokom) ale známka ostane neporušená, užívateľ úspešne potvrdí prijatie a prebehne presun kreditu.

3.2 Autenticita

Nato, aby sme docielili zachovanie autenticity, museli by sme generovať známku na základe jej obsahu. V rámci tohto modelu by to znamenalo, že by sme pred odosielaním museli skopírovať obsah správy do klienta, ktorý by následne na jeho základe vytvoril známku(napríklad Hashovacou funkciou). Potom by nebolo ani nutné vygenerovanú známku pripájať do správy, pretože správa sama je známkou. Príjemca by obsah prijatej správy skopíroval do klienta a na jeho základe by sa vyhodnotila jej validita. Pri poškodení správy by sa vygenerovala iná známka a správa by bola zaradená medzi spam. Takto vieme zaručiť autenticitu ale vôbec sme si nepomohli v jednoduchosti – stále je nutné duplicitne vypĺňať alebo kopírovať údaje.

3.3 Ďalšie uvažované možnosti implementácie :

a) Na úrovni SMTP a POP3/IMAP
b) Na aplikačnej úrovni – jadro systému
c) Na aplikačnej úrovni – mailový klient

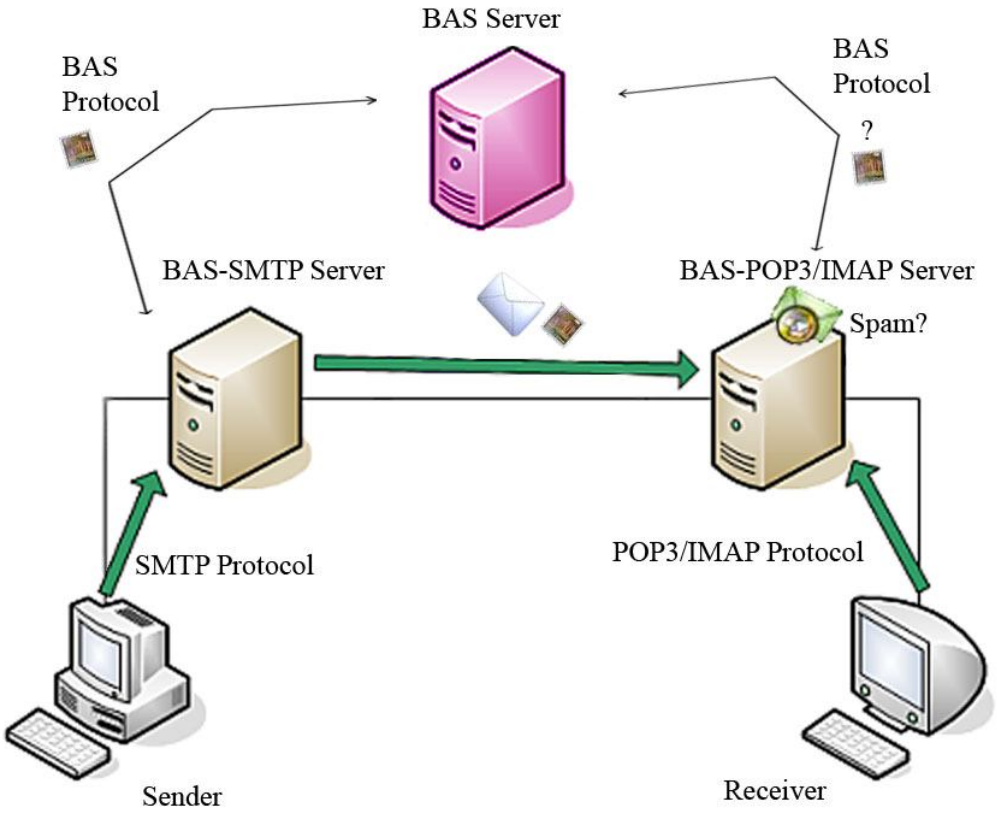
Spoločným znakom týchto metód je, že sú pohodlné pre užívateľa. Znamená to, že klient nemusí dvakrát písať to isté, stačí len, že sa prihlási na BAS server, zvyšok prebieha už automaticky.

3.3.1 Implementácia na úrovni SMTP a POP3/IMAP (Obr. 3.1)

V tomto prípade by bolo nutné naimplementovať vlastný SMTP server, POP3/IMAP server a BAS server. Pomocou protokolov ESMTP, POP3/IMAP využívajúcich TLS alebo SSL je možné urobiť bezpečnú autentifikáciu využitím loginu[1] a hesla. Tieto informácie by slúžili aj na autentifikáciu na BAS server. Náš SMTP server by pri posielaní správy poslal na BAS server známku s odtlačkom odosielanej správy. POP3/IMAP server po obdržaní

[1] emailovej adresy

správy odošle BAS serveru známku na kontrolu. Ak kontrola neprebehne úspešne, zaradí poštu medzi spam alebo ju rovno odstráni. Presun kreditov by fungoval rovnako ako v predošlom príklade. Nevýhodou je, že pre ľubovoľný iný SMTP alebo POP3/IMAP server táto metóda fungovať nebude. Nebude fungovať ani v prípade ak mailový klient nepodporuje bezpečnú autentifikáciu.



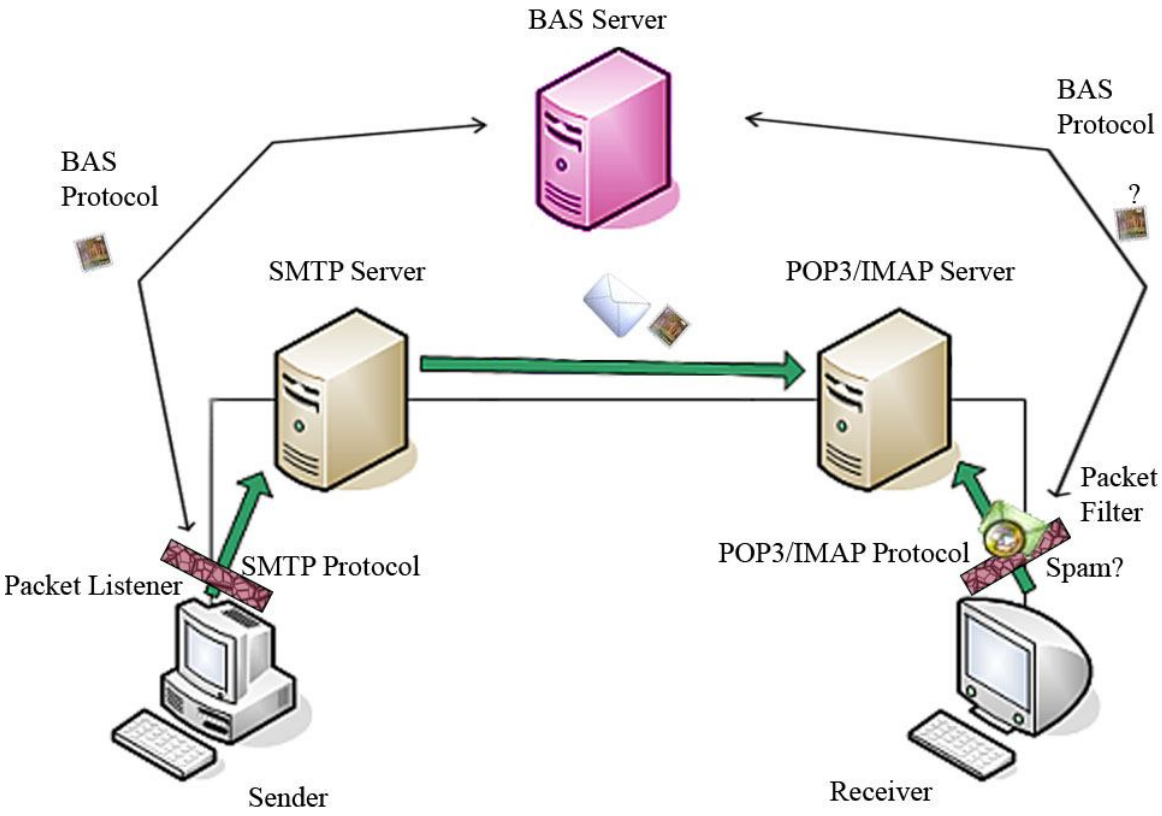
Obr. 3.1: Komunikácia v implementácii na úrovni mailových serverov

3.3.2 **Implementácia na aplikačnej úrovni – jadro systému** (Obr. 3.2)

Súčasťou tejto implementácie je BAS server a BAS klient, nachádzajúci sa na jednotlivých počítačoch.

Jedná sa o akési exotické riešenie spočívajúce v odchytávaní paketov. Pri snahe odoslať mail, by náš klient odchytil príslušný paket a odoslal známku na BAS server. Pri prijatí mailu, teda odchytení prichádzajúceho paketu zo správou, by klient skontroloval známku a v prípade nezhody mail zadržal. Autentifikáciu na BAS server by zabezpečila lokálna aplikácia v ktorej by

užívateľ vyplnil meno a heslo. Táto implementačná metóda má však niekoľko nevýhod. Jednou z nich je, že mail sa zadržiava až potom čo sa ho snažíme získať z mailovej schránky. V praxi to znamená, že ak nám príde spam do našej schránky, mail klient nám oznámi prítomnosť novej správy. Následne až pri pokuse stiahnuť si ju zistíme, že sa jedná o spam. Omnoho väčší problém by však nastal ak by sa použila ľubovoľná šifrovaná komunikácia. Pri šifrovanej komunikácii by sa nemuselo podariť rozpoznať príslušný mailový paket a naviac by bolo nemožné z neho dostať akúkoľvek informáciu. Bez informácii akými sú odosielateľ alebo príjemca, nie je možné urobiť overenie známky či prevod kreditov.

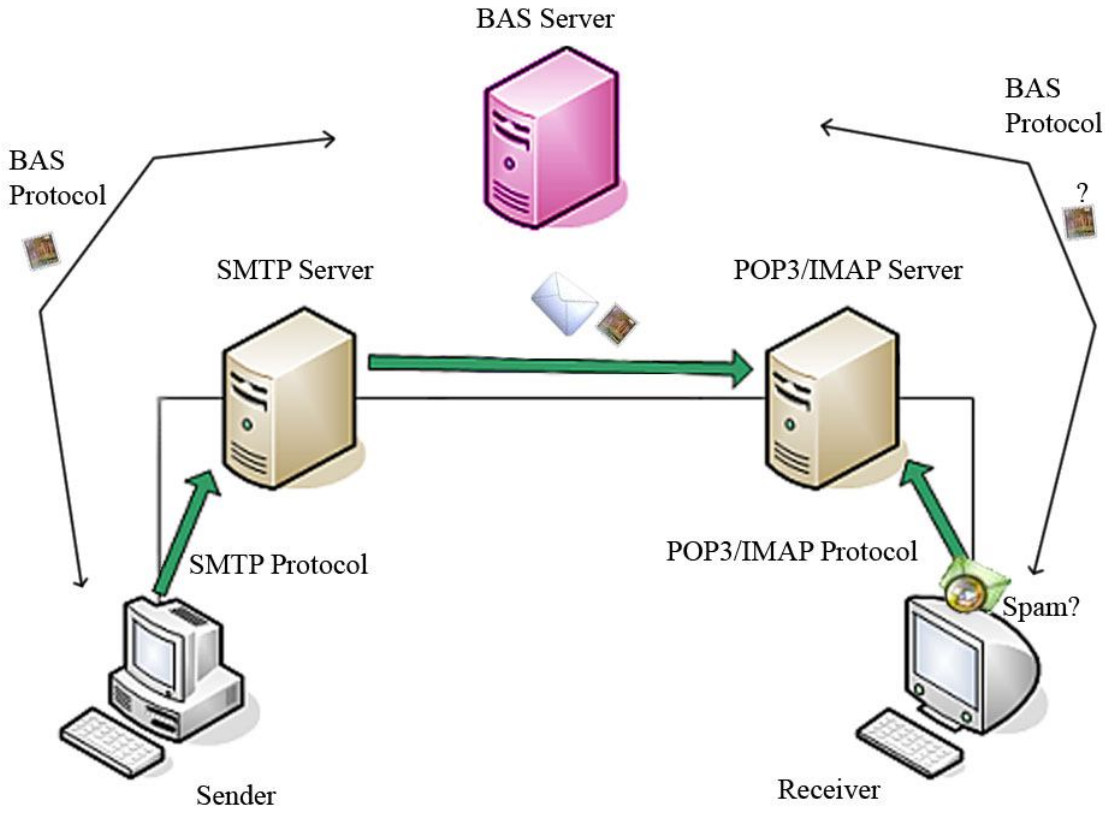


Obr. 3.2: Komunikácia v implementácii na úrovni jadra systému

3.3.3 **Implementácia na aplikačnej úrovni – mailový klient** (Obr. 3.3)

Táto implementácia pozostáva z BAS servera a prídavného modulu do konkrétneho mailového klienta plniaceho funkciu BAS klienta.

Modul je súčasťou daného programu a vie zistiť kedy mailový klient posiela správu a kedy obdrží novú. Nebol by preto problém zo správy zistiť odosielateľa, príjemcu a jej telo. Problém by nenastal ani pri použití šifrovanej komunikácie, pretože mailový klient má prirodzene prístup k dešifrovaným údajom. Prídavný modul by zabezpečil aj patričnú autentifikáciu na BAS server. Posielanie a overenie známky, výmena kreditov by fungovali rovnako ako v predošlých prípadoch. Nezáleží na tom, aký SMTP, POP3 alebo IMAP3 sever požívame. Jedinou nevýhodou je, že sme viazaný na konkrétny mailový klient.



Obr. 3.3: Komunikácia v implementácii na úrovni jadra systému

Táto práca sa ďalej zaoberá implementáciou BAS klienta na úrovni konkrétneho mailového klienta. Zvoleným klientom je Mozilla Thunderbird(verzia 2.0) - voľne dostupný mailový klient s podporu pre viaceré platformy.

Kapitola 4

Vlastná Implementácia

Súčasťou implementácie :

- BAS Server – naprogramovaný v C#
- BAS Klient – prídavný modul (Extension) do Thunderbirdu

4.1 BAS Server

Úlohou BAS Servera je počúvať na zvolenom porte a čakať kým sa pripojí BAS Klient. BAS Server pri tom využíva TCP Sockety[1] [RFC793]. Pri pripojení klienta na TCP Server, vzniká pevné spojenie ktoré pretrváva, až pokiaľ ho jedna zo strán nepreruší. TCP komunikácia garantuje doručenie správy a v prípade opakovaného zlyhania informuje príslušný TCP Socket.
Po vytvorení pripojenia, klient zo serverom komunikuje prostredníctvom jednoduchého textového protokolu – BAS protokolu.

4.1.1 BAS protokol

- **Prihlásenie**
  Príkaz : **Login** $uzivatel
  Odpoveď : **OK**
  Popis : *Ak užívateľ* $uzivatel *existuje v systéme tak sa pošle správa* OK*. Bezpečná autentifikácia je rozobratá v kapitole 5.*
- **Zistenie stavu kreditov**
  Príkaz : **Credits**
  Odpoveď : $pocetkreditov

[1] connection oriented, reliable

Popis : *Zistí stav kreditov na účte prihláseného. Ak nie je užívateľ prihlásený, príkaz sa nevykoná.*

- **Zistenie či je dostatok kreditov na odoslanie správy**

  Príkaz : **EnoughCredits "**$prijemcovia**"**

  Odpoveď : **Enough** *alebo* **NotEnough**

  Popis : *Zistí počet legitímnych príjemcov*[1] *a overí či má prihlásený užívateľ dostatok kreditov. Pokiaľ áno pošle sa* Enough*, inak* NotEnough*. Ak nie je užívateľ prihlásený príkaz sa nevykoná.*

- **Informovanie o odoslaní správy**

  Príkaz : **Send "**$prijemcovia **"** $znamka

  Odpoveď : *žiadna*

  Popis : *Pre každého legitímneho príjemcu si server poznačí informáciu o odoslanej správe s danou známkou* $znamka*. Zároveň sa z konta užívateľa odpočíta X kreditov, kde X je počet legitímnych príjemcov nie väčší ako celkový počet kreditov. V prípade, že počet legitímnych príjemcov prekračuje počet kreditov, prevyšujúci príjemcovia zo zoznamu nebudú zaznačení na server*[2]*. Ak nie je užívateľ prihlásený príkaz sa nevykoná.*

- **Zistenie či je známka validná**

  Príkaz : **Check "**$odosielatel**"** $znamka

  Odpoveď : **Equals** *alebo* **NEquals**

  Popis : *Zistí, či prihlásený užívateľ má správu od odosielateľa* $odosielatel *so známkou* $znamka*. V prípade zhody sa pošle* Equals*, užívateľovi sa pripíše 1 kredit na konto a odstráni sa záznam o správe na serveri, v opačnom prípade sa pošle správa* NEquals*. Ak nie je užívateľ prihlásený príkaz sa nevykoná.*

---

[1] registrovaných na BAS serveri

[2] ich správa nebude mať známku

Každý príkaz je zakončený oddeľovačom $\r\n (0x240D0A hexadecimálne), ktorý predstavuje znak doláru a štandardný oddeľovač riadkov na systémoch windows.

```
Príklad - odosielateľ :
S: <wait for connection on TCP port 2222>
C: <open connection>
S: <connection accepted>
C:    LOGIN peter.cierny@gmail.com
S:    OK
C:    CREDITS
S:    12
C:    SEND "Jozef Kamenny <jokam123@zoznam.sk>, Juraj Medeny
      <jurkooo86@zoznam.sk>" 8ab1ef049a82df056ee723a
C:    CREDITS
S:    10
C: <close connection>
```

Príklad ukazuje proces odoslania známky na server S klientom C. Najprv sa klient prihlási príkazom LOGIN, zistí svoje kredity príkazom CREDITS a odošle dvom užívateľom poštu so známkou. Následne opäť zistí svoj stav kreditov, ktorý sa zmenšil o 2.

```
Príklad – príjemca :
S: <wait for connection on TCP port 2222>
C: <open connection>
S: <connection accepted>
C:    LOGIN jokam123@zoznam.sk
S:    OK
C:    CREDITS
S:    3
C:    CHECK "Cierny Peter <peter.cierny@gmail.com>" 8ab1ef049a82df056ee723a
S:    EQUALS
C:    CREDITS
S:    4
C: <close connection>
```

V tomto príklade je znázornený proces overenia známky pre klienta C serverom S. Klient sa prihlási na server príkazom LOGIN a zistí stav kreditov pomocou CREDITS. Ďalej overí známku od odosielateľa peter.cierny@gmail.com príkazom CHECK. Server potvrdí platnosť známky odpoveďou EQUALS. Nakoniec klient znovu zistí stav svojich kreditov, ktorý sa zväčšil o 1.

4.1.2 Organizácia dát

Dáta na BAS serveri sú uložené v databáze. Na správu databázy je použitý SQL Server Compact Edition. Jadro databázy tvoria 2 tabuľky :

Tabuľka Používateľov

UserID – číselný identifikátor, primárny kľúč

Email – reťazec, email používateľa, zároveň je aj loginom

Kredity – kredity v číselnom formáte

Tabuľka Mailov

MailID – číselný identifikátor, primárny kľúč

SenderID – odosielateľ správy, cudzí kľúč z tabuľky používateľov

ReceiverID – príjemca správy, cudzí kľúč z tabuľky používateľov

Známka – MD5 hash, založený na obsahu správy

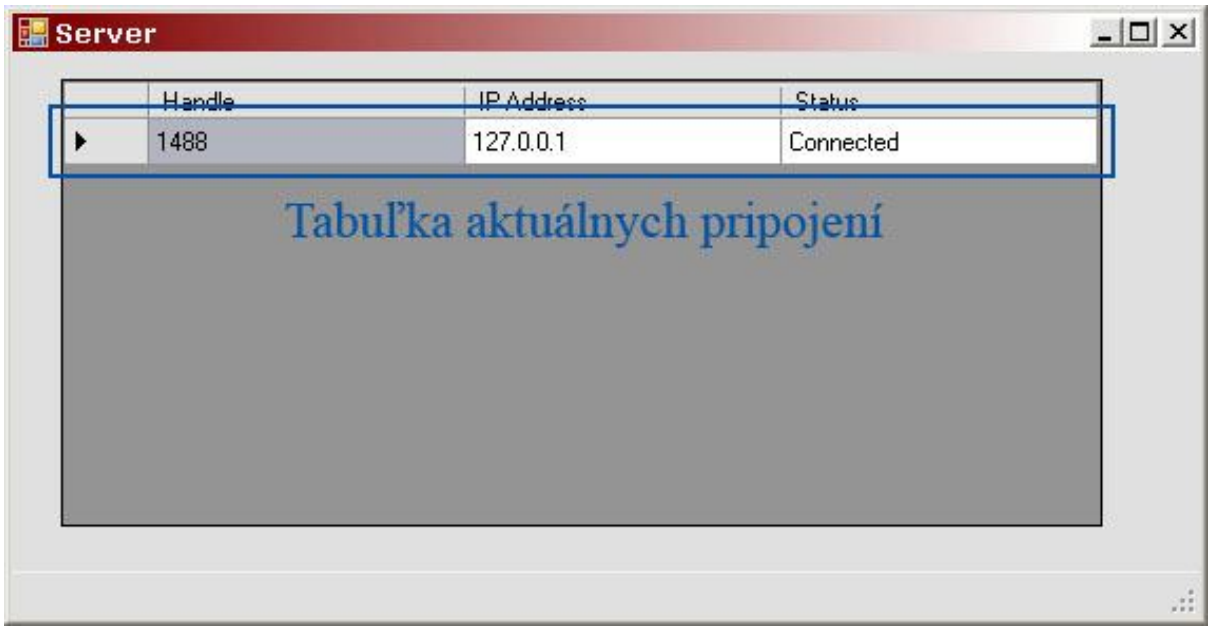
Server si udržiava aj informácie o aktuálnych pripojeniach :

Socket – socket pripojenia, obsahuje informácie o pripojení a zároveň slúži ako jeho identifikátor

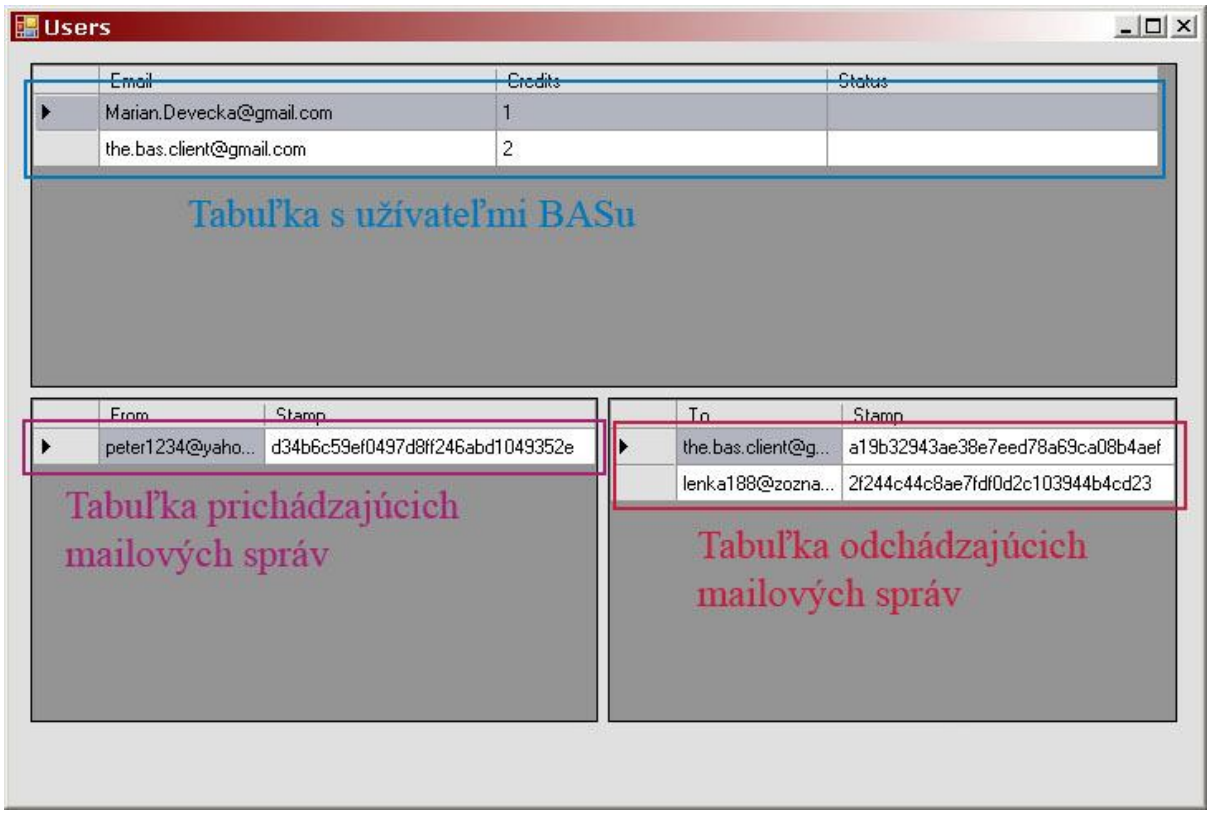
Prijaté dáta – keďže dáta môžu prísť rozdelené do viacerých paketov je potrebné mať premennú do ktorej sa priebežne ukladajú

User – referencia na prihláseného používateľa, ak sa neprihlásil je prázdna

TimeStamp – Čas prihlásenia. Aby sa predišlo „mŕtvym“ pripojeniam, po 5 minútach sa pripojenie automaticky zruší.



Obr. 4.1: znázorňuje okno BAS Servera s aktuálnymi pripojeniami. V tomto prípade je pripojený jeden klient s IP adresou 127.0.0.1 (Localhost).



Obr. 4.2: znázorňuje okno BAS Servera s registrovanými užívateľmi. Ku každému užívateľovi sa zobrazujú aj prichádzajúce a odchádzajúce správy so známkou. V tomto prípade sú registrovaní dvaja užívatelia, pričom prvý z nich má jednu prichádzajúcu a dve odchádzajúce správy.

4.2 BAS Klient – extension do Thunderbirdu

Skôr než sa dostaneme ku konkrétnej implementácii, pozrieme sa na to, ako funguje mailový klient Mozilla Thunderbird a následne si ukážeme, ako sa do takéhoto prostredia robia prídavné moduly.

4.2.1 Mozilla Thunderbird

Thunderbird podobne ako ostatné Mozilla produkty je založený na Mozilla aplikačnom frameworku. Tento multiplatformový framework(umožňujúci písať programy ktoré fungujú na viacerých operačných systémoch) pozostáva z renderovacieho jadro Gecko, sieťovej knižnice Necko a ďalších častí.

Gecko je open-sourcové renderovacie jadro pre prehliadače s podporou pre Internetové štandardy ako HTML [RFC1866], CSS 1-2 [RFC2318], W3C DOM (Document object model), XML [RFC3076], Javascript, atď... Renderovacie jadro umožňuje previesť textový obsah (HTML, XML, Javascript, ...) spolu s formátovaním (napr. Kaskádové štýly CSS) do grafickej formy, ktorá sa dá zobraziť na obrazovku. Gecko naviac umožňuje tieto dokumenty aj parsovať a ich obsah modifikovať. Je napísané v C++ a beží na viacerých operačných systémoch vrátane MS Windows, BSD, Linux, MAC OS, Solaris, OS/2, AIX a OpenVMS. Momentálne je druhým najpopulárnejším renderovacím jadrom, hneď po Tridente, využívanom v Exploreri a nasledovaný WebKitom, ktorý používajú Safari a Google Chrome. [1, 2, 4, 9]

Súčasti operujúce na Gecku v Mozilla Thunderbirdu vytvárajúce jeho výzor a funkčnosť sú :

- Jar súbory
- XPCOM Komponenty

Jar súbory

Jar súbory sú zip archívy s nasledovnou štruktúrou :

- Adresár Contents
  Obsahuje súbory, ktoré popisujú základný grafický vzhľad a implementujú udalosti, ktoré riadia chod aplikácie(stlačenie tlačítka, pohyb myši...).
  Súbory :
  Xul – sú xml súbormi, ktoré definujú grafický vzhľad okien, dialógov a ich grafických komponentov(podobne ako pri HTML), Gecko ich dokáže renderovať do grafickej podoby.
  Js – javascript, skriptový kód ktorý riadi funkcionalitu
- Adresár Locale
  Obsahuje podadresáre k jednotlivým jazykom, v ktorých sú súbory definujúce jazykové preklady.
  Súbory :
  Dtd – definujú texty a popisy, multijazyčná podpora

- Adresár Skin
  Obsahuje podadresáre k jednotlivým oknovým schémam - skinom.
  Súbory :
  Css – kaskádové štýly, známe z HTML, definujú vzhľad komponentov a
  Obrázkové súbory Png, Jpg ...

Na rozdiel od iných mailových klientov(ako napr. Outlook), veľká časť funkčného kódu sa kompiluje/interpretuje počas behu. Využíva sa pri tom metóda „Lazy Load“, čo znamená, že časť programu sa nahrá až keď ju je treba. Výhodou je, že sa šetria pamäťové zdroje a rýchlejšie sa spustí program. Nevýhodou môže byť dlhší čas pri nahrávaní nových častí počas behu, avšak to len pri prvom prístupe.

Komponenty

Klasický javascript je skriptový jazyk, ktorý nám umožňuje robiť výpočty, meniť výzor grafických častí okien, otvárať nové okná, zatvárať ich a podobne. Čo nám neumožňuje, je napríklad prístup k súborom na disku, sieťovým zdrojom, prístup k threadom a processom, atď...
Mozilla produkty využívajú na rozšírenie funkcionality javascriptu tzv. XPCOM Komponenty.
XPCOM – Cross Platform Component Object Model, je v preklade Multi-platformový komponentový objektový model podobný Microsoft COM. Zahŕňa podporu pre viacero jazykov, čím umožňuje ich použitie v Javascripte, Jave a Pythone. Jadro XPCOMu tvorí len niekoľko tried a komponentov, ktoré poskytujú základnú podporu pre prácu zo súbormi a pamäťou, riadenie threadov, definuje základné dátové štruktúry(reťazce, polia, varianty)... Zvyšok komponentov poskytujú iné súčasti ako napríklad Gecko, Thunderbird alebo komponenty prídavných modulov. V Thunderbirde sa XPCOM komponenty nachádzajú v adresári Components. [1, 2, 4]

XPCOM Komponent pozostáva z dynamickej knižnice pre danú platformu[1] a binárneho interfacového súboru Xpt definujúceho jeho rozhranie. Rozhranie popisuje exportované premenné a funkcie, spolu s ich typmi a parametrami. XPIDL je platformovo a strojovo nezávislý jazyk na popis rozhrania. Xpt súbory sú skompilovanou formou XPIDL. Existuje možnosť vytvárať komponenty aj v Javascripte. [1, 2]

Most medzi Javascriptom a XPCOM komponentami vytvára XPConnect. XPConnect je technológia, ktorá umožňuje objektom v Javascripte transparentne zaobchádzať s XPCOM komponentami a naopak. Prepojenie môže byť vytvorené iba pri prítomnosti interfacových Xpt súboroch. XPConnect priamo nepodporuje Microsoft COM a ani iné jazyky ako Javascript. [1, 2]

4.2.2 Prídavné Moduly (Extensiony)

Súčasťou každého dobrého programu je možnosť prispôsobiť si ho podľa vlastných predstáv. Konfiguračné schémy a súbory z časti napĺňajú túto požiadavku. Avšak neumožňujú pridať do programu dodatočnú, novú funkcionalitu. Na tieto účely sú v programoch tzv. prídavné moduly – Add-ony, Add-iny alebo Extensiony. Mozilla Thunderbird zaužíva pojem Extensiony a má pre ne veľmi dobrú podporu. Základ extensionu v Thunderbirde je súbor s príponou xpi. Tento súbor je zip archívom ktorý obsahuje :

- Install.rdf
- Chrome.manifest
- Adresár Chrome
- Adresár Components

Install.rdf

Formát súboru je Rdf/Xml a obsahuje informácie potrebné pre inštaláciu extensionu. Príkladom je verzia a názov modulu, autor, univerzálny identifikátor, identifikátor cieľovej aplikácie, inštalovaný balíček a iné.

[1] súbory s príponami "dll" vo Windowse, v Linuxe je to "so"

Chrome.manifest

Určuje ako časti modulu zapadajú do cieľovej aplikácie.

Adresár Chrome

Podobne ako v samotnom Thunderbirde adresár chrome obsahuje .jar archívy, v ktorých sa nachádzajú adresáre a súbory popisujúce grafické rozhranie, funkcionalitu, jazykovo-definičné súbory...

Adresár Components

Obdobne, v tomto adresári sa nachádzajú XPCOM komponenty modulu.

Každý prídavný modul sa musí napájať na pôvodnú aplikáciu, ktorej funkcionalitu obohacuje. Otázkou je ako a kam pripojiť jednotlivé časti? Predstavme si, že chceme rozšíriť Thunderbird o nové tlačítko v hlavnom okne. Vizuálne komponenty hlavného okna reprezentuje niektorý zo XUL súborov, v tomto prípade to je súbor mailWindowOverlay.xul. V tomto XML súbore si nájdeme miesto, kde chceme pridať naše tlačítko. Naše tlačítko je reprezentované elementom <Button> a rovnako je aj miesto kam ho chceme pridať reprezentované xml-elementom. Každý element má svoj unikátny identifikátor. Na základe elementu a identifikátora vieme vytvoriť XUL-Overlay – XUL súbor ktorý nám zadefinuje novo pridané tlačítko. To ako jednotlivé XUL súbory/Overlaye do seba zapadajú nám definuje Chrome.manifest.

4.2.3 BAS klient – Extension do Thunderbirdu

Úlohou nášho extensionu je zabezpečiť pri odoslaní správy aj odoslanie známky na BAS server a zároveň pri prijatí správy zistiť validitu známky. O prijatej správe musí vedieť rozhodnúť, či je legitímna alebo spam. Mal by ďalej vedieť graficky zobraziť počet kreditov. Nutnou súčasťou je aj okno s nastaveniami BAS klienta.

Komunikáciu BAS klienta zo serverom nevieme zabezpečiť cez javascript alebo existujúce XPCOM komponenty. Je preto nutné vytvoriť si vlastný XPCOM komponent, ktorý poskytuje nízko úrovňové funkcie na sieťovú TCP/IP komunikáciu.

BAS XPCOM komponent

Implementácia XPCOM komponentu je založená na Gecko SDK knižniciach a jazyku C++. Na prácu s TCP/IP socketmi sa využíva C++ knižnica Boost. Kompilácia je určená len pre platformu Win32.

BAS komponent exportuje nasledovné funkcie :

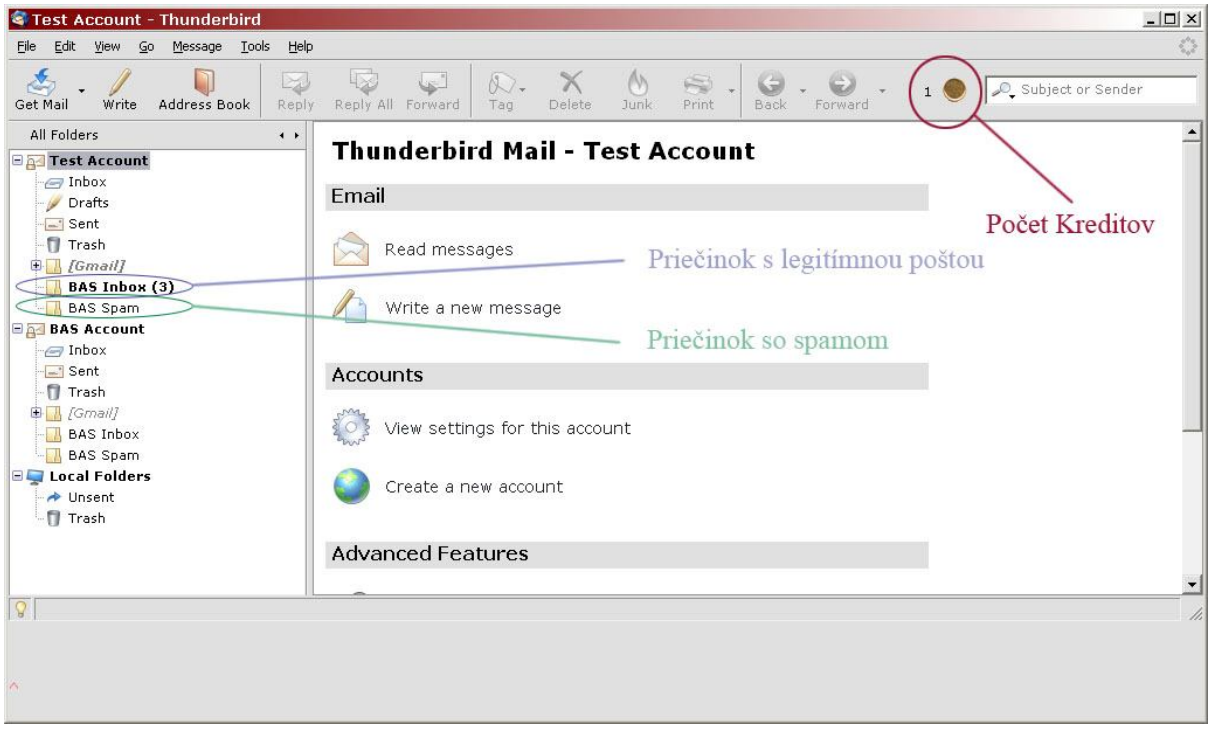
- **Connect**
  Parametre : **adresa**, **port**
  Návratová hodnota : žiadna
  Popis : Pokúsi sa vytvoriť TCP spojenie so serverom na IP adrese **adresa** *a porte* **port.** *Pokiaľ neuspeje vyhodí sa výnimka.*
- **Connected**
  Parametre : žiadne
  Návratová hodnota : **true** *alebo* **false**
  Popis : Zistí, či je vytvorené nejaké TCP spojenie.
- **Disconnect**
  Parametre : žiadne
  Návratová hodnota : žiadna
  Popis : Zruší existujúce TCP spojenie.
- **Login**
  Parametre : **emailova adresa**
  Návratová hodnota : žiadna
  Popis : Pokúsi sa prihlásiť do systému pomocou dotazu „LOGIN“, užívateľským menom je **emailova adresa.**
- **RequestCredits**
  Parametre : žiadne
  Návratová hodnota : žiadna
  Popis : Pošle na server dotaz „CREDITS“ na zistenie kreditov.
- **EnoughCredits**
  Parametre : **prijemcovia**
  Návratová hodnota : žiadna
  Popis : Pošle na server dotaz „ENOUGHCREDITS“, ktorý má zistiť či je dostatok kreditov na odoslanie správy príjemcom **prijemcovia**.

- **CheckStamp**
  Parametre : **odosielatel**, **znamka**
  Návratová hodnota : žiadna
  Popis : Pošle na server dotaz „CHECK“, ktorý vyžaduje test známky **znamka** *od odosielateľa* **odosielatel**.
- **SendStamp**
  Parametre : **prijemcovia**, **znamka**
  Návratová hodnota : žiadna
  Popis : Pošle na server dotaz „SEND“, ktorý oznamuje serveru odoslanie emailu so známkou **znamka** *príjemcom* **prijemcovia**.
- **Read**
  Parametre : žiadne
  Návratová hodnota : **sprava**
  Popis : Funkcia Read načíta správu ktorú odoslal server.

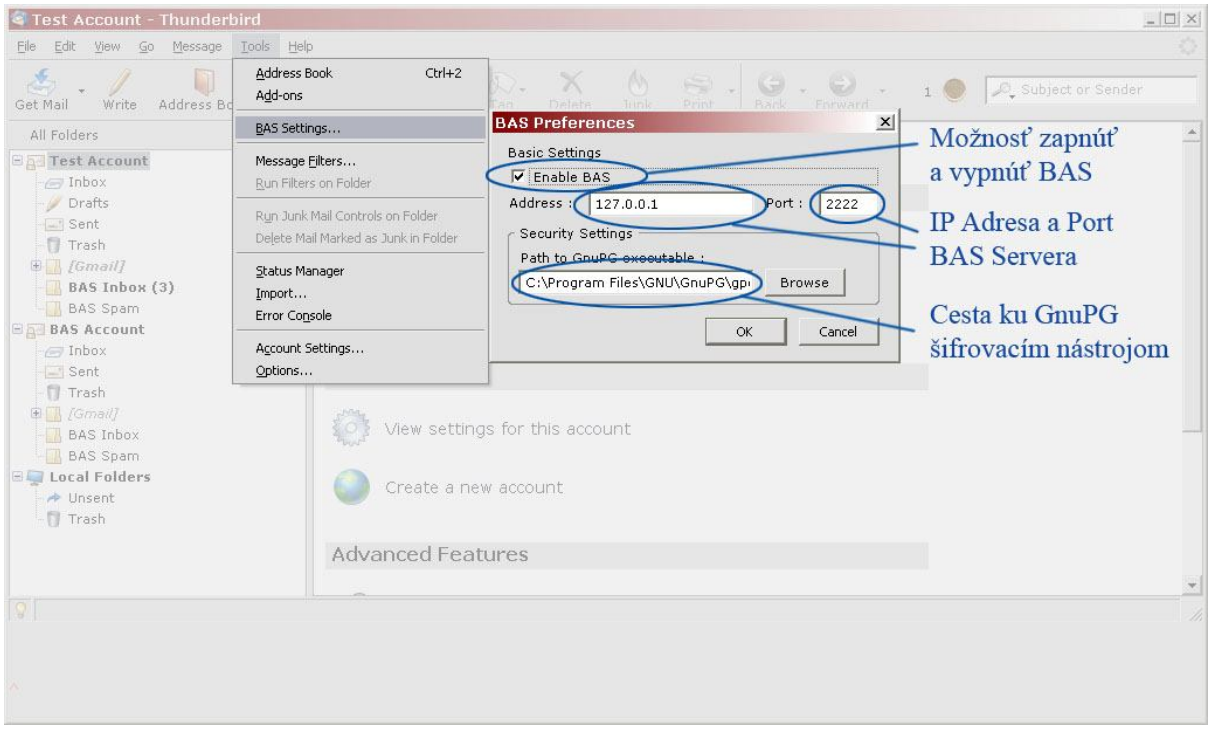
Pri písaní novej správy v Thunderbirde BAS extension zabezpečuje niekoľko vecí. Najprv sa po stlačení tlačítka na odoslanie správy skontroluje pomocou funkcie EnoughCredits, či má daný užívateľ dostatok kreditov na odoslanie správy zvoleným respondentom. Pokiaľ nie, otvorí sa okno, ktoré informuje o nedostatku kreditov a možnosti poslať správy bez podpory BAS. Po úspešnom odoslaní Thunderbird uloží túto správu do priečinka odoslaných správ (zvyčajne Sent). Náš extension vie, že bola odoslaná správa a zo správy vyberie príjemcov a telo správy. Pomocou MD5 hashu vytvorí z jej tela známku. Potom odošle notifikáciu o novej správe BAS serveru pomocou funkcie SendStamp. Aktualizujú sa dostupné kredity funkciou RequestCredits a zobrazí sa ich počet v hlavnom okne.

Po obdržaní novej správy, Thunderbird túto správu uloží do priečinku nových správ (zvyčajne Inbox). Po tom ako BAS extension zistí prítomnosť neprečítanej správy v tomto priečinku, začne sa preverovať jej legitimita. Zistí sa odosielateľ a telo. Podobne ako pri odosielaní sa známka vyrobí z tela pomocou MD5 hashu. Extension potom pomocou funkcie CheckStamp zistí validitu známky. Ak validácia prebehla úspešne, čo znamená, že známky boli zhodné, prijatá správa sa presunie do priečinku BAS Inbox. Ak známka nebola validná alebo server nemal záznam

o odoslanej správe(klasická správa bez využitia BAS), správa – spam sa presunie do priečinku BAS Spam a označí sa ako prečítaná. Mailový klient Thunderbird pri prijatí novej správy oznámi túto skutočnosť výstražným oknom. BAS extension zabezpečuje aby sa pri prijatí spamu takéto okno nezobrazilo. Nakoniec sa aktualizujú dostupné kredity a ich počet sa zobrazí v hlavnom okne.



Obr. 4.3: znázorňuje mailový klient Mozilla Thunderbird s nainštalovaným prídavným modulom BAS. Na obrázku je vidieť vizuálne zobrazenie kreditov, priečinok, kam sa ukladajú legitímne správy(BAS Inbox) a priečinok, kam sa ukladá spam(BAS Spam).



Obr. 4.4: znázorňuje nastavenia BAS v mailovom klientovi Mozilla Thunderbirde s nainštalovaným prídavným modulom BAS. Pomocou nastavení je možné vypnúť/zapnúť BAS, nastaviť adresu a port na ktorom sa nachádza BAS Server a cestu ku GnuPG šifrovacím nástrojom.

Kapitola 5

Bezpečnosť a Šifrovaná komunikácia

Pri posielaní akýchkoľvek správ cez internet je dobré si uvedomiť, že správa nejde priamo k cieľu. Prechádza cez viacero routrov a počítačov, ktoré ju môžu odchytiť a následne si ju prečítať alebo aj modifikovať.
Rozlišujeme pritom 2 klasické typy útokov :

- Útok opakovaním (replay attack) – útočník využíva staršie správy v aktuálnom behu.
- Útočník uprostred (man in the middle) – útočník nepozorovane pôsobí ako prostredník v komunikácii.

Základný BAS protokol nie je bezpečný.

```
C1:   LOGIN peter.cierny@gmail.com
S:    OK
C1:   CREDITS
S:    12
C1:   SEND "Jozef Kamenny <jokam123@zoznam.sk>, Juraj Medeny
      <jurkooo86@zoznam.sk>" 8ab1ef049a82df056ee723a
C1:   CREDITS
S:    10
```

Klient C1 sa nikdy nepreukázal, že je skutočne vlastníkom konta „peter.cierny@gmail.com“. Rovnako sa môže aj klient C2 prihlásiť na server stačí, že pozná daný login.
Nepomôže nám ani ak rozšírime tento protokol o heslo. Hoci už nebude stačiť len jednoducho poznať alebo uhádnuť login, útočník, cez ktorého takáto správa prechádza, si náš login s heslom odchytí.
Nato aby sme vhodne zaručili bezpečnosť využijeme asymetrické šifrovanie.

5.1 Asymetrické šifrovanie

Jedná sa kryptografiu – proces utajovania informácii, ktorej základom je využitie dvoch k nejde kľúčov, súkromného a verejného. Prostredníctvom verejného kľúča

vieme zašifrovať ľubovoľnú správu do nezrozumiteľnej formy – ciphertextu. Ale len držiteľ súkromného kľúča vie túto správu dešifrovať. V dnešnej dobe sa využíva najmä trojica kryptovacích algoritmov, ktoré používajú asymetrické šifry – RSA, DSA a ElGamal. [8]
Balancovaný Antispamový Systém na účely kryptovania využíva GNU Privacy Guard, ktorý má podporu pre všetky spomínané algoritmy. GNU Privacy Guard alebo GnuPG je úplnou a voľne dostupnou implementáciou OpenPGP štandardu, ktorý je definovaný v [RFC4880]. GnuPG umožňuje šifrovanie aj podpisovanie údajov, používa univerzálny systém na správu kľúčov a dá sa ľahko integrovať do iných aplikácii. [14]

Využime teda asymetrické šifry v našom BAS protokole. Predpokladajme, že každý užívateľ BASu bude vlastniť svoju dvojicu kľúčov – verejný a súkromný. Ďalej predpokladajme, že aj BAS server bude mať takúto dvojicu. Svoj súkromný kľúč pozná iba vlastník, ale verejné kľúče sú dostupné komukoľvek. BAS server vie rozlíšiť ku ktorému užívateľovi patrí príslušný verejný kľúč a užívatelia vedia ktorý verejný kľúč patrí serveru.
Označme (S)Ka, správu S zašifrovanú verejným kľúčom užívateľa A. Ďalej označme E(X) útočníka, ktorý sa vydáva za užívateľa X.
Zašifrovaná komunikácia BAS protokolu by potom vyzerala nasledovne :

```
C->S:   (LOGIN peter.cierny@gmail.com heslo)Ks
S->C:   (OK)Kc
   -    Kc zistí server na základe loginu "peter.cierny@gmail.com"
C->S:   (CREDITS)Ks
S->C:   (12)Kc
...
```

Toto riešenie žiaľ nerieši možnosť zneužitia. Útočník síce nebude vedieť rozlúštiť komunikáciu, avšak môže sa mu napriek tomu podariť prihlásiť do systému.

```
C->E(S):   (LOGIN peter.cierny@gmail.com heslo)Ks
   -    Útočník získa správu, ktorú môže použiť aby sa napojil na konto
      "peter.cierny@gmail"
E(C)->S:   (LOGIN peter.cierny@gmail.com heslo)Ks
...
```

Pridajme do tela správy aj sieťovú identitu užívateľa – jeho IP adresu.

```
C->E(S):   (LOGIN 192.168.65.33 peter.cierny@gmail.com heslo)Ks
   -    Útočník získa správu, ale nemôže ju použiť nato aby sa napojil na
      konto "peter.cierny@gmail", pretože nemá požadovanú IP adresu.
```

Neodbytný útočník je však vytrvalý a naďalej odpočúva komunikáciu.

…

```
C->S:   (SEND 192.168.65.33 "Jozef Kamenny <jokam123@zoznam.sk>"
      8ab1ef049a82df056ee723a)Ks
```

…

Odchytenú správu si zapamätá a replay útokom pošle správu opakovane serveru. Koniec koncov môže nasimulovať celú komunikáciu už od loginu. Výsledkom je, že BAS server vytvorí duplikáty odoslania správy s tou istou známkou(čo sa reálne môže stať aj v skutočnosti) a okradne tak užívateľa o všetky jeho kredity.

Na ochranu proti replay útokom využijeme príležitostné slová. Príležitostné slovo je ľubovoľný reťazec, ktorý zabezpečuje čerstvosť správy. Dobrým zvykom je pri každom použití príležitostného slova čiastočne ho zmeniť.

Do BAS protokolu zavedieme nový príkaz AUTH, ktorý požaduje od servera príležitostné slovo, ktoré sa vždy náhodne generuje pri napojení sa klienta na server.

```
S: <connection accepted>
   -    Server vygeneruje príležitostné slovo "sprava84352"
C->S:   (AUTH 192.168.65.33 peter.cierny@gmail.com heslo)Ks
S->C:   (DONE sprava84352)Kc
   -    Príkaz AUTH nám vráti príležitostné slovo
C->S:   (LOGIN 192.168.65.33 peter.cierny@gmail.com heslo sprava84353)Ks
   -    Príležitostné slovo zväčšíme o číslo 1 (požadovaná zmena)
S->C:   (OK sprava84354)Kc
C->S:   (CREDITS 192.168.65.33 sprava84355)Ks
S->C:   (12 sprava84356)Kc
C->S:   (SEND 192.168.65.33 "Jozef Kamenny <jokam123@zoznam.sk>"
      8ab1ef049a82df056ee723a sprava84357)Ks
…
```

Príležitostné slovo platí počas jedného spojenia, pri novom spojení server vytvorí nové slovo. Útočník ho nepozná a po každej výmene v komunikácii sa mení, čím sa zaručuje bezpečnosť proti opakovanému útoku.

Všimnime si, že už nie je nutné ani použiť heslo, nik iný ako užívateľ C ktorý vlastní konto "peter.cierny@gmail" a jeho súkromný kľúč, sa príležitostné slovo nedozvie.

Niektoré IP siete využívajú tú istú verejnú IP adresu pre viacero počítačov naraz. V takom prípade použitie IP adresy ako sieťového identifikátora nemusí byť najvhodnejšie. Nahradíme ju preto ďalším príležitostným slovom. Tento raz slovo generuje klient pri každom napojení.

```
S: <connection accepted>
   -    Príležitostné slovo – server : "sprava9641", klient : "sprava1304"
C->S:   (AUTH peter.cierny@gmail.com sprava1304)Ks
S->C:   (DONE sprava1305 sprava9641)Kc
C->S:   (LOGIN peter.cierny@gmail.com sprava1306 sprava9642)Ks
   -    Nie je nutné už druhýkrát uvádzať login
S->C:   (OK sprava1307 sprava9643)Kc
C->S:   (CREDITS sprava1308 sprava9644)Ks
S->C:   (12 sprava1309 sprava9645)Kc
C->S:   (SEND "Jozef Kamenny <jokam123@zoznam.sk>" 8ab1ef049a82df056ee723a
      sprava1310 sprava9646)Ks
```

Len vlastník súkromného kľúča konta "peter.cierny@gmail.com" a BAS server môžu poznať obe príležitostné slová. Pri každej výmene sa slová obmenia čo zaručuje čerstvosť komunikácie. Komunikácia je preto bezpečná.

Kapitola 6

Možnosti vylepšenia Balancovaného Antispamového Systému

6.1 Šifrovaná Komunikácia

Použitie asymetrického šifrovania pre bezpečnú komunikáciu má za následok veľké spomalenie celého systému. Dôvodom je, že samotný algoritmus asymetrického šifrovania trvá nejaký čas a zároveň je v našej implementácii použité GnuPG z príkazového riadku. Vždy, keď sa ide správa šifrovať alebo dešifrovať, spustí sa "gpg.exe" s požadovanými parametrami na vykonanie šifrovacej funkcie. Toto nie je ideálne riešenie, omnoho vhodnejšie a rýchlejšie by bolo využitie dynamickej knižnice s požadovanými šifrovacími funkciami. Namiesto čistého asymetrického šifrovania by bolo lepšie využiť kombináciu asymetrického a symetrického šifrovania, ktorá je omnoho rýchlejšia. Toto využíva práve TLS a SSL [RFC2246]. Asymetrické šifrovanie sa pritom využije len na začiatku na výmenu symetrického kľúča. Zvyšok komunikácie by bol preto omnoho rýchlejší.

6.2 Neskoré odoslanie známky

Terajšia implementácia funguje tak, že pri posielaní správy sa najprv skontroluje počet kreditov a v prípade, že je všetko v poriadku, odošle sa samotná správa a až následne sa odošle známka správy na BAS server. Mohlo by sa preto stať, že príjemca obdrží správu a odošle známku na kontrolu skôr, než sa samotná známka dostaví na server. Vhodným riešením by bolo odoslať známku pred samotným odoslaním správy a počkať na potvrdenie od servera. Nanešťastie jadro Thunderbirdu neposkytuje vhodný spôsob, ako zistiť kompletný obsah správy pred tým, než sa uloží do priečinku odoslaných správ(čo nastáva až po odoslaní správy).

6.3 Rozšírená autenticita

V našej implementácii je autenticita založená na tele správy. Plánom do budúcna by bolo rozšíriť ju aj na mailové prílohy.

6.4 Doba platnosti

Predstavme si situáciu, že posielame legitímnu správu kamarátovi v rámci BAS systému. Po odoslaní správy sa nám odpočíta 1 kredit z konta. Tento kredit sa akoby dočasne stratí, až pokiaľ príjemca správu nepotvrdí, kedy sa pripíše na jeho konto. Čo ak náš príjemca stratil medzičasom kľúč pre svoje konto alebo sa elektronická správa po ceste stratí? Výsledkom je, že aj náš kredit sa pritom ako keby stratí.

Riešením je zaviesť pre každú známku dobu platnosti. Ak by správa nebola potvrdená do doby uplynutia platnosti, záznam so správou by sa zo servera vymazal a kredit by sa pripísal späť na konto odosielateľa. V mnohých situáciach by bolo pre samotného odosielateľa výhodné, keby si mohol nastaviť vlastnú dobu platnosti. Napríklad chcem rozoslať pár priateľom pozvánky na oslavu, ktorá sa koná cez víkend. Nastavím preto dobu platnosti do piatku večera. V prípade, že niektoré správy nebudú prečítané, vrátia sa mi kredity a naviac by bolo možné zistiť aj o ktorých príjemcov sa jedná.

6.5 Mailing listy

Zatiaľ sme uvažovali len o balancovanej komunikácii medzi dvojicami ľudí, kde zvyčajne pomer prijatých a odoslaných správ je okolo 1:1. Čo ak ale chceme dostávať poštu o nových produktoch alebo vylepšeniach napríklad od výrobcu nášho počítača. Náš výrobca by posielal svojim klientom legitímne správy, avšak len málokedy by dostal naspäť odpoveď. Strácal by preto kredity. Jedným z riešení tohto problému by bolo nastavenie automatickej odpovede v našom mailovom klientovi. Odosielateľ takýchto správ by bol pridaný do zoznamu automatických odpovedí a vždy keď by sme dostali od neho správu, dostal by naspäť prázdnu odpoveď len za účelom navrátenia kreditov. Toto riešenie však nijako nezaručuje, že ľudia ho naozaj budú používať a nebudú si takto získané kredity nechávať. Inou možnosťou je, že

klient by sa najprv musel nahlásiť do požadovaného mailing listu a to prostredníctvom zaslania správy. Spolu s ňou by obdržal aj zálohu vo forme 1 kreditu. Klient by sa mohol odhlásiť poslaním odhlasovacej správy, na ktorú by mu mailing list odpovedal 2 správami v hodnote 2 kreditov. Keďže klient zaslal mailinglistu 2 kredity vo forme prihlasovacej správy(zálohy) a odhlasovacej správy, stav by bol vyrovnaný. Mailinglist by očakával na každú jeho informačnú správu auto-odpoveď. Pokiaľ by ju však nedostal, využil by zálohu ako kompenzáciu a klienta by odstránil zo svojho mailinglistu, čo by znova vyrovnalo stav kreditov. Kto však zaručuje, že mailinglist nebude len pascou, na ktorý sa prihlásime a potom nám už nikdy skutočne nepošle žiadnu správu a naviac sa nám nedovolí ani odhlásiť. Preto by túto funkciu mal vykonávať BAS server, ktorý by ponúkal bezpečné prihlásenie a odhlásenie z ľubovoľného mailinglistu a naviac by automaticky implementoval auto-odpovede.

Kapitola 7

Záver

Táto práca nastolila existujúci problém spamu a poskytla stručný prehľad mnohých existujúcich spôsobov, ako sa proti nemu brániť. Hlavným prínosom bolo preskúmanie nápadu nového experimentálneho antispamového systému – Balancovaného antispamového systému. Preskúmali sa vlastnosti takéhoto systému a bolo prezentovaných niekoľko návrhov na jeho implementáciu – naivná implementácia, implementácia na úrovni mailových serverov, implementácia na úrovni jadra systému a implementácia na úrovni mailového klienta. Spomedzi nich bola prakticky realizovaná implementácia na úrovni mailového klienta. Za mailového klienta bol zvolený voľne dostupný Mozilla Thunderbird. Súčasťou implementácie bol BAS Server a BAS Klient pozostávajúci z XPCOM komponentu a prídavného modulu do Thunderbirdu. Komunikácia medzi BAS Serverom a BAS Klientom je zabezpečená prostredníctvom sieťového TCP protokolu, na ktorom je postavený textový BAS protokol. Aby bola zaručená bezpečnosť, je táto komunikácia asymetricky šifrovaná. Dáta ukladá BAS Server do databázy, pričom využíva SQL Server Compact Edition.
Aj napriek tomu, že sa takýto systém nemusí v budúcnosti reálne používať, práca môže slúžiť ako inšpirácia pre ďalšie návrhy a tým napomôcť v boji proti spamu.

Literatúra

[1] Mozilla Developer Center, XPCOM, GECKO, XUL, Extensions. https://developer.mozilla.org/

[2] Mozilla Projects, Thunderbird, XPCOM. http://www.mozilla.org/projects/

[3] Building XPCOM Components. http://www.iosart.com/firefox/xpcom/

[4] Boswell, D.; King, B.; Oeschger, I.; Collins, P., & Murphy, E. (2002). *Creating Applications with Mozilla*. O'Reilly & Associates, Inc.

[5] Duntemann, Jeff (2004). *Degunking Your Email, Spam, and Viruses.* Paraglyph, Incorporated (http://site.ebrary.com/)

[6] Zdziarski, Jonathan A. (2005). *Ending Spam : Bayesian Content Filtering and the Art of Statistical Language Classification*. No Starch Press, Incorporated (http://site.ebrary.com/)

[7] Thomas, J.P., & Essaaidi, M. (2006). *Information Assurance and Computer Security*. IOS Press (http://site.ebrary.com/)

[8] Oppliger, Rolf (2005). *Contemporary Cryptography.* Artech House, Incorporated (http://site.ebrary.com/)

[9] Wikipedia the free Encyclopedia. http://www.wikipedia.org/

[10] MAAWG (October 2007). *Email Metrics Perspective: The Network Operators' perspective.* http://www.maawg.org/about/MAAWG20072Q_Metrics_Report.pdf

[11] SpamAssasin Homepage. http://spamassassin.apache.org/

[12] Project Honey Pot Homepage. http://www.projecthoneypot.org/

[13] Unspam Homepage. http://www.unspam.com/

[14] GnuPG Homepage. http://www.gnupg.org/

Appendix

Návod na kompiláciu

BAS Server

Prerekvizity :

- **Microsoft Visual C# 2008 Express Edition**

Kompilácia :

Stačí otvoriť BASServer.csproj vo Visual štúdiu a projekt skompilovať. Nezáleží pritom, či sa použije Release alebo Debug kompilácia.

BAS Klient – XPCOM Komponent

Prerekvizity :

- **Microsoft Visual C++ 2008 Express Edition**
- **Boost C++ Libraries (http://www.boost.org/)**
- **Gecko SDK (xulrunner verzia 1.8.1.3)**

Kompilácia :

Otvorte BASComponent.vcproj vo Visual štúdiu. Nastavte kompiláciu v móde Release. V Project Options – C/C++ -> General -> Additional Include Directories nastavte cesty k Gecko a Boost header súborom(.h, .hpp). V Linker -> General -> Additional Library Directories nastavte cesty ku Gecko a Boost knižniciam. Skompilujte.

BAS Klient – Thunderbird Extension

Prerekvizity :

- **Winrar (prípadne iný pakovací program podporujúci Zip)**

Prídavný modul do pozostáva zo súboru xpi. Na zbalenie adresárovej štruktúry prídavného modulu do xpi použite "!make.bat". Zmeňte v ňom cestu k baliacemu programu Winrar. Nakopírujte dynamickú knižnicu BAS XPCOM Komponentu do adresára "components". Spustite "!make.bat".

Návod na spustenie

Prerekvizity :

- **BAS Server**
- **Mozilla Thunderbird 2**
- **BAS Thunderbird Extension**
- **SQL Server Compact Edition**
- **GnuPG (http://www.gnupg.org/)**

V GnuPG si najprv vyrobíme potrebné kľúče pomocou príkazu "gpg.exe --gen-key". Zvolíme "DSA and Elgamal", prázdny passphrase. Email sa musí zhodovať zo skutočnou mailovou adresou. Identita-Email BAS servera je "bas.server@server".

Spustíme BAS server, nastavíme v okne Settings cestu k "gpg.exe" a server spustíme tlačítkom Start Server. V okne Users pridáme požadovaných užívateľov, identifikovaných podľa emailovej adresy.

V Mozilla Thunderbirde zvolíme Tools -> Addons -> Install. Vyberieme BAS.xpi, potvrdíme a reštartujeme Thunderbird.
V Tools -> BAS Settings nastavíme IP adresu BAS Servera, port (štandardne 2222) a cestu ku "gpg.exe". Nakoniec už len zaškrtneme políčko BAS Enabled.