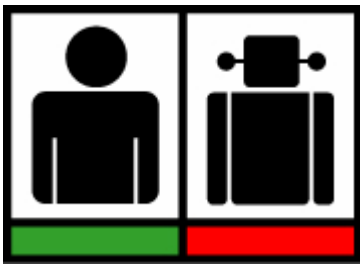


Internet mail anti-spamsystem (ASS)

Förslag till arkitektur

version 1.0

Projektarbete inom Dataföreningens kurs certifierad IT-arkitekt, kurs K28

Abstract

Detta projektarbete beskriver ett *förslag på arkitektur* för ett *anti-spamsystem* som bygger på *captcha* teknik. Arbetet omfattar; en *inventering* av befintliga filtreringstekniker med deras för- och nackdelar, samt en mindre *enkät* undersökning hos e-post användare. Projektarbetet har utförts av författaren som en del av utbildningen i Dataföreningens kurs ”Certifierad IT-arkitekt”.

2007-11-28
Toni Thomsson
toni@tonjac.org

Revision: 2148

Sammanfattning

En stor del av all elektronisk post som skickas över Internet är av karaktären ”skräppost” (även kallad SPAM eller oönskad post). Arbetet som beskrivs i detta dokument har gjorts som ett försök att hitta en lösning (arkitektur för ett mjukvarusystem) som minimerar mängden ”oönskad post” samtidigt som risken för att ”ren” post inte når mottagaren minimeras.

Arbetet innehåller en *inventering* av befintliga filtreringstekniker med deras för- och nackdelar. Inventeringen och utvärderingen av de befintliga filtreringsteknikerna visar att det inte finns några filter som fungerar helt perfekt utan att de bästa lösningarna innehåller en kombination av flera olika tekniker eller filter. Arbetet omfattar också en liten *enkätundersökning* till e-post användare. Undersökningen visar att många har problem med skräppost framförallt privat och att skräppost oftast är av typerna; *reklam, nätfiske* eller *spridning av skadlig programvara* vilka i princip alltid är *maskingenererade*.

Lösningen som föreslås i detta arkitekturförslag bygger på att *filtrera bort maskingenererad post* med hjälp av *captcha-teknik*. Captcha-tekniken innebär att förvrängda siffror och/eller bokstäver renderas som en digital bild. Bilden skickas till avsändaren med en uppmaning om att ange koden i bilden. Att tolka en sådan bild är relativt enkelt för en människa men mycket svårt för en dator (maskin). Eftersom du som mottagare troligtvis vill att viss post (nyhetsbrev eller andra liknande tjänster) som är maskingenererad skall komma fram kompletteras lösningen med en *vitlista* där adresser till *önskade maskiner* anges och därför godkänns som avsändare. Till sist innehåller lösningsförslaget en öppning för att integrera andra *externa filtreringstekniker* som kan köras för att göra *extra kontroller* på post som systemet anser vara maskingenererad.

Innehållsförteckning

1. Inledning

1.1 Bakgrund

Idag är större delen av post som skickas över Internet av karaktären ”skräppost”. Skräppost kan vara utskick av reklam, phishing försök och distribution av virus, trojaner och spyware. Det finns olika typer av skydd i form av ”filter” som man kan använda sig av som användare för att filtrera bort skräppost. Tyvärr fungerar dagens konventionella metoder för rensning av skräppost inte alltid bra. Skräppost släpps ibland igenom filtret (false positives) och ibland tas ”ren” post bort felaktigt (false negatives). Detta innebär inte bara säkerhetsrisker och ”irritationsmoment” när man försöker läsa sin e-post, det kan också leda till att man måste överge sin e-post adress och byta till en ny.

1.2 Problem

Föreslå en arkitektur för ett anti-spamsystem (ASS) som löser problemet med filtrering av skräppost på ett mer pålitligt och robust sätt än dagens konventionella metoder.

Följande frågor pekar på de viktigaste problemen som måste lösas:

- Hur minimeras risken att systemet filtrerar bort ”ren” post?
- Hur minimeras risken att systemet släpper igenom skräppost?

1.3 Avgränsning

Arbetet är begränsat till att hitta en lösning med hjälp av befintliga och etablerade tekniker och standards som t ex SMTP [1], POP3 [2], IMAP [3] och http [4]. Ambitionen är inte att föreslå ett nytt protokoll för postförmedling över Internet som är konstruerat på ett sätt som omöjliggör spam.

1.4 Målgrupp

Dokumentet vänder sig i första hand till arkitekter och konstruktörer som skall utveckla ett ASS samt till dess beställare eller ägare. För att kunna ta till sig alla detaljer i dokumentet krävs att läsaren har erfarenhet av professionell systemutveckling och vanliga Internet standards. Vissa delar av dokumentet kan läsas av alla som är intresserade av ämnet ”anti-spam”. Se *Sammanfattning* och *10.3 Begrepp och förkortningar*.

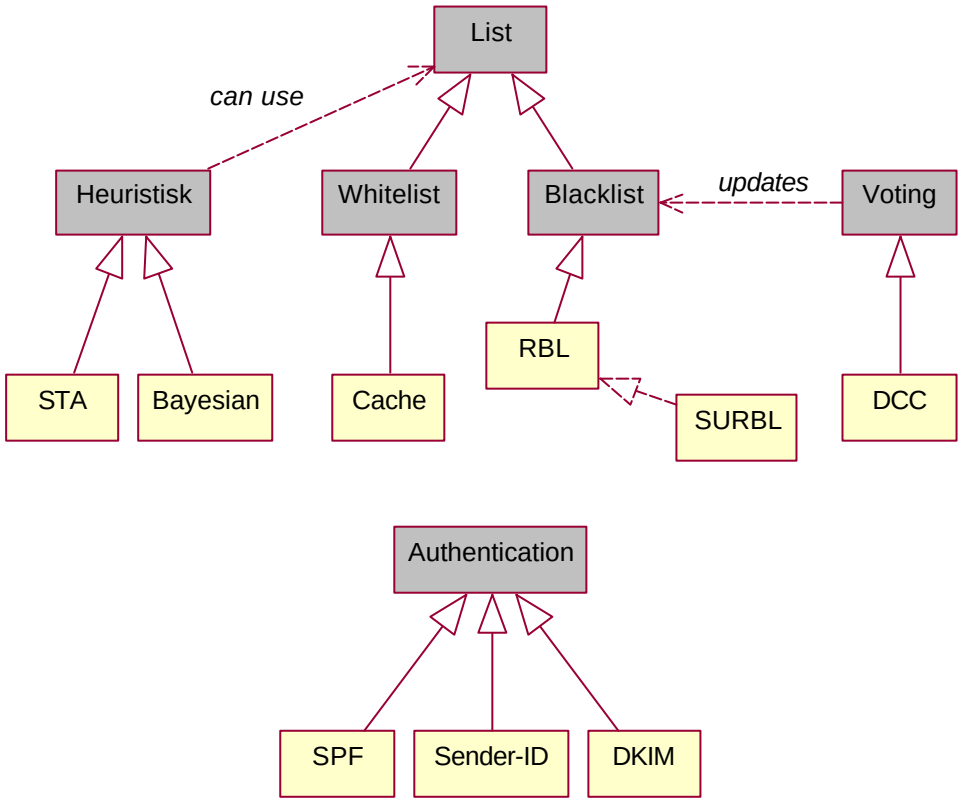
1.5 Metod

Metod som använts i arbetet med att hitta en lösning av problemet har varit att:

- Inventera tekniker som används i ASS och utvärdera dem avseende för- och nackdelar. Hur effektiva är de olika teknikerna?
- Göra undersökningar (främst via Internet) om vad som skrivits i ämnet i form av böcker, rapporter, projekt och standards mm. Finns det någon föreslagen lösning som kan tänkas fungera men som ännu inte implementerats?
- Intervjua e-post-användare genom en enkätundersökning.
- Använda sig av personliga erfarenheter av spam och av tekniker som kan användas i ett ASS.

2. Filtreringstekniker och andra skydd

Det finns många olika tekniker för att filtrera bort skräppost som alla kan delas in i följande kategorier: ”Listor”, ”Heuristisk”, ”Voting” och ”Authentication”, se figur 2.1. I efterföljande avsnitt går vi igenom de olika filtreringsteknikerna och tittar på hur de fungerar och vilka brister de har. Vi tittar också på ett exempel på hur man kan kombinera olika tekniker för att nå ett bättre resultat. Tekniken i exemplet kallas ”Reputation” och är hämtat från Google’s e-post-tjänst Gmail. Till sist går vi kort igenom en teknik (captcha) som används för att skilja på människor och maskiner. Captha används inte direkt som en filtreringsteknik (åtminstone inte i någon större känd tillämpning) utan indirekt som ett komplement för att försvåra för spammare att ”spamma” via gratistjänster på Internet.



Figur 2.1 Kategorisering av filtreringstekniker

2.1 Blacklist

Blacklist eller svartlista är en lista som oftast innehåller IP-adresser till en ”spammare”. Ett ASS jfr. IP-adressen till den som skickar ett meddelande med alla registrerade IP-adresser i svartlistan för att avgöra om meddelandet är spam eller inte. Svartlistor med IP-adresser brukar kallas RBL (realtime spam blacklist). Det finns flera sådana här listor på Internet, t ex SpamCop, ORDB och Spamhouse. Ett annat exempel på svartlista är SURBL (spam URI realtime blacklist) som använder sig av URL:ar istället för IP-adresser. ASS går igenom inkommande meddelanden genom att titta på ev. URL:ar i meddelandet och jämför dessa med en svartlista av SURBL-typ.

En nackdel med alla typer av svartlistor är att det ständigt krävs uppdateringar av listan. Många måste få spam och anmäla det som spam innan listan uppdateras och när väl detta är gjort kanske spammaren har bytt IP-adress eller ändrat URL:arna i utskickade mejl. Fördelen med listor är att det är en enkel teknik som är enkel att implementera och integrera.

2.2 Whitelist

Whitelist eller vitlista är en lista som används direkt i ASS. När ett ASS har bedömt en viss avsändare som ”ren” kan denna avsändare läggas i en cache så att hela proceduren med att göra bedömningen kan hoppas över nästa gång.

2.3 Heuristisk

Heuristisk bedömning av mejl innebär att ett system försöker titta på innehållet i ett meddelande och med hjälp av en algoritm avgöra om meddelandet är spam. Oftast är inte svaret från en sådan här algoritm ja eller nej utan en siffra från 0-10 (sk. ranking) där 10 innebär att meddelandet helt säkert är spam. En heuristisk algoritm kan använda någon typ av lista som hjälp, t ex en lista med ”fula ord”. Flera olika varianter finns t ex Statistical Token Analysis (STA) och Bayesian.

En stor nackdel med algoritm baserad bedömning av ett meddelande är att spammaren kan ”prova” sitt meddelande och redigera texten (buy vi@gr@) så att meddelandet får en låg ranking. Spam från ”ambitiösa” spammare brukar typiskt ligga på 5.

2.4 Voting

Voting eller röstning uppdaterar en lista genom röstning. Ett exempel på röstningsteknik är DCC (Distributed Checksum Clearinghouse) som fungerar på följande sätt. ASS beräknar en checksiffra på ett meddelande och skickar sedan denna till en central röstningsdatabas. Om tillräckligt många röstar på att meddelandet är spam märker ASS meddelandet som spam.

Spammare kan enkelt sätta detta röstningssystem ur spel genom att ändra några tecken i varje meddelande eftersom ett enda tecken ger en helt annan checksiffra.

2.5 Authentication

Authentication innebär att systemet försöker avgöra om avsändaren är ”behörig” att skicka meddelanden. Detta sker oftast med hjälp av DNS (Domain name system) enligt standarden SPF (Sender Policy Framework) [5] som arbetats fram av Network Working Group. Enkelt uttryckt fungerar SPF genom att man slår upp domänen som IP-adressen för avsändaren tillhör och sedan kontrollerar en lista i DNS:en om avsändaren får skicka meddelanden från denna domän. Denna metod är den enda standardiserade metoden mot spam som finns idag. Den är ett ambitiöst försök att stoppa spam och blir säker mer och mer effektiv. Problemet i dagsläget är att metoden inte används i någon större utsträckning, ca 3% av alla domäner på Internet är konfigurerade enligt SPF. Det finns flera andra förslag till standard som alla fungerar på liknande sätt och som inte heller har fått något större genomslag. Sender-ID [7] är ett förslag framtaget på initiativ av Microsoft och övriga förslag finns i RFC 4405 [8], RFC 4407 [9] och DomainKeys Identified Mail (DKIM) i RFC 4871 [14].

Metoden fungerar inte mot spammare som köpt sin egen domän. En spammare som t ex köpt domänen *wespamalot.com* kan konfigurera denna domän så att kontrollen går igenom. Metoden fungerar inte heller i fall där e-post-konton har ”kidnappats”. Kidnappning av konton fungerar så att en spammare registrerar konton på en gratis e-

post-tjänst för att sedan använda dessa konton vid utskick av spam.

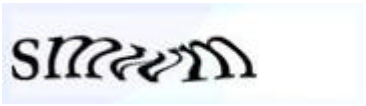
2.6 Reputation

Eftersom ingen teknik (som finns idag) fungerar perfekt när det gäller att avgöra om post är spam eller inte använder en del ASS en kombination av tekniker. Gmail (Google's e-post-tjänst) använder sig av ett system som de kallar "reputation" [6] eller rykte på svenska. Ryktet räknas ut för en domän med hjälp av flera tekniker som t ex RBL, whitelist och Authentication och bedöms och justerar sig självt genom att man tar hänsyn till hur användarna rapporterar spam. Ryktet är en siffra mellan 0 och 100 där hundra betyder att domänen aldrig spammar.

2.7 Captcha

På Internet är det vanligt att man vill skydda en tjänst från "robotar". En captcha är en "robotfälla", ett test som är lätt att lösa för människor, men inte för automatiska datorprogram. Captchor används vid inloggning på gratis e-posttjänster och ibland även före avsändning av e-post för att förhindra missbruk av e-posttjänsten (utskick av spam) med hjälp av automatiska program, robotar. Captchor används även vid många former av registreringar på webbsidor, som till exempel forum och gästböcker.

Vanliga captchor består av förvrängda eller överlappande ord eller teckenkombinationer som avsändaren måste avläsa och skriva in korrekt i ett fält. Sådana förvrängda tecken är relativt lätta för människor att läsa (förutsatt att de inte är synskadade), men datorprogram för teckenigenkänning klarar dem inte.



Figur 2.2 Exempel på en captcha - vilka bokstäver är det?

2.7.1 Knäckande av captchas

Det har visat sig mycket svårt att besegra captchas på helautomatisk väg, men utsändare av spam sägs ha börjat erbjuda pornografiskt material på webbsidor i utbyte mot att besökaren löser en captcha, som först hämtats någon annanstans ifrån. Resultatet används sedan för att skapa konton hos någon gratis e-posttjänst på nätet, exempelvis hotmail, som spammaren använder för att ta emot svar från intresserade kunder. Det behövs många sådana konton eftersom varje konto normalt bara kan ta emot några hundra brev.

3. Analys av problemet och förslag till lösning

Nu har vi tittat på de vanligaste tekniska lösningarna för filtrering av skräppost som finns idag samt vilka fördelar och nackdelar dessa lösningar innebär. I detta avsnitt går vi igenom och analyserar problemställningen samt föreslår en lösning av problemet på konceptuell nivå. Lösningen konkretiseras i arkitekturförslaget som beskrivs i kapitel 5 *Perspektiv på systemets arkitektur (viewpoints)* och 6 *Vyer av systemets arkitektur (views)*. Kraven som ställs på arkitekturen kan du läsa om i kapitel *4 Intressenterna och deras krav på systemet*.

3.1 Definition av ”Skräppost”

Först går vi igenom vår definition av *skräppost* eftersom denna definition ligger till grund för problemställningen. Definitionen är baserad på den enkätundersökning, se *11.1 Enkät till E-post användare* och *11.2 Sammanställning av enkät*, som vi gjort för att bilda oss en uppfattning om de största problemen som användare upplever med sin elektroniska post.

Med *skräppost* (även kallat SPAM) avses i detta dokument definitionen som redovisas i tabell 3.1.

”Skräppost” *är elektronisk oönskad post av typerna:* ***reklamutskick, nätfiske*** *och* ***spridning av skadlig programvara****. Post av dessa typer är i princip alltid* ***maskingenererad****.*

Tabell 3.1 Definition av skräppost

3.2 Dagens lösningar

När vi har tittat på dagens anti-spam lösningar och jämför dessa med hur användare upplever problemen med spam kan man konstatera att det finns stora brister i hur skyddet fungerar. Det går troligtvis inte att hitta en enkel lösning till att stoppa oönskad post med dagens ”öppna” system för postförmedling över Internet. Dagens bästa lösningar verkar bestå av en kombination av tekniker.

3.3 Anonymitet är roten till problemet

Om man som avsändare inte kunde vara anonym skulle troligtvis de värsta avarterna av spam försvinna så det är troligt att lösningen ligger i identifikation av avsändaren. Det är möjligt med dagens protokoll att identifiera användare genom att använda PKI (Public Key Infrastructure) men det kräver utbyte och signering av publika nyklar vilket i sin tur betyder att du i förväg måste bestämma vem som skickar brev till dig. Om vi skall uppnå att avsändaren identifierar sig på ett trovärdigt sätt (utan manuell spridning och signering av nyklar) måste vi antagligen bygga om postförmedlings-protokollen från grunden och det var inte meningen och ambitionen med detta arkitekturförslag. Så hur kan vi komma närmare en lösning av problemet med dagens teknik och protokoll?

3.4 Sortera bort maskinerna

Om vi går tillbaka till enkätundersökningen och tittar på vad användare upplever är de värsta problemen med spam så kan man se att sådan post i princip alltid är maskingenererad. Så om vi alltid sorterar bort post som är maskingenererad så har vi tagit bort en stor del av problemet. Vi löser dock inte allt genom denna grova sortering. I scenarier där du registrerar dig på en tjänst, nyhetsgrupp eller liknande skickas ofta maskinellt genererade brev (som du verkligen vill ha) till dig.

3.5 Lösningsförslag

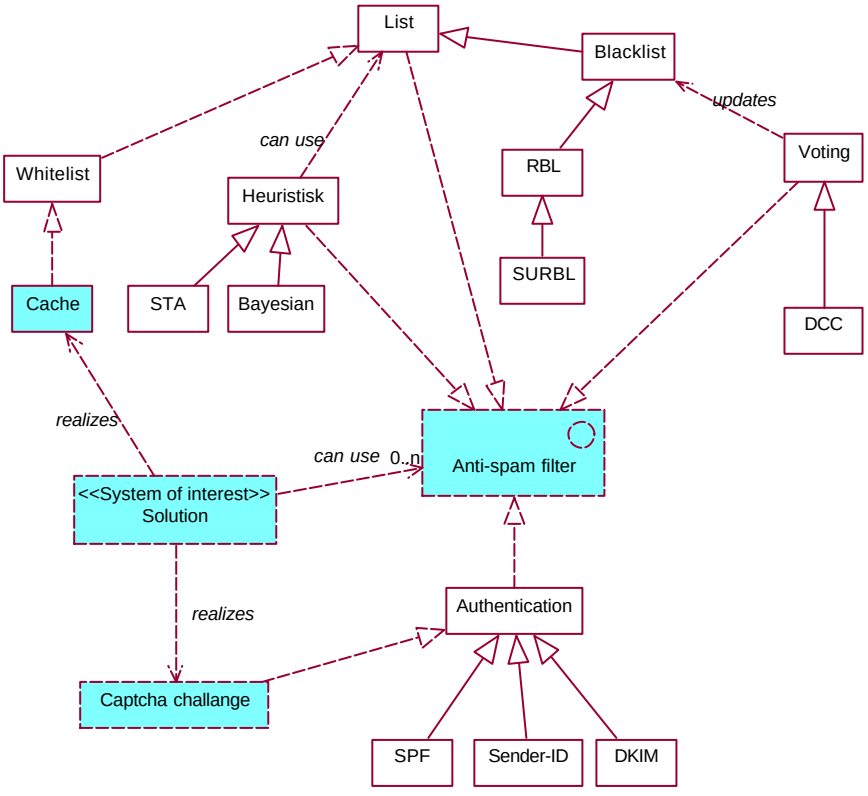
När vi skall ge förslag till lösning har vi tre viktiga fakta att ta hänsyn till:

- Vi vill filtrera bort de allra flesta maskingenererade breven, men…
- … vi vill släppa igenom ”vissa” maskingenererade brev
- Andra ASS (som t ex Google’s GMail) visar att den bästa lösningen är en kombination av befintliga filter

Vi tar hänsyn till ovanstående punkter och föreslår följande lösning:

- Alla brev som tas emot blockeras och läggs i karantän tills vi vet att en människa skickat meddelandet. Vi tar reda på att avsändaren är en människa genom att skicka en Captcha-utmaning till avsändaren i ett blockeringsmeddelande (se figur 3.3).
- Avsändare som auktoriserat sig genom att lösa Captcha-utmaningen sparas i en cache så ett de inte behöver gå igenom processen nästa gång de skickar ett brev till samma mottagare.
- Brev som inte skickas av människor matchas mot en vitlista innan de släpps igenom.
- Vi ger möjlighet att integrera andra externa filter om behov och önskemål finns eller uppstår i framtiden.

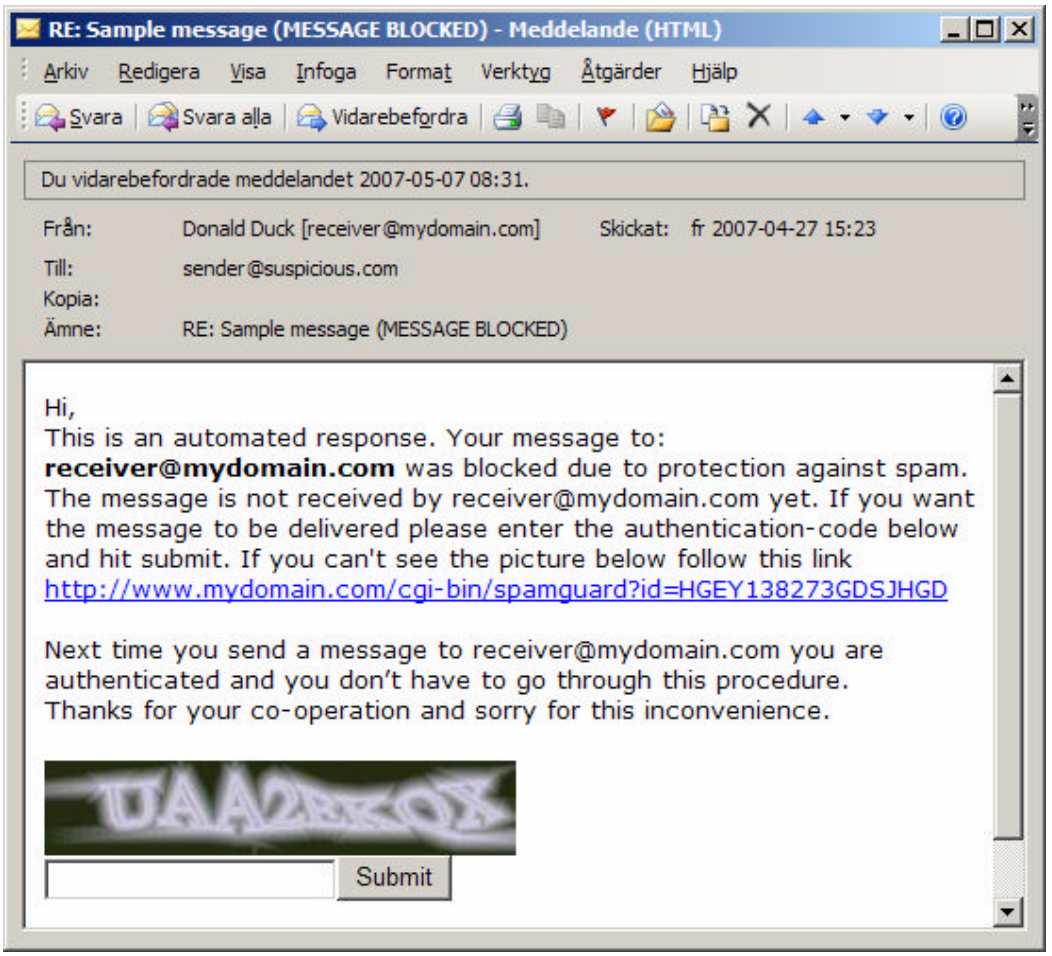
I figur 3.2 kan du se hur lösningsförslaget (Solution) relaterar till befintliga filtreringstekniker från bilden i figur 2.1 och vilka tekniker som lösningen realiserar (skuggade).



Figur 3.2 Lösningsförslaget i relation till befintliga filtreringstekniker

3.5.1 Blockeringsmeddelande

Ett blockeringsmeddelande (se figur 3.3) är ett vanligt e-brev (i HTML-format) som skickas tillbaka till avsändaren med en uppmaning att ange koden som bifogats som en Captcha-bild.



Figur 3.3 Exempel på blockeringmeddelande

4. Intressenterna och deras krav på systemet

Det finns flera andra intressenter till ett ASS förutom slutanvändaren som använder systemet för att ta emot post. I detta avsnitt går vi igenom vilka dessa intressenter är samt hur deras krav och behov på ett ASS ser ut. Sist i detta kapitel sammanfattas kraven i en prioriterad lista där varje krav får en identitet. Kravens identiteter används längre fram i dokumentet för att återkoppla till kraven vid motiveringar av arkitekturförslaget.

4.1 Ägare/beställare

Med intressenten ägare/beställare avses här den person som beställt denna SAD (Software architecture description) vilket i detta fall är samma person som författaren av dokumentet.

Systemet skall fungera som ett skydd mot oönskad post oberoende av programvara som används i miljön hos brukaren. Systemet skall kunna byggas på med externa filtreringsmetoder. Ägaren vill ha möjlighet att fritt distribuera systemet och därför får inte systemet bygga på kommersiell programvara som inte kan distribueras fritt eller som kräver att brukaren måste betala licensavgift.

Systemet skall fungera hos medelstora företag. Företaget kan ha upp till 200 konton där varje konto i snitt får 30 brev/dygn (totalt 6000 brev/dygn). Storleken på ett brev är i snitt 100KByte. Systemet skall klara att av att samtidigt ta emot meddelanden från upp till 20 levererande servrar. Dessa krav skall kunna uppfyllas på en maskin med en billig processor (Intel P4 1.8GHz) med 1GB RAM på en Windows-2000+ eller Linux 2.2+ plattform.

4.2 Slutanvändare

Slutanvändare är den person som tar emot post via systemet. Slutanvändaren hämtar posten från servern (hos brukaren) där systemet är installerat med hjälp av sin mail-klient programvara som kan vara av olika typ.

Hjälp med att förstå vilka krav och behov en e-post användare har på ett ASS kan vi få i enkätundersökningen. Från slutsatserna i undersökningen (tabell 10.2) har nedanstående punkter använts för att formulera en del av slutanvändarens krav.

- De flesta är restriktiva med att lämna ut sin arbetsmejladress på Internet. Den privata mejladressen lämnas däremot ut ganska flitigt!
- De flesta har ett SPAM-filter som skydd. Trots detta får de flesta ganska ofta skräppost (i alla fall privat).
- De flesta använder en vanlig mail-klient när man läser sin mejl, men vanligt är även webbläsare.

Kraven som kan härledas från undersökningen har kompletterats med vanliga krav på ett ASS och prestandakrav på ett vanligt mailsystem utan ASS.

4.2.1 Krav härledda från undersökning

Systemet skall skydda slutanvändarens konto mot oönskad post på ett effektivt sätt trots att användaren lämnat ut sin adress så att ”obehöriga” kan komma åt den på Internet. Systemet måste vara effektivare än dagens vanliga SPAM-filter och det får inte ställa krav på att mail-klienten är av en viss typ.

4.2.2 Vanliga krav på ett ASS

Systemet skall upplevas som ”osynligt” av en slutanvändare och användaren skall inte behöva veta någon om hur systemet installeras eller konfigureras. Det bör dock finnas möjlighet för en slutanvändare att välja hur e-posten sorteras in i olika ”inboxar” beroende på beslut som anti-spam systemet tagit om så önskas. Det bör även gå att ställa in systemet så att misstänkt spam direkt raderas.

4.2.3 Vanliga prestandakrav på en mailserver utan ASS

Eftersom mottagning av brev inte är att betrakta som en synkron process har slutanvändaren inga speciella krav på systemet avseende svarstider. Slutanvändaren vill dock att systemet klarar av att ta emot brev utan onödig fördröjning. Onödig fördröjning kan t ex vara att systemet är så hårt lastat att det inte svarar på tilltal och därför misslyckas med att ta emot eller leverera post.

4.3 Brukare och driftspersonal

Brukare är den person eller det företag/organisation som har programvaran installerad på sin server för att skydda e-postkonton mot spam. Brukaren har kanske personal som installerar programvara på företagets servrar samt övervakar systemen i drift och det är de personerna som avses med intressenten driftspersonal.

Systemet bör vara enkelt att installera och det skall finnas möjlighet att övervaka hur systemet ”mår”. Systemet skall fungera i de flesta miljöer oavsett operativsystem, mail- server och klient. Systemet måste vara säkert, det får inte vara känsligt för DOS (Denial of service) attacker och risken för minnesfel som kan utnyttjas för inbrott måste vara minimerad. En brukare vill kunna lita på att systemet gör det som utlovas.

4.4 Konstruktör

Med konstruktör avses här den eller de personer som är med och bygger (realiserar) ASS enligt detta arkitekturförslag.

Konstruktören vill att designen av systemet skall vara tydlig så att han eller hon förstår hur systemet är tänkt att fungera. Det bör finnas tydliga avgränsningar mellan olika ingående komponenter så att konstruktion av olika delar kan ske parallellt och oberoende av varandra. Tydliga komponenter bidrar också till att underlätta vid enhetsprovning.

4.5 Arkitekt

Med arkitekt avses den person som ansvarar för designen av systemets fundamentala grundstruktur samt att systemet har förutsättningar att uppfylla ställda krav.

Systemets arkitektur måste vara tydlig så att arkitekter förstår grundidén med systemet och hur systemet är tänkt att fungera. Detta underlättar arbetet vid en eventuell framtida utveckling och utbyggnad av systemet och till slut vid systemets avveckling.

4.6 Testare

Testare är den eller de personer som genomför tester för att verifiera systemets funktion och kravuppfyllnad.

En testare vill att systemet skall vara så enkelt som möjligt att prova samt att alla krav skall vara tydligt formulerade och helst också mätbara så att verifieringen av dem är enkel.

4.7 Sammanställning och prioritering av krav

I tabell 4.1 sammanfattas och prioriteras intressenters krav. Kraven är prioriterade enligt bedömning gjord av arkitekten (författaren av detta dokument) och är gjord enligt följande modell:

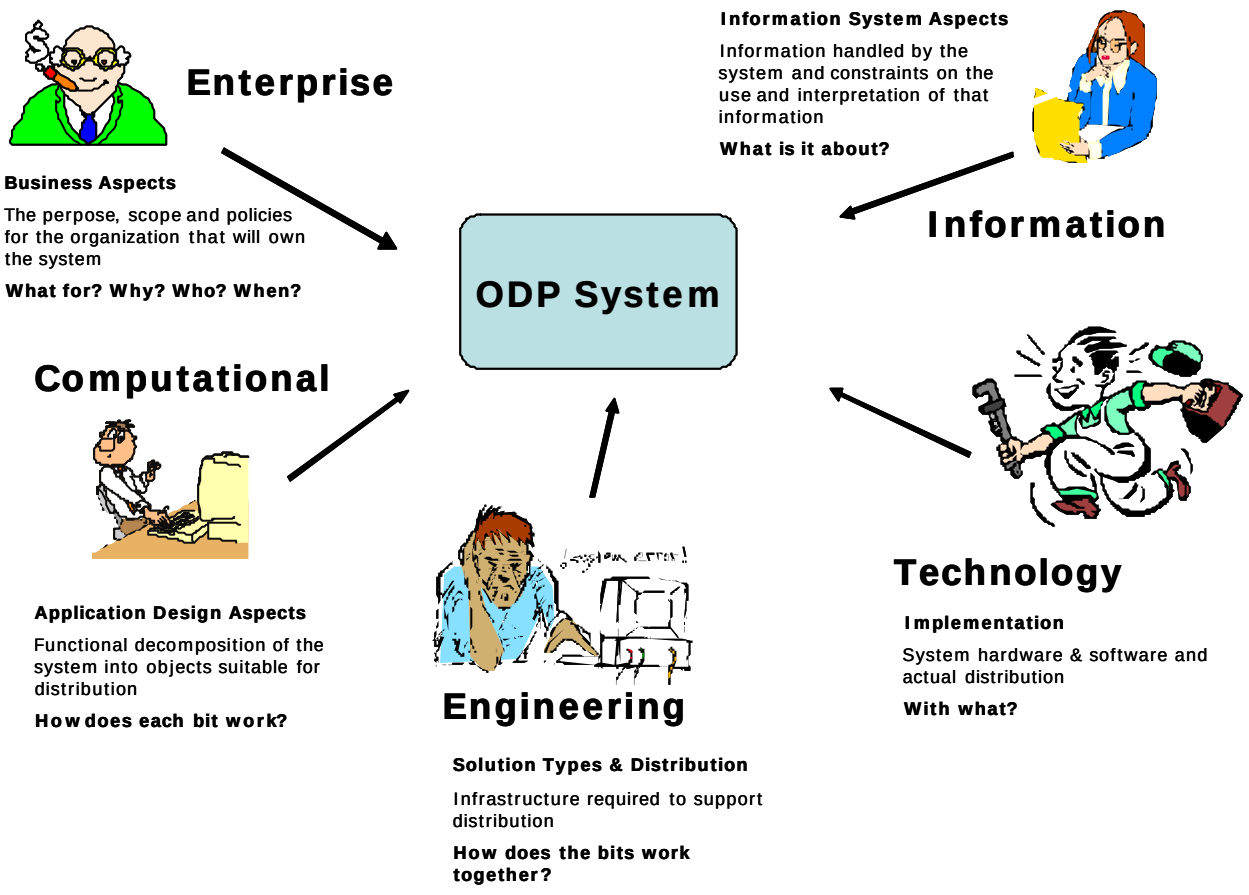
1–Absolut krav, 2–Borde krav, 3-Önskemål.

Id	*Benämning*	*Intressent*	*Funktion/ kvalitetskrav*	*Prio*
K1	*Systemet filtrerar ut oönskad post*	ÄGARE, BRUKARE, SLUTANVÄNDARE	**Funktion**	1
K2	*Systemet är oberoende av typ av: operativsystem, mailserver och mailklient*	ÄGARE, BRUKARE	**Modifierbarhet**	1
K3	*Systemet bygger ej på programvara som kräver att brukaren erlägger licensavgift*	ÄGARE, BRUKARE	**Funktion**	1
K4	*Systemet klarar av att ta emot 6000 brev/dygn fördelat på 200 konton och en genomsnittlig storlek av 100KB.*	ÄGARE, BRUKARE, SLUTANVÄNDARE	**Prestanda**	1
K5	*Systemet klarar av 20 samtidiga sessioner mot levererande servrar*	ÄGARE, BRUKARE	**Prestanda**	1
K6	*Systemet är inte känsligt för DOS-attacker*	BRUKARE	**Säkerhet**	1
K7	*Det går att lita på systemets funktion*	BRUKARE	**Säkerhet**	1
K8	*Det finns möjlighet till sortering av mottagen post till olika inkorgar och direkt radering av misstänkt SPAM*	SLUTANVÄNDARE	**Funktion**	1
K9	*Systemet går att övervaka i drift*	BRUKARE, DRIFTSPERSONAL	**Säkerhet**	1
K10	*Systemet är utbyggbart med externa anti-spam tekniker*	ÄGARE, BRUKARE	**Modifierbarhet**	2
K11	*Systemet är fritt från minnesfel*	BRUKARE	**Säkerhet**	2
K12	*Systemets arkitektur är tydligt beskriven*	ARKITEKT	**Förståbarhet**	2
K13	*Systemets design tydligt beskriven*	KONSTRUKTÖR	**Förståbarhet**	3
K14	*Systemet är enkelt att installera*	BRUKARE, DRIFTSPERSONAL	**Funktion**	3

Tabell 4.1 Prioriterade krav

5. Perspektiv på systemets arkitektur (viewpoints)

Perspektiv och vyer (viewpoints och views på engelska) är ett standardiserat sätt att beskriva arkitekturen för programvara. Enligt standarden ”Recommended Practice for Architectural Description of Software-Intensive Systems” (IEEE 1471) [14] skall ett antal perspektiv väljas ut och beskriva systemet ur olika aspekter och för olika intressenter. Det finns flera olika framtagna referensmodeller som följer IEEE 1471 som man kan använda sig av. I IEEE 1471 anges “Reference Model for Open Distributed Processing” (RM-ODP) [17] som exempel. Ett annat exempel är “The 4+1 View Model of Architecture” [15] av P. Kruchten som ingår i utvecklingsprocessen ”IBM Rational Unified Process” (RUP). I detta dokument kommer vi att beskriva arkitekturen med hjälp av perspektiven som föreslås i RM-ODP (se figur 5.1).



Figur 5.1 RM-ODP perspektiv[1] (viewpoints)

I följande avsnitt går vi igenom varför perspektiven valts genom att beskriva för vem perspektivet främst är avsett och vad vi försöker beskriva i motsvarande vy. Vyerna beskrivs i nästa avsnitt: *6 Vyer av systemets arkitektur (views).*

[1] Beskrivning från Antonio Vallecillo’s presentation av UML for ODP vid WODPEC 2006 i Hong Kong

5.1 Enterprise viewpoint

”Enterprise” perspektivet beskriver normalt hur systemet passar in i företagets affärsprocess. Vid beskrivningen av detta system (ASS) så har vi valt att utelämna detta perspektiv eftersom systemet inte är riktat till någon speciell typ av verksamhet. Systemet skall kunna användas i alla företag och organisationer som använder sig av e-post och vill skydda användarnas konton mot oönskad post.

5.2 Information viewpoint

“Information” perspektivet beskriver informationen som hanteras av systemet. Här beskrivs vilka begrepp det rör sig om samt hur de är relaterade till varandra. Här beskrivs också hur informationen flödar genom systemet. I detta dokument är ”Information” perspektivet uppdelat i två olika abstraktionsnivåer, konceptuellt och logiskt. Den konceptuella nivån är främst riktad till ägare/beställare samt slutanvändare, brukare och testare. På logisk nivå riktar sig perspektivet till utvecklare, arkitekter och testare.

5.3 Computational viewpoint

”Computational” perspektivet beskriver hur applikationen är designad ur ett systemutvecklingsperspektiv. Här beskrivs hur systemet uppdelat funktionellt i delsystem och hur dessa delsystem realiseras av komponenter och hur komponenterna fungerar. Här beskrivs också gränssnitten som används för kommunikation mellan komponenter och externa system. Detta perspektiv är riktat till utvecklare och arkitekter.

5.4 Engineering viewpoint

”Engineering” perspektivet beskriver infrastrukturen som krävs runt systemet för att systemet skall fungera. Här beskrivs hur systemet installeras och driftsätts samt hur systemet är distribuerat och beroenden till externa system och komponenter. Protokoll som används för kommunikation mellan systemet och externa system beskrivs också här. Perspektivet riktar sig i första hand till brukare, driftspersonal och testare men också till utvecklare och arkitekter.

5.5 Technology viewpoint

”Technology” perspektivet beskriver normalt hårdvaran och mjukvaran närmast hårdvaran där man valt att köra systemet. Vid beskrivningen av detta system (ASS) så har vi valt att utelämna detta perspektiv eftersom systemet skall kunna köras oberoende av plattform.

6. Vyer av systemets arkitektur (views)

I detta avsnitt beskrivs vyerna för de perspektiv vi valt för att beskriva arkitekturförslaget. Varje vy beskriver systemet ur en eller flera intressenters perspektiv.

6.1 Enterprise view

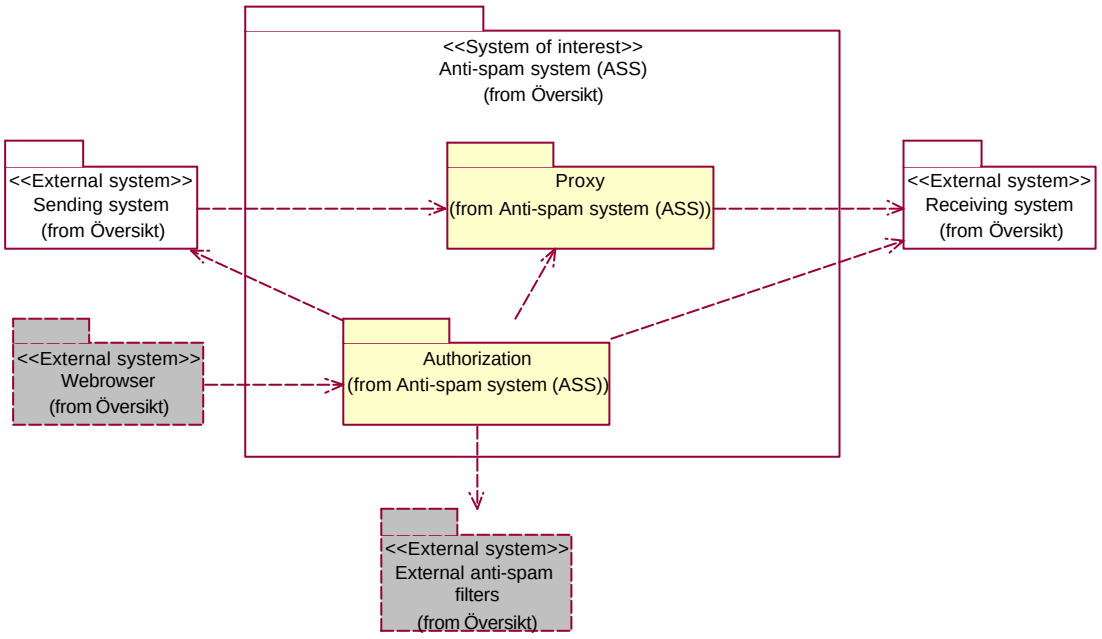
Denna vy beskrivs inte i detta dokument eftersom systemet inte är knutet till en specifik verksamhet. Systemet kan användas av alla verksamheter som använder sig av Internet mail.

6.2 Information view (konceptuell nivå)

I denna vy går vi igenom informationen som hanteras av systemet samt hur systemet är uppbyggt på en konceptuell och översiktlig nivå. Vyn beskriver också konceptet för hur systemet fungerar genom att beskriva systemets dynamik i en flödesmodell.

6.2.1 Översikt av systemet och dess omgivande miljö

Modellen i Figur 6.1 nedan visar anti-spam systemet med dess huvudsakliga ingående delar och hur det placeras in i sin omgivande miljö med externa system till vilka systemet har beroenden.



Figur 6.1 Anti-spam systemet och dess omgivande miljö

Sending system

Avsändaren använder valfritt program (mail-klient) för att skapa och skicka brev till en mottagare. Breven skickas från avsändaren via en reläande mail server och tas emot av ASS.

Anti-spam system (ASS)

ASS lagrar alla brev tills avsändaren godkänts. När en avsändare är godkänd vidarebefordras brevet till mottagarens system.

Receiving system

Mottagaren tar emot brev från sin mailserver med hjälp av sin mail-klient.

Webbrowser

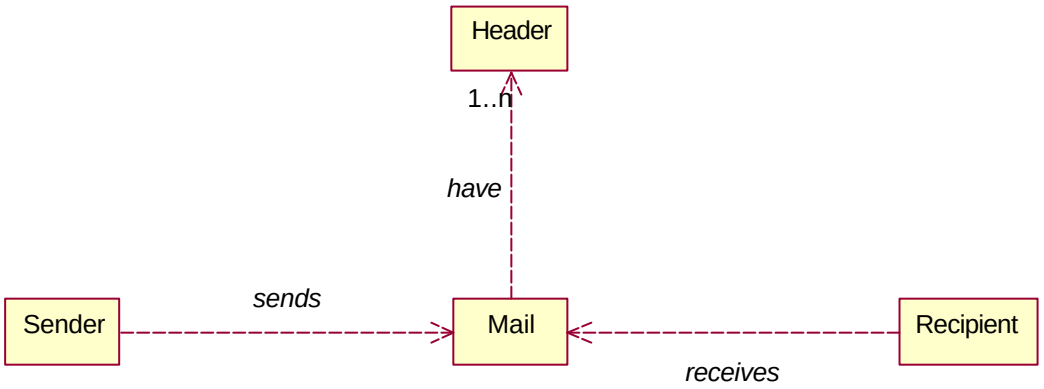
En webbläsare används av avsändaren för att auktorisera sig genom att ange svar på den utmaning som ASS skickar till avsändaren första gången som denne skickar post till aktuell mottagare.

External anti-spam filters

Systemet kan byggas ut med externa filtreringsmekanismer för att komplettera systemets egen filtrering av SPAM.

6.2.2 Statisk entitetsmodell

Modellen i Figur 6.2 nedan visar benämningar och relationer mellan de viktigaste entiteterna som hanteras av anti-spam systemet.



Figur 6.2 Konceptuell informationsmodell

Mail

Ett elektroniskt brev, även kallat e-post, e-meddelande eller e-brev.

Egenskaper: Header(s), Text, Bifogad(e) filer, Mottagare, Avsändare

Header

Ett brev innehåller en eller oftast flera headerfält. Dessa fält är en del av SMTP protokollet och behövs av e-postservrar som förmedlar posten över Internet. Många system lägger till egna headerfält för eget bruk, dessa behöver inte vara en del av protokollet och prefixas därför med ”X-”.

Egenskaper: Namn, Värde

Sender

Avsändare. Den person som skapar och skickar ett brev.

Egenskaper: Namn, Adress

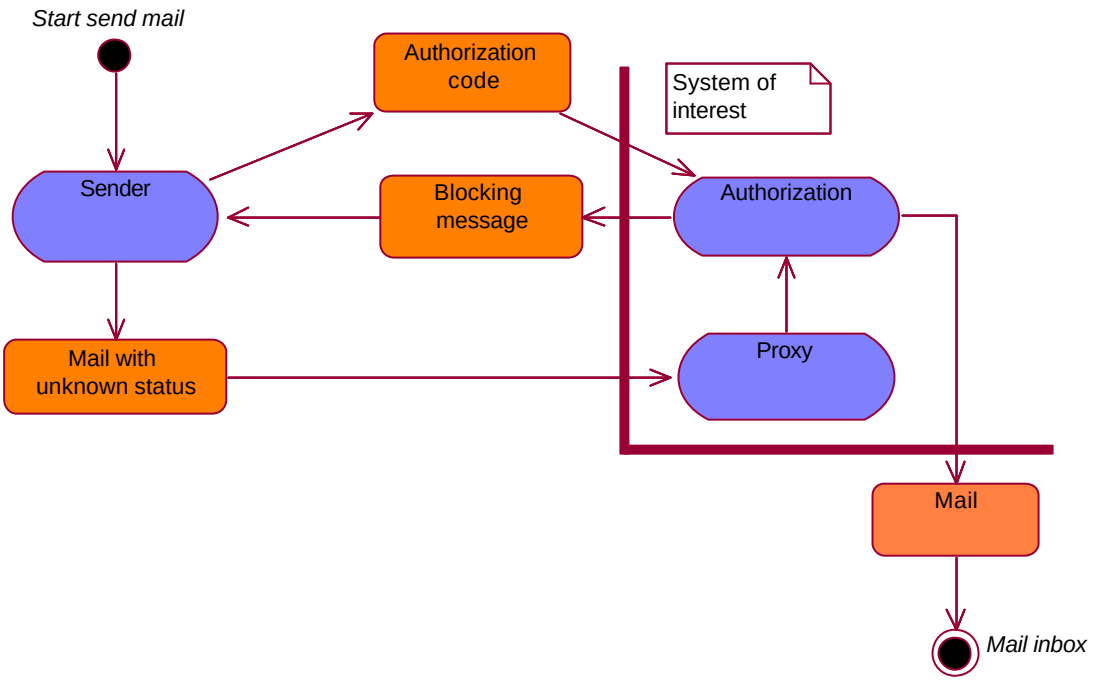
Recipient

Mottagare. Den person som brevet är adresserat till av avsändaren.

Egenskaper: Namn, Adress

6.2.3 Översiktlig flödesmodell

I Figur 6.3 nedan beskrivs översiktligt hur anti-spam systemet fungerar. Avsändaren skickar ett brev till en mottagare vars konto skyddas av anti-spam systemet. Brevet tas emot och lagras men skickas inte vidare till mottagaren. Ett blockeringsbrev (Figur 3.3) skickas till avsändaren med uppmaning om att auktorisera sig genom att ange en kod som är renderad i en Captcha bild. Eftersom koden visas som en bild är det svårt att på maskinell väg läsa av koden. När avsändaren angivit rätt kod vidarebefordras brevet till mottagaren. Nästa gång avsändaren skickar ett brev till samma mottagare är avsändaren godkänd och behöver därför inte gå igenom proceduren på nytt.



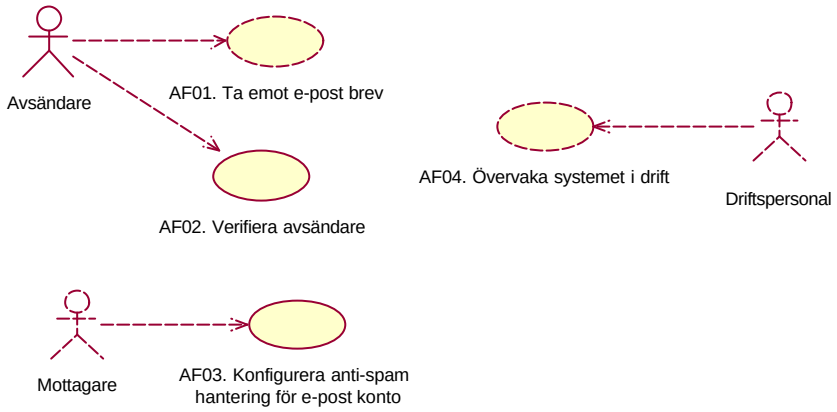
Figur 6.3 Översiktlig flödesmodell för Anti-spam systemet

6.3 Information view (logisk nivå)

I den här vyn beskriver vi systemet genom att gå igenom vilka användningsfall som valts ut som systemets kritiska användningsfall (användningsfall med arkitektonisk påverkan). De kritiska användningsfallen beskrivs i detaljerade scenarier på logisk nivå.

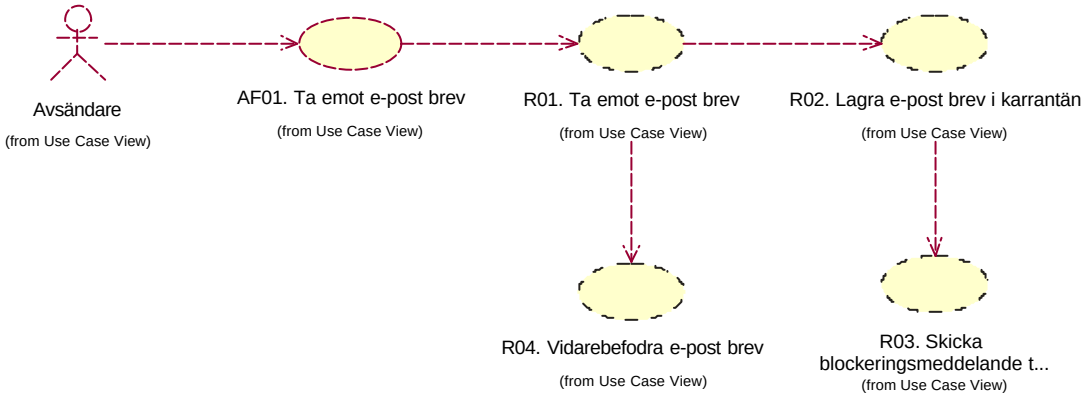
6.3.1 Kritiska användningsfall

Systemet har fyra användningsfall (AF): ”Ta emot e-post brev” (AF01), ”Verifiera avsändare” (AF02), ”Konfigurera anti-spam hantering för e-post konto” (AF03) och ”Övervaka systemet i drift” (AF04). Av dessa användningsfall är AF01, AF02 och AF03 speciellt viktiga för arkitekturen.

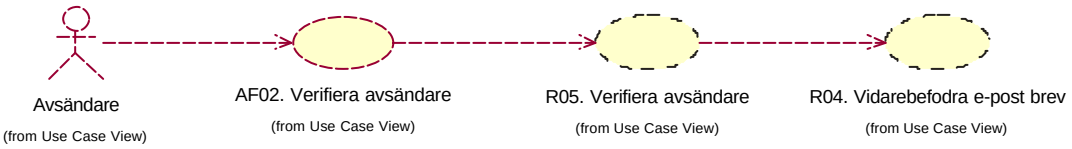


Figur 6.4 Användningsfallsöversikt

AF01 och AF02 är speciellt viktiga ur ett arkitektoniskt perspektiv eftersom de representerar själva fundamentet för systemet. Du hittar beskrivningen av dem under punkterna *6.3.2* och *6.3.3* och under *6.3.4 Vidarebefordra e-post brev* beskrivs ett scenario som båda dessa AF har gemensamt (se figur 6.5 och 6.6).



Figur 6.5 Nedbrytning av AF01 ”Ta emot e-post brev” i realiseringar

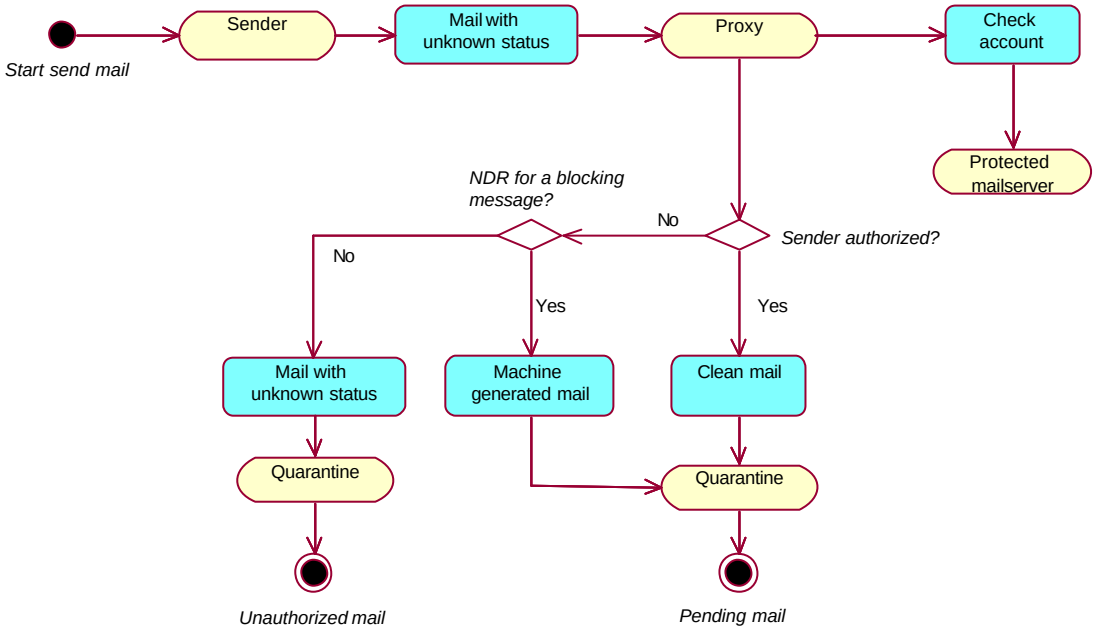


Figur 6.6 Nedbrytning av AF02 ”Verifiera avsändare” i realiseringar

AF03 har valts ut som ett kritiskt användningsfall eftersom ett av kraven [K2] säger att systemet måste vara oberoende av programvara på klienten.

6.3.2 *Ta emot e-post brev (AF01)*

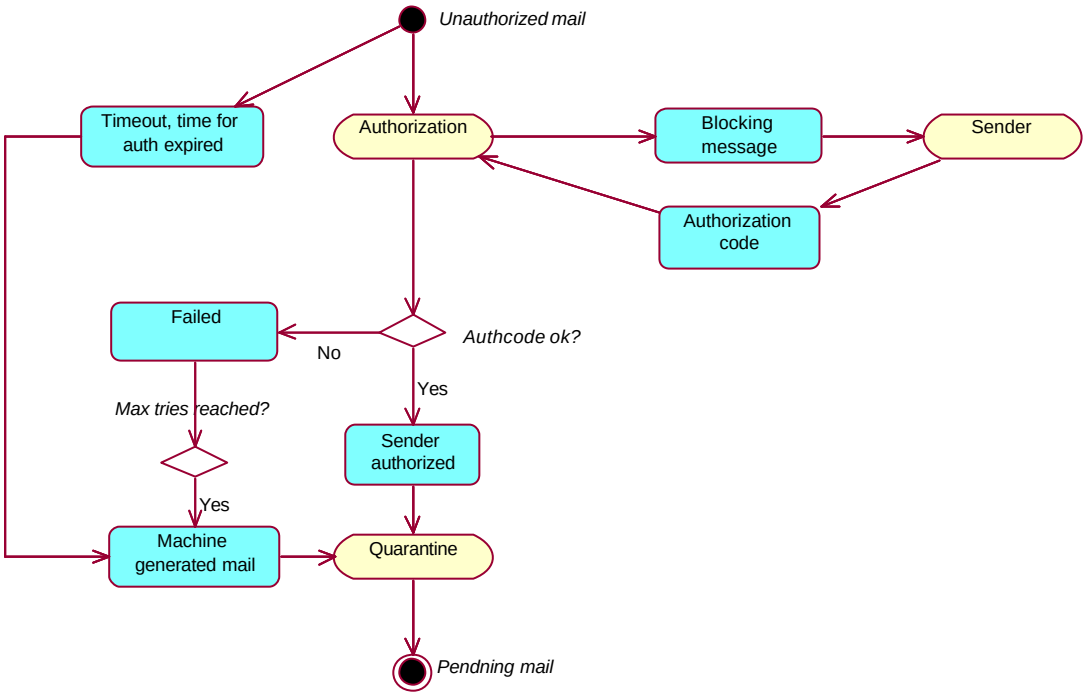
All post som skall till den skyddade mailservern tas om hand av anti-spamsystemets proxy där den lagras innan den vidarebefordras till mottagaren. När en reläande mailserver begär att få lämna ett brev kontrollerar först proxyn den skyddade mailservern om aktuellt konto finns. Om avsändaren tidigare skickat post till mottagaren och då blivit godkänd (eller om avsändaren vitlistats) märks brevet för att skickas vidare till mottagaren via den skyddade mailservern. Om brevet är ett NDR (Non Delivery Report) för ett tidigare skickat blockeringsmeddelande markeras avsett brev som ”maskingenererat”. Posten lagras efter mottagning i ”karantänen” i väntan på utskick av blockeringsmeddelande eller annan vidare behandling.



Figur 6.7 Ta emot e-post brev

6.3.3 *Verifiera avsändare (AF02)*

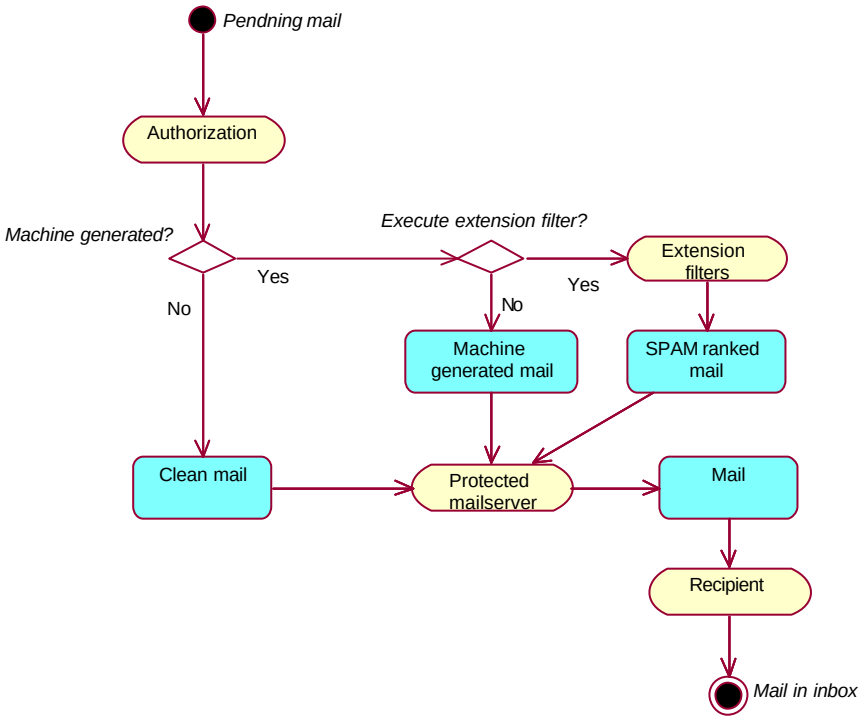
För all post som tagits emot av anti-spam systemet skickas ett blockeringsmeddelande till avsändaren. Avsändaren kan sedan auktorisera sig genom att skicka in rätt kod via en http-POST. Om avsändaren anger rätt kod kommer brevet att märkas för att skickas vidare till mottagaren. En avsändare har begränsad tid och begränsat antal försök på sig att auktorisera sig, när denna tid gått ut eller när max antal försök genomförts anser systemet att skickat brev är maskingenererat.



Figur 6.8 Verifiera avsändare

6.3.4 *Vidarebefordra e-post brev (R04)*

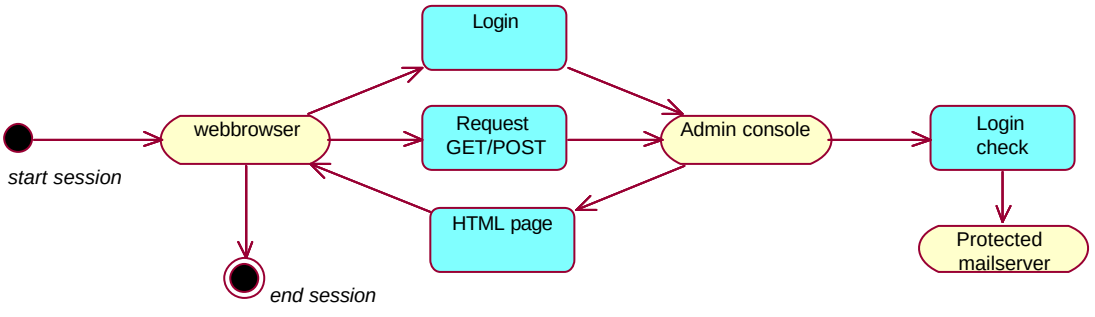
När avsändaren för ett brev blivit auktoriserad eller när tiden för detta gått ut vidarebefordras brevet till mottagaren. Mottagaren kan konfigurera sitt konto så att all post som inte godkänts via auktoriseringsförfarandet direkt raderas utan att skickas vidare. Om systemet är konfigurerat för att köra andra filtreringsmekanismer på brevet görs det för ett meddelande som anses vara maskingenererat för att vidare försöka avgöra om brevet kan anses vara skräppost.



Figur 6.9 Vidarebefordra e-post brev

6.3.5 *Konfigurera anti-spam hantering för e-post konto (AF03)*

När en användare (mottagaren) vill ändra inställningar i anti-spam systemet för sitt konto surfar han eller hon in i systemet med hjälp av en webbläsare. Användaren använder samma logininformation som när e-post skall hämtas till inkorgen från mailservern via POP3 eller IMAP protokollen. Systemet har alltså inte en separat kontohantering för varje användare, användarnamn (mailadress) och lösenord kontrolleras med hjälp av den skyddade mailservern.



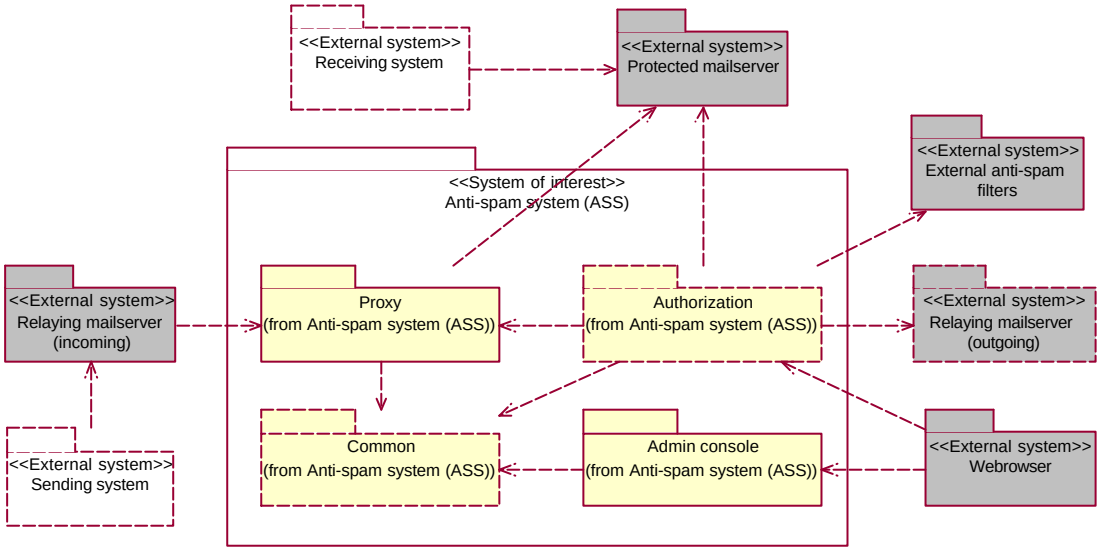
Figur 6.10 Konfigurera anti-spam hantering för e-post konto

6.4 Computational view

I denna vy går vi först igenom systemet översiktligt genom att illustrera beroenden till externa system och hur systemet grovt är indelat i delsystem. Ansvaret för varje delsystem beskrivs och de externa systemen som systemet är beroende av beskrivs kort. Vidare beskrivs hur delsystemen byggs upp av komponenter och principerna för hur dessa fungerar och kommunicerar med varandra samt vilka protokoll som används. Här beskrivs val av programspråk och motivering till detta val. Tredjeparts komponenter som används i systemet beskrivs också kort i denna vy.

6.4.1 Systemets uppdelning i delsystem och beroenden till externa system

I figur 6.11 illustreras systemets beroenden till externa system och hur systemet grovt är indelat i fyra delsystem. Delsystemens ansvar och de externa beroendena beskrivs kortfattat under egna rubriker först i detta avsnitt och längre fram beskrivs hur delsystemen bryts ner i komponenter och hur dessa fungerar.



Figur 6.11 Systemets delsystem och beroenden till externa system

6.4.2 Delsystem

Proxy

Ansvarar för att ta emot e-post från levererande servrar och lagra posten tills den kontrollerats. Fungerar som en proxy mellan den levererande servern och den mottagande servern.

Authorization

Ansvarar för att auktorisera avsändare och vidarebefordra e-post till den mottagande servern samt integration med andra externa anti-spam filtreringssystem och tekniker.

Common

Innehåller funktionalitet som är gemensam för alla delsystem.

Admin console

Ansvarar för att erbjuda ett gränssnitt mot användaren (mottagaren) så att inställningar för hur anti-spam hanteringen skall ske för ett e-post konto kan administreras.

6.4.3 Externa system

Protected mailserver

Mottagande server med konton som anti-spam systemet skall skydda.

Relaying mailserver (incoming)

Levererande mailserver som vidarebefordrar post från avsändaren. Denna server kan finnas hos avsändaren men vanligast är att det är en ”reläande” server som finns hos avsändarens ISP (Internet service provider).

Relaying mailserver (outgoing)

Sändande mailserver som vidarebefordrar blockeringsmeddelanden från anti-spam systemet. Denna server kan vara samma som den skyddade men vanligast är att man måste använda en ”reläande” server som finns hos mottagarens ISP (Internet service provider).

Webbrowser

“Surfkanal” där avsändare auktoriserar sig och mottagare kan administrera sina inställningar för ett konto. Accessen sker med en vanlig webbläsare.

External anti-spam filters

Externa anti-spam filtreringssystem eller tekniker som har integrerats i systemet.

6.4.4 Val av programmeringsspråk

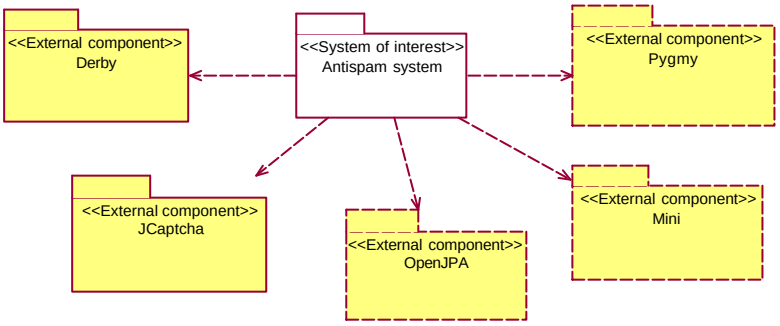
Systemet implementeras i programmeringsspråket Java. Eftersom systemet skall kunna köras oberoende av miljön [K2] så det är lämpligt att välja ett programmeringsspråk som stödjer någon form av virtuell maskin. Det finns flera alternativa programmeringsspråk som stödjer att bygga plattformsoberoende system. Vi väljer Java eftersom det är ett mycket populärt språk och tillgången av fria tredjepartskomponenter är stort.

6.4.5 Funktionalitet realiserad av externa komponenter

Systemet innehåller funktionalitet som inte behöver realiseras av systemet eftersom funktionaliteten kan återanvändas genom att använda färdiga tredjepartskomponenter (kommersiella eller opensource). Eftersom systemet skall kunna distribueras fritt [K3] användes uteslutande opensource komponenter. Funktionalitet som behövs i systemet och kan erbjudas av tredjepartskomponenter är:

- Persistent lagring av data i en relationsdatabas
- Mappning av java-objekt till tabeller i en relationsdatabas
- Hantering av captcha-bilder, rendering och kontroll
- HTTP-protokollet, grundfunktionalitet för en webserver
- Generering av dynamiska HTML-sidor

I följande avsnitt beskrivs kort de komponenter som valts i detta arkitekturförslag för att realisera ovanstående funktionsområden (se även figur 6.12).



Figur 6.12. Externa systemberoenden

Derby komponent

Derby [11] är en opensource relationsdatabas med stöd för SQL och inbäddade databaser.

JCaptcha komponent

JCaptcha [10] är en opensource komponent för rendering och kontroll av Captcha bilder.

OpenJPA komponent

OpenJPA [12] är en opensource komponent som implementerar JPA (Java Persistence API). JPA används för att mappa java-objekt till tabeller i en relationsdatabas utan att manuellt skriva SQL.

Pygmy komponent

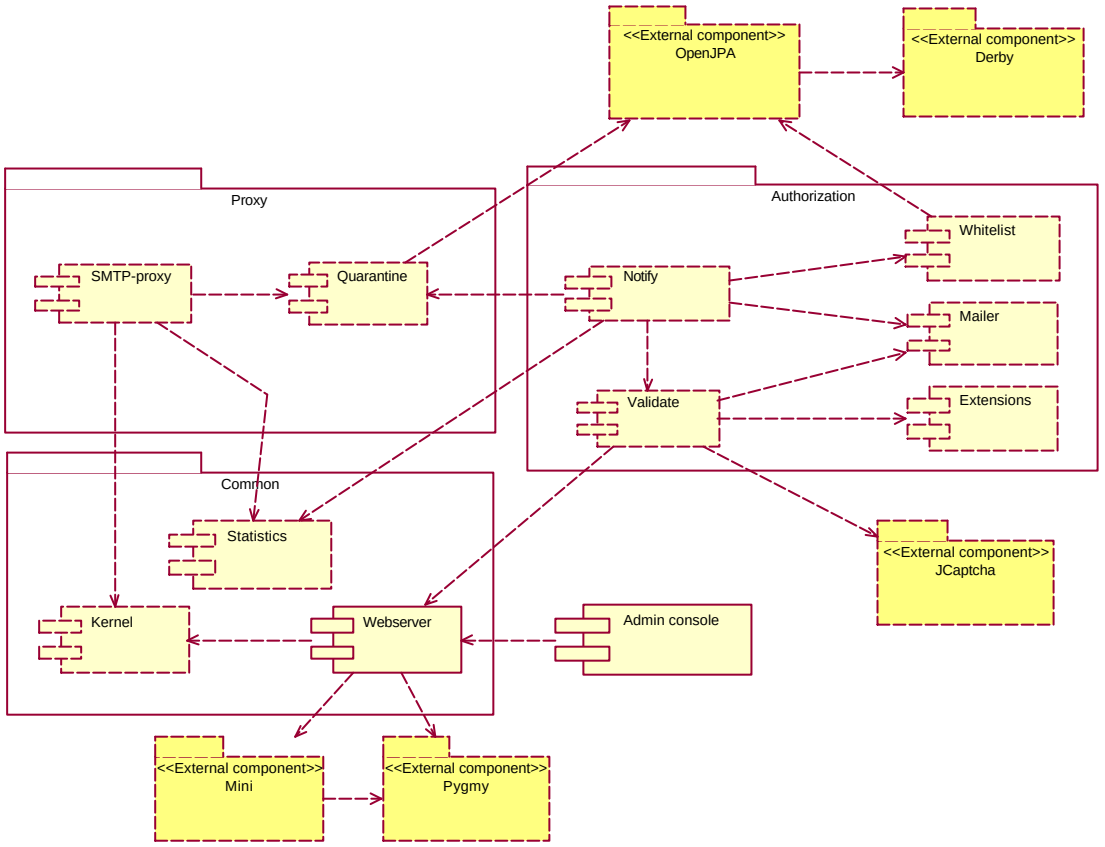
Pygmy [13] är en opensource komponent som innehåller grundläggande HTTP-server funktionalitet.

Mini komponent

Mini [20] är en enkel applikationsserver som ursprungligen designades för att bygga små och resurssnåla webapplikationer i C++. Senaste versionen stödjer Java och bygger på Pygmy. Grundfunktionaliteten i Mini är en enkel modell för generering av HTML-sidor med dynamiskt innehåll från statiska mallar (jfr. ASP och JSP).

6.4.6 *Systemets ingående komponenter*

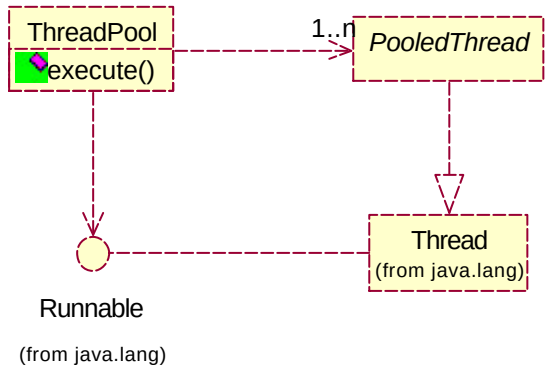
I modellen (Figur 6.13) nedan illustreras systemets ingående komponenter och relationerna mellan dem. I kommande avsnitt beskrivs vilket ansvar dessa komponenter har i systemet. I figuren kan du också se hur olika delar av systemet är beroende av tredjeparts komponenter som beskrivits i föregående avsnitt.



Figur 6.13. Anti-spam systemets ingående komponenter

Kernel komponent

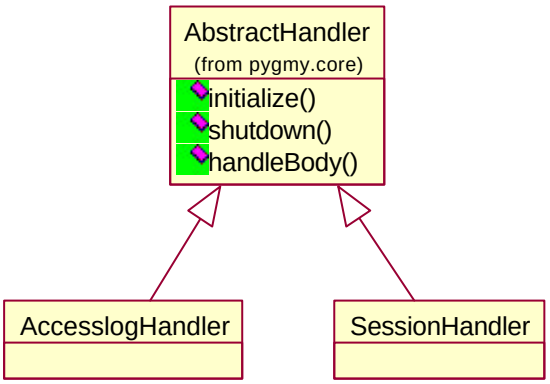
Ansvarar för start och stopp av systemet samt hantering av systemets trådar. Tillhandahåller trådpooler som används av övriga komponenter som behöver trådas av. Flera delar av systemet har behov av trådar för samtidig exekvering. Ett exempel på detta är att systemet skall kunna ta emot post från flera levererande servar samtidigt [K5].



Figur 6.14 Trådpool

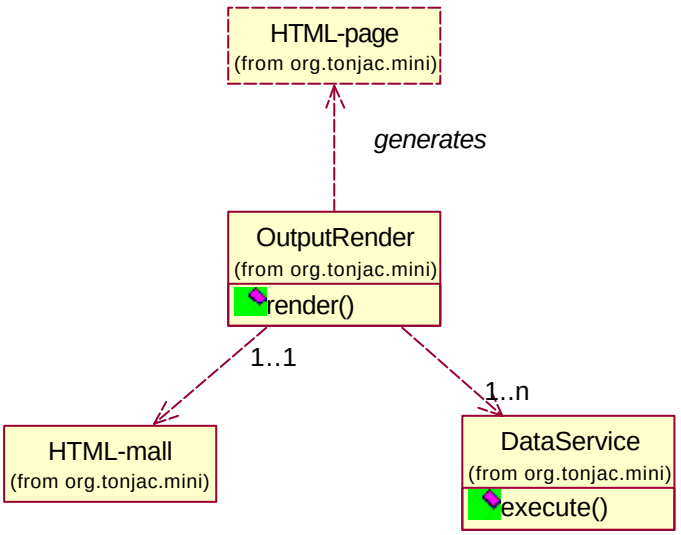
Webserver komponent

Webbserverfunktionalitet som implementeras med hjälp av funktionalitet i Pygmy och Mini som bas[1]. Webbserver komponenten ansvarar för access-loggning, autentisering av användare och generering av HTML-sidor. Access-loggning sker enligt W3C-standarden för loggning på webservrar [18]. Autentisering av användare sker enligt RFC 2617 ”HTTP Authentication” [19]. Användare och lösenord används från konton på den skyddade mailservern, ingen egen kontohantering finns i anti-spam systemet. Kontroll vid inloggning sker enligt protokollet POP3 eller IMAP.



Figur 6.15 Pygmy-handlers för access-loggning och http-sessioner

Generering av HTML-sidor sker enligt modellen i Mini. Mini applikations server har en enkel modell för uppbyggnad av HTML-sidor med dynamiskt innehåll. Modellen bygger på tre beståndsdelar; HTML-mall, datatjänst och renderare.



Figur 6.16 Modell (Mini) för generering av HTML-sidor med dynamisk innehåll

[1] Alternativ till ”Pygmy/Mini” skulle kunna vara Tomcat mfl. Valet föll på Pygmy/Mini eftersom denna kombination ger ett litet fotavtryck (både i storlek och minne) samt att konfigurationen är mycket enkel och samtidigt flexibel.

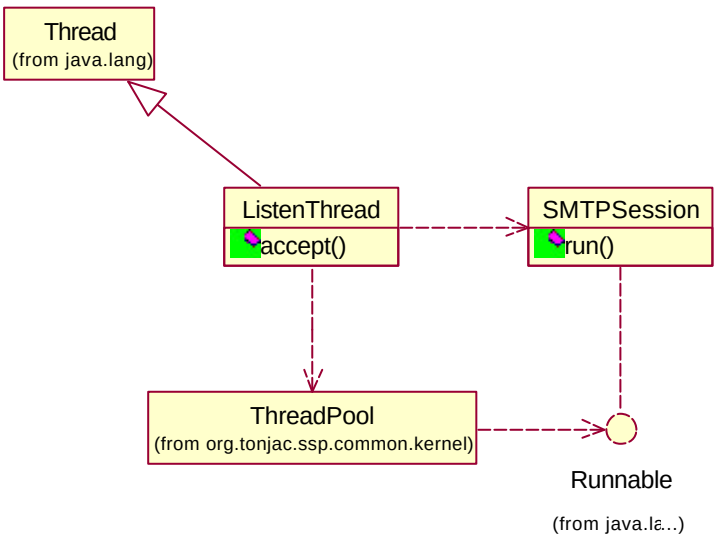
Statistics komponent

Statistik komponenten ansvarar för att bokföra statistik i systemet. Statistiken omfattar att logga alla viktiga händelser i systemet. Dessa händelser kan övervakas av systemets brukare [K9]. För att åtkomst att läsa loggen skall vara så enkel som möjligt loggas informationen i en vanlig ASCII-textfil. Exempel på viktiga händelser är:

- Meddelande mottaget
- Blockeringsmeddelande skickat
- Auktorisering genomförd
- Auktorisering misslyckades
- Resultat från extern filter mekanism
- Auktoriseringstiden har gått ut
- Meddelande vidarebefordrat
- Meddelandet raderades

SMTP-proxy komponent

Lyssnar på inkommande SMTP-anrop och fungerar som en proxy mellan avsändande mailserver och den skyddade mailservern. Varje initierad session hanteras av en ny tråd så att flera servrar kan leverera post samtidigt [K5].



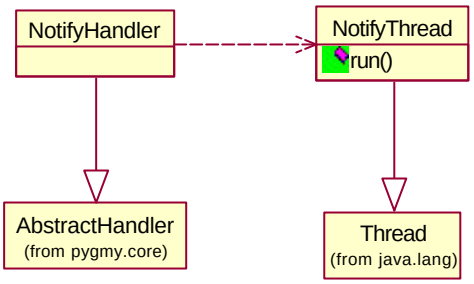
Figur 6.17 SMTP-proxy

Quarantine komponent

Ett lagringsställe där posten lagras och ”ligger i karantän” tills avsändaren auktoriserat sig, tills ett NDR tagits emot eller tills tiden för auktorisationen har gått ut.

Notify komponent

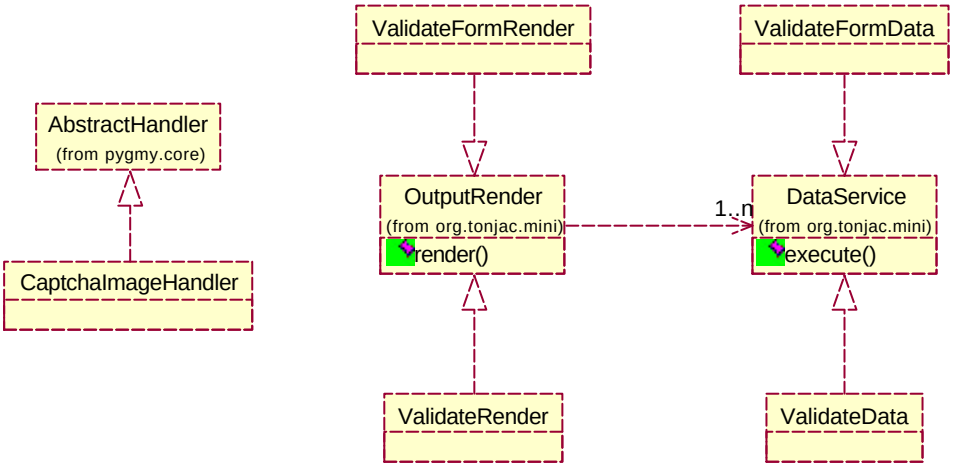
Ansvarar för att hämta upp meddelanden som lagrats i karantänen och antingen direkt vidarebefordra meddelandet om avsändaren är godkänd (finns i vitlistan) eller begära auktorisation av avsändaren [K1].



Figur 6.18 Pygmy-handler för notifieringstråden

Validate komponent

Komponenten ansvarar för att skapa blockeringsmeddelanden samt att kontrollera auktorisationsförsök.



Figur 6.19 Pygmy-handler för captcha-renderaren och Mini-struktur för generering av blockerings-meddelande n och validering

När ett blockeringsmeddelande skapas genereras ett globalt unikt id som kopplas till mottaget brev i ett extra SMTP-headerfält: ”X-SSP-ID” (se figur 6.19a-c nedan). Id:t används för att knyta ihop framtida auktorisationsförsök med mottaget brev. Auktorisationsförsök initieras genom en http-POST från avsändaren.

När kontroll av ett meddelande gjorts märks meddelandet med bedömningen som anti-spam systemet gjort. Bedömningen avser om brevet anses vara maskingenererat eller inte. För maskingenererade meddelanden kan även bedömningen omfatta en spam-ranking från en extern filtermekanism. Bedömningen lagras i ett extra SMTP-headerfält; ”X-SSP-RESULT”, och eventuellt som ett tillägg på meddelandets ”Subject”, se exempel nedan. Informationen i ”Subject” eller headerfält kan användas av användaren för att sortera posten till olika ”inkorgar” [K8].

```
X-SSP-ID: 1E7CFA1095.11482830C3D.E8000.0000001
X-SSP-RESULT: Machine-generated, 7
Subject: [Machine-generated][7] Detta är ett exempel
...
```

Figur 6.19a Exempel på del av listning av ett maskingenererat brev som är kontrollerat av ASS

```
X-SSP-ID: 1E7CFA1095.11482830C3D.E8000.0000002
X-SSP-RESULT: Machine-generated
Subject: Detta är ett exempel på ett maskingenererat brev
...
```

Figur 6.19b Exempel på del av listning av ett maskingenererat brev som är kontrollerat av ASS

```
X-SSP-ID: 1E7CFA1095.11482830C3D.E8000.0000003
X-SSP-RESULT: OK
Subject: Detta är ett exempel på ett ok brev
...
```

Figur 6.19c Exempel på del av listning av ett brev som klarat kontrollen i ASS

Mailer komponent

Denna komponent ansvarar för att skicka post enligt SMTP-protokollet.

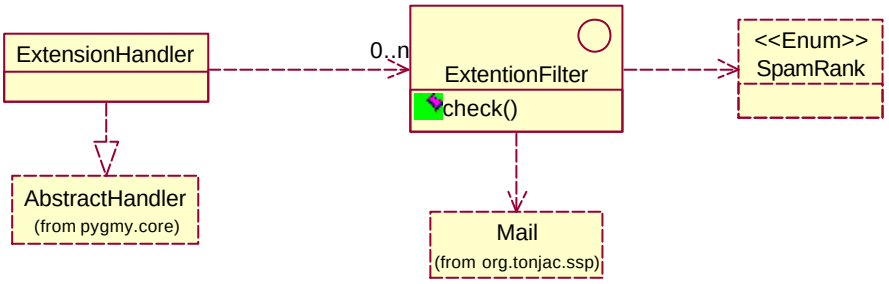
Whitelist komponent

Ett lagringsställe för godkända avsändare som auktoriserat sig via systemet eller som vitlistats av mottagaren.

Extensions komponent

Komponent för hantering av inkoppling av andra spam-filtreringsmekanismer [K10] som kan användas för att bygga på systemet. Externa filtreringsmekanismer som skall kopplas in kapslas av kod som implementerar gränssnittet *ExtensionFilter*:

```
public interface ExtensionFilter
{
      SpamRank check( Mail mail );
}
```



Figur 6.20 Hantering av utbyggnadsfilter

Externa filtreringsmekanismer laddas dynamiskt via arkiv som installeras under katalogen ”extension” på filsystemet under installationspunkten av ASS.

Admin console komponent

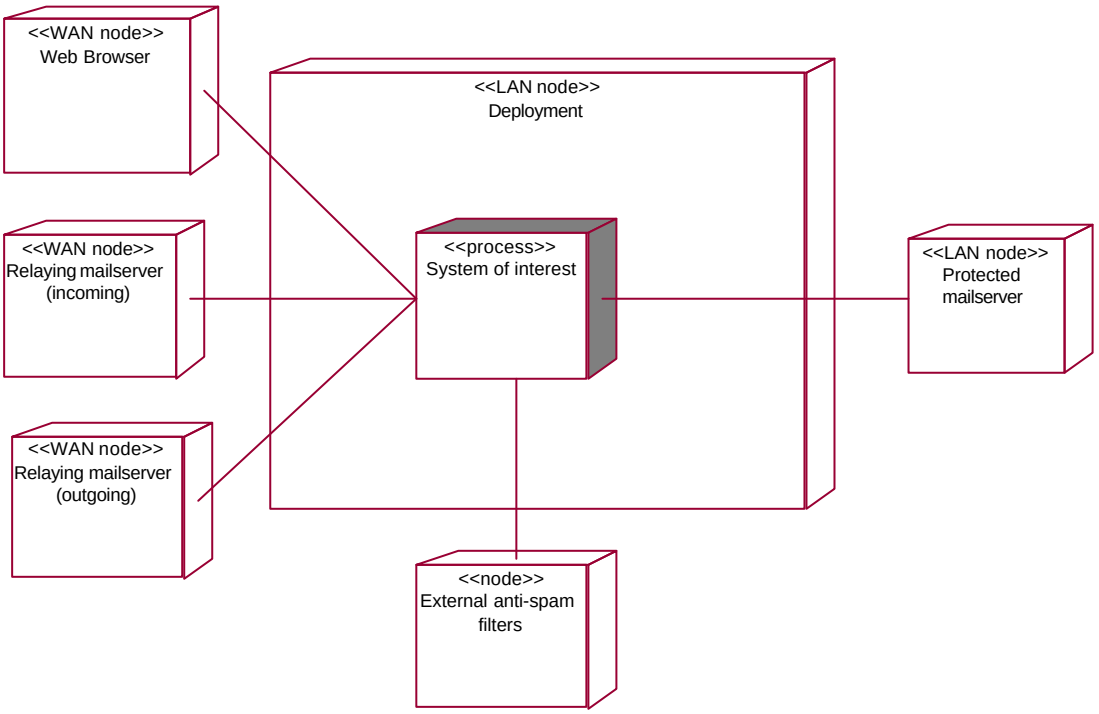
Webapplikation som används för att göra förändringar i systemet som t ex ändring av inställningar för ett e-post-konto. Applikationen byggs enligt ”Mini-konceptet” för generering av dynamiska HTML-sidor.

6.5 Engineering view

I den här vyn beskrivs infrastrukturen som krävs för att systemet skall fungera när det är driftsatt. Här beskrivs hur systemet och externa system (som det finns beroenden till) är distribuerade på olika nätverksnoder. Här beskrivs systemets processer och trådar samt vilka portar och protokoll som används av systemet.

6.5.1 Processer och nätverksnoder

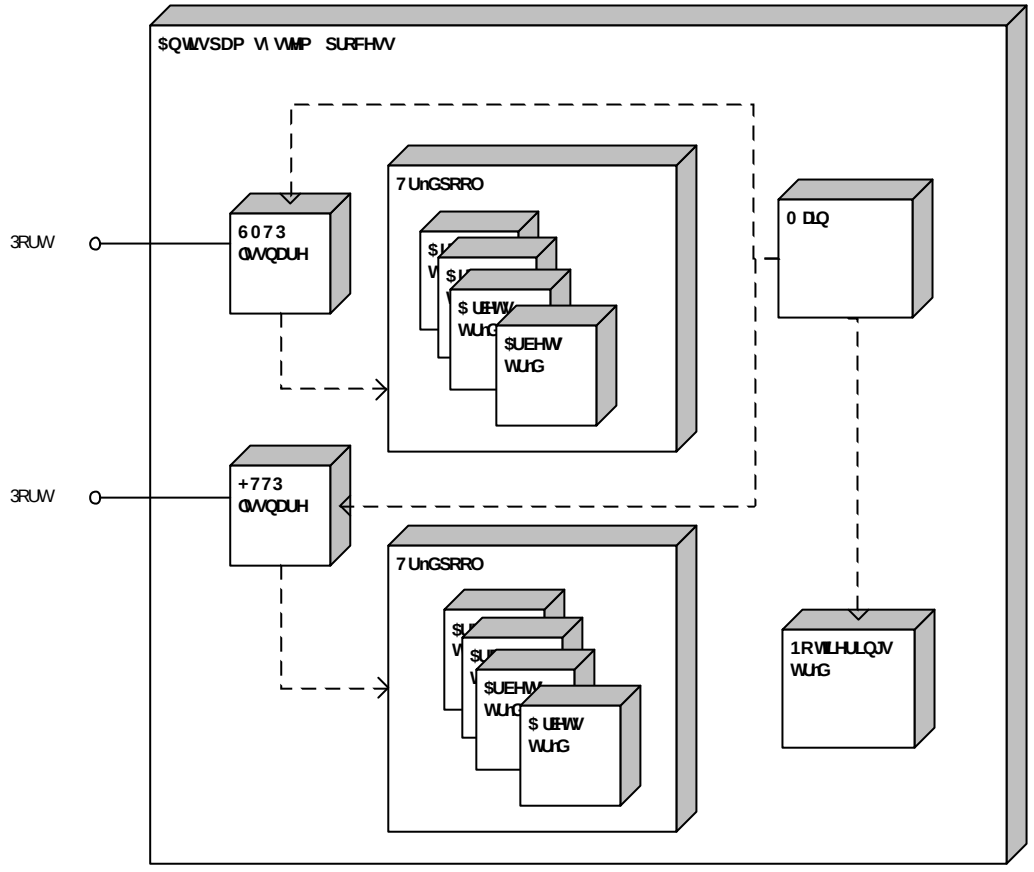
Anti-spam systemet består av en JVM-process (Java virtual machine) som kan köras på en nätverksnod som är fristående från mailservern som skall skyddas. Anti-spam systemet kan även köras på samma nod som den skyddade mailservern under förutsättning att TCP-porten som servern lyssnar på kan konfigureras till någon annan port än 25 (smtp) eftersom anti-spam systemet måste ta denna port i anspråk. Anti-spam systemet lyssnar efter inkommande auktorisationsförsök på port 80 (http) från WAN (Wide area network). Om anti-spam systemet byggs ut med andra externa anti-spam system eller tekniker kan dessa exekvera på andra nätverksnoder inom eller utanför aktuellt LAN (Local area network). Anti-spam systemet skickar blockeringsmeddelanden till avsändare och behöver därför tillgång till en mailserver som kan vidarebefordra e-post utanför aktuellt LAN. Denna mailserver kan vara samma som den skyddande mailservern om den stödjer ”relay”, men troligtvis krävs att man använder mottagarens ISP:s relay-server.



Figur 6.21 Processer och nätverksnoder

6.5.2 Systemets trådar

Systemet (Figur 6.22) har fyra grundtrådar. Förutom huvudtråden finns två trådar som lyssnar efter http- resp. SMTP anrop och en tråd som ansvarar för att vidarebefordra av post. Utöver grundtrådarna finns ett konfigurerbart antal arbetstrådar i två olika pooler som utför arbetet enligt HTTP och SMTP protokollen.



Figur 6.22 Systemets trådar

Hudvudtråd (main)

Systemets huvudtråd ansvarar för att starta upp systemet. Vid uppstart startas trådpoolerna, SMTP- och HTTP-lyssnartrådarna samt notifieringstråden. När alla trådar startats väntar huvudtråden på en signal om stopp.

HTTP lyssnartråd

HTTP-lyssnartråden accepterar anrop på port 80 enligt http-protokollet. Varje http-anrop (GET eller POST) tas om hand av en ledig tråd (arbetstråd) i trådpoolen.

SMTP lyssnartråd

SMTP-lyssnartråden accepterar anrop på port 25 enligt SMTP-protokollet. Varje SMTP-session tas om hand av en ledig tråd (arbetstråd) i trådpoolen.

Notifieringstråd

Notifieringstråden ansvarar för att vidarebefordra post där avsändaren auktoriserat sig eller där tiden för detta har gått ut. Tråden ansvarar också för att skicka ut auktoriseringsbegäran till alla avsändare.

6.5.3 Transaktioner

Transaktionerna i systemet kodas manuellt med hjälp av JPA. Alla transaktioner måste synkroniseras med SMTP-protokollet så att innehållet i databasen är konsistent med vad tagits emot via SMTP.

6.5.4 Säkerhet

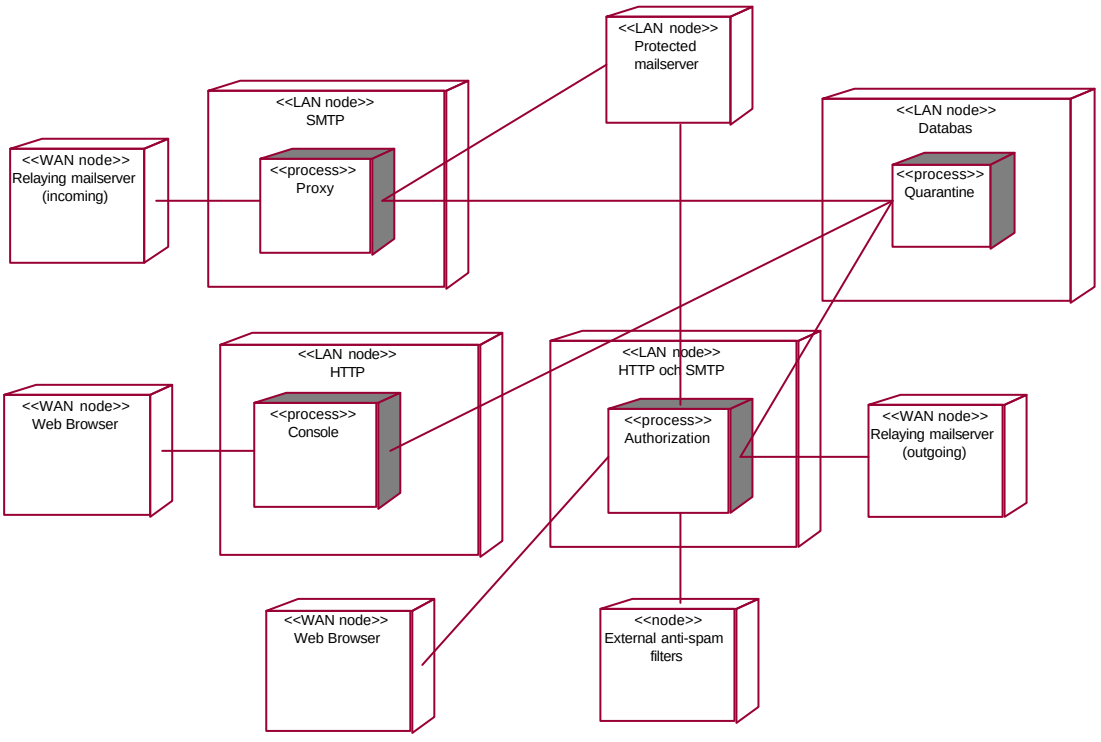
Inloggning mot systemet (admin console) sker enligt ”Basic Authentication” i http-standarden [19]. Eftersom algoritmen för kodning av lösenord är helt öppen måste kommunikationen krypteras. Detta kan göras med enkelriktad SSL och servercertifikat.

Eftersom det är viktigt att man kan lita på vad systemet gör distribueras systemet med öppen källkod. Det är också viktigt att systemet är fritt från minnesfel eftersom sådana fel kan utnyttjas för att göra dataintrång. För att minimera risken för minnesfel skrivs systemet i Java.

Systemet behöver minst två portar öppna mot Internet 25 (smtp) och 80 (http). Om andra filtreringssystem integreras i aktuell installation kan flera andra portar också behöva öppnas i företagets brandvägg.

6.5.5 Prestanda och skalbarhet

Kraven på prestanda är inte speciellt höga och systemet bör klara kraven med marginal trots att all funktionalitet i anti-spam systemet körs i samma process och på samma nod. Prestanda kraven bör också kunna uppfyllas om även den skyddade mailservern körs på samma nod. Om systemet skall byggas ut för att klara större volymer så kan man först och främst köra anti-spam systemet och mailservern på skilda noder. Anti-spam systemet kan sedan delas upp och köras på fyra olika noder (figur 6.23) vilket ger förutsättningar för att uppnå maximal prestanda.



Figur 6.23 Uppdelning av anti-spam systemet på fyra noder för maximal prestanda

6.5.6 Fail-over

Fail-over är inte speciellt viktigt för mottagning av brev eftersom SMTP protokollet säger att; brev skickas om från avsändaren om de inte kan levereras hos mottagaren. Det räcker det med en enkel omstartfunktion som startar om systemet om det skulle gå ner. Om det är viktigt att tillgängligheten för auktoriseringsförsöken, som sker via http är hög, kan man dela systemet på två lika http-noder (figur 6.23) och komplettera med hårdvara som sköter fail-over mellan noderna.

6.6 Technology view

Denna vy beskrivs inte i detta dokument eftersom systemet skall kunna köras oberoende av hårdvara.

7. Motivering av arkitekturförslaget

I detta avsnitt går vi igenom alla krav och motiverar hur väl kravet uppfylls av arkitekturen som beskrivs i föregående avsnitt.

7.1 Bedömning av kravuppfyllnaden med motivering

I tabell 7.1 nedan finns samma krav som redovisas i tabell 4.1 under rubriken *4.7 Sammanställning och prioritering av krav*. Varje krav har bedömts för hur väl kravet uppfylls enligt följande skala:

0 – Inte alls, 1 – Delvis, 2 – Mycket väl

Bedömningen av varje krav motiveras i kolumnen ”Kommentar” med en kort beskrivande text samt en referens till var detta beskrivs i arkitekturförlaget.

Id	*Benämning*	*Prio*	*Kommentar*	*Uppf.*
K1	*Systemet filtrerar ut oönskad post*	1	Fokus ligger på att sortera bort maskingenererad post. Maskingenererad post som inte är oönskad måste vitlistas. Se *6.3.3 Verifiera avsändare*	2
K2	*Systemet är oberoende av typ av: operativsystem, mailserver och mailklient*	1	Systemet skrivs i språket Java som är ett plattformsoberoende språk. Se *6.4.4 Val av programmeringsspråk*. Standardprotokoll används i kommunikationen mellan systemet och mail -server och -klient. Se *6.5.1 Processer och nätverksnoder*.	2
K3	*Systemet bygger ej på programvara som kräver att brukaren erlägger licensavgift*	1	Alla komponenter som valts i förslaget är produkter som distribueras enligt GPL eller LGPL. Se *6.4.5 Funktionalitet realiserad av externa komponenter*.	2
K4	*Systemet klarar av att ta emot 6000 brev/dygn fördelat på 200 konton och en genomsnittlig storlek av 100KB.*	1	Bedömningen är att detta krav skall klaras utan problem. Troligtvis klarar systemet betydligt större volymer. Se *6.5.5 Prestanda och skalbarhet*.	2
K5	*Systemet klarar av 20 samtidiga sessioner mot levererande servrar*	1	Avgörs av antalet trådar i trådpoolen för SMTP lyssnaren. Detta antal är fritt konfigurerbart. Se *6.5.2 Systemets trådar.*	2
K6	*Systemet är inte känsligt för DOS-attacker*	1	Arkitekturen tar inte höjd för detta på något speciellt sätt.	0
K7	*Det går att lita på systemets funktion*	1	Systemet kommer att distribueras med öppen källkod. Se *6.5.4 Säkerhet*.	2
K8	*Det finns möjlighet till sortering av mottagen post till olika inkorgar och direkt radering av misstänkt SPAM*	1	Systemet märker varje behandlat brev med beslutsinformation i både ”subject” och i SMTP-header. Se *6.4.6 Validate komponent*.	2

Id	*Benämning*	*Prio*	*Kommentar*	*Uppf.*
K9	*Systemet går att övervaka i drift*	1	Systemet har funktioner för loggning. Se *6.4.6 Webserver komponent* och *6.4.6 Statistics komponent*.	2
K10	*Systemet är utbyggbart med externa anti-spam tekniker*	2	Utbyggnad av systemet sker genom att implementera ett definierat gränssnitt. Se *6.4.6 Extensions komponent*.	2
K11	*Systemet är fritt från minnesfel*	2	Eftersom systemet skrivs i Java kan man anse att risken för minnesfel är liten, minnesfel i JVM kan trots allt finnas. Se *6.4.4 Val av programmeringsspråk*	1
K12	*Systemets arkitektur är tydligt beskriven*	2	Förhoppningsvis bidrar detta dokument till att systemet går att förstå.	2
K13	*Systemets design tydligt beskriven*	3	Ytterliggare detaljeringar behövs men det ligger utanför scopet av detta dokument. Måste beskrivas i design-dokument av designer/konstruktör.	1
K14	*Systemet är enkelt att installera*	3	Arkitekturen tar inte höjd för detta på något speciellt sätt.	0

Figur 7.1 Motivering av kravuppfyllnad

7.2 Samlad bedömning av kravuppfyllnaden

När man summerar bedömningen av hur pass väl arkitekturen uppfyller ställda krav kan man säga att den mycket väl uppfyller kraven, men att den brister på några ställen. Arkitekturen uppfyller inte kraven *K13-14* fullt ut vilket kanske är naturligt eller åtminstone inte allvarligt eftersom dessa krav är lågt prioriterade. Kravet *K11* är inte heller helt säkrat av arkitekturen eftersom man förlitar sig på att JVM är fritt från minnesfel, detta anses dock vara tillräckligt säkert. Kravet *K6* har inte tagits höjd för i arkitekturen. Problemet är ett känt problem när man exponerar system mot Internet och alla leverantörer av webservrar och mailservrar brottas med problematiken. Det borde alltså finnas många lösningar och tips att tillgå när man realiserar systemet eller när man arbetar vidare på arkitekturen. Bedömningen är att det i alla fall inte finns några direkta hinder i arkitekturen för att man bygger på systemet med DOS-skydd.

8. Slutsatser och rekommendationer

Att hitta en lösning för att eliminera skräppost har visat sig vara mycket svårt. En stor del av problemet ligger i att protokollen för postförmedling är öppna och från början framtagna för att fungera i en LAN liknande miljö. Protokollen byggdes upp för ARPANET (Advanced Research Projects Agency Network) som var ett litet isolerat nätverk. Om det skulle vara svårare att vara anonym när man skickar e-post via Internet skulle nog problemen med SPAM vara betydligt mindre. Nästa stora utmaning med att hitta en lösning är att alla personer har olika uppfattning om vad skräppost är.

Lösningen som föreslås i detta arbete är naturligtvis inte ”den perfekta” men den kan fungera som en ”basplatta” att bygga vidare specialanpassade lösningar på. Med basplatta avses att lösningen sorterar ut vad som skickats av maskiner vilket troligtvis är en bra startpunkt för en anti-spam lösning. Lösningen skall också ses som något av en tillfällig lösning eftersom en riktigt bra lösning inte är möjlig utan att protokollen görs om så att anonymitet blir omöjlig, detta kommer troligtvis inte att ske under de närmaste åren.

Lösningen har en uppenbar brist: om alla mailsystem i världen skulle installera system som bygger på denna lösning så bryter systemet samman eftersom all post skulle uppfattas som maskingenererad (Ett blockeringsmeddelande från ett system skickas som svar på ett blockeringsmeddelande från ett annat system). Ett annat problem med lösningen är ”förtroende frågan”. När du får ett blockeringsmeddelande med en uppmaning om att ange en kod, varför skall du lita på vad som sägs i meddelandet och inte bara kasta det och tänka ”skräppost”?

Slutsatsen är trots allt att lösningen har potential för att lösa problemet tills nya protokoll realiseras och rekommendationen är ta fram en referensimplementation så att man kan testa lösningens effektivitet. Vidare rekommenderas att man arbetar vidare med lösningens brister och då i första hand ”förtroende frågan”.

9. Diskussion

I detta avsnitt redovisar jag mina personliga erfarenheter av detta projektarbete.

Jag tycker att det har varit mycket intressant att arbeta med ämnet ”SPAM” eftersom jag har ett personligt intresse av att ”rädda” min privata e-post-adress. Detta har motiverat mig till att arbeta på med projektet och inte prioritera ned det trots hög belastning på arbetet. Det har varit extra lärorikt att arbeta med ett ämne som inte tillhör samma domän som jag arbetar med professionellt till vardags. Det blir t ex tydligt varför vissa saker behöver beskrivas när man arbetar utanför domänen där man kanske är lite ”hemmablind”. Jag valde att beskriva arkitekturförslaget (SAD:en) enligt ”RM-ODP” i stället för ”Kruchtens 4+1 modell” (vilken jag arbetar med professionellt) och det visade sig vara nyttigt och har lett till att jag kommer att arbeta för en del förändringar i hur vi dokumenterar arkitekturen på mitt företag.

Arbetssättet som jag valde, inventera och undersöka befintliga tekniker samt att genomföra en enkätundersökning hos e-post-användare, har fungerat väl. Enkät undersökningen borde dock ha gjorts i större skala och jag borde ha tänkt noggrannare på hur frågorna skulle sammanställas innan genomförandet. Det visade sig nämligen vara besvärligt och tidsödande att sammanställa vissa frågor. Till sist vill jag även nämna att jag väldigt tidigt ”snöade” in på en specifik lösning med captcha-teknik. Att tidigt låsa sig till en lösning kan vara farligt och kan starkt kan bidra till att man blir blind och därför inte arbetar för den bästa tänkbara lösningen.

10. Dokumentinformation

I detta avsnitt hittar du information som rör själva dokumentet. Här finns förteckningar över använd litteratur och referenser samt en ordlista som beskriver ord och förkortningar som används i dokumentet. Sist finns en tabell som redovisar revisionshistoriken för dokumentet.

10.1 Litteraturförteckning

Här redovisas den viktigaste litteraturen som har använts av författaren vid författandet av detta projektarbete.

Dokumenttitel	Författare/utgivare	Adress/ISBN
1. A practical guide to enterprise architecture	James McGovern mfl.	ISBN 0-13-141275-2
2. Software Architecture in Practice	Len Bass mfl.	ISBN 0-321-15495-9
3. Pattern-oriented software architecture	Frank Buschmann mfl.	ISBN 0-471-95869-7
4. CEAS 2006 papers	The Third Conference on Email and Anti-Spam (CEAS 2006)	http://www.ceas.cc/2006/cfp.html

Tabell 10.1 Litteraturförteckning

10.2 Referenser

Här beskrivs alla referenser som används i dokumentet.

Dokumenttitel	Utgivare/författare	Adress/Dnr
1. RFC 2821 – Simple Mail Transfer Protocol	Network Working Group	http://www.ietf.org/rfc/rfc2821.txt
2. RFC 1939 – Post Office Protocol – Version 3	Network Working Group	http://www.ietf.org/rfc/rfc1939.txt
3. RFC 2060 - INTERNET MESSAGE ACCESS PROTOCOL - VERSION 4rev1	Network Working Group	http://www.ietf.org/rfc/rfc2060.txt
4. RFC 2616 - Hypertext Transfer Protocol -- HTTP/1.1	Network Working Group	http://www.ietf.org/rfc/rfc2616.txt
5. RFC 4408 - Sender Policy Framework (SPF) for Authorizing Use of Domains in E-Mail, Version 1	Network Working Group	http://tools.ietf.org/html/rfc4408
6. Sender Reputation in a Large Webmail Service	Bradley Taylor, Google Inc. 2006	http://www.ceas.cc/2006/19.pdf
7. RFC 4406 - Sender ID: Authenticating E-Mail	Network Working Group	http://www.ietf.org/rfc/rfc4406.txt
8. RFC 4405 - SMTP Service Extension for Indicating the Responsible Submitter of an E-Mail Message	Network Working Group	http://www.ietf.org/rfc/rfc4405.txt
9. RFC 4407 - Purported Responsible Address in E-Mail Messages	Network Working Group	http://www.ietf.org/rfc/rfc4407.txt
10. JCaptcha	JCaptcha	http://jcaptcha.sourceforge.net/
11. Derby	The Apache DB project	http://db.apache.org/derby/
12. OpenJPA	Apache OpenJPA	http://openjpa.apache.org/
13. Pygmy	Pygmy tiny webserver	http://pygmy-httpd.sourceforge.net/
14. DomainKeys Identified Mail (DKIM) Signatures	Network Working Group	http://www.ietf.org/rfc/rfc4871.txt
15. IEEE 1471 Recommended Practise for Architectural Description of Software-Intensive Systems	IEEE Computer Society 2000	http://standards.ieee.org/reading/ieee/std_public/description/se/1471-2000_desc.html
16. The 4+1 View Model of Architecture	Kruchten Philippe IEEE Software 1995	http://doi.ieeecomputersociety.org/10.1109/52.469759
17. ISO/IEC 10746 Open distributed processing	Information technology 1996	http://standards.iso.org/ittf/PubliclyAvailableStandards/
18. Extended Log File Format, W3C Working Draft WD-logfile-960323	W3C	http://www.w3.org/TR/WD-logfile
19. RFC 2617 - HTTP Authentication: Basic and Digest Access Authentication	Network Working Group	http://tools.ietf.org/html/rfc2617
20. Mini application server	tonjac.org	http://www.tonjac.org/mini/

Tabell 10.2 Referenser

10.3 Begrepp och förkortningar

Här ges kortfattade förklaringar till ord och förkortningar som används i dokumentet.

Ord/förkortning	Beskrivning
ARPA	**Defense Advanced Research Projects Agency** (**DARPA**) är en avdelning i USA:s försvarsdepartement som utvecklar teknologi för militära ändamål. Dess ursprungliga namn var **Advanced Research Projects Agency (ARPA)**, men namnet byttes till DARPA (D för *defense*) 23 mars 1972
ARPANET	(Advanced Research Projects Agency Network) var den tekniska föregångaren till Internet. Det var utvecklat av ARPA.
ASP	**ASP**, *Active Server Pages*, är en teknik utvecklad av Microsoft för att kunna baka in programkod (ASP-kod) i HTML-koden i webbsidor. ASP används för att skapa dynamiska webbsidor och webbapplikationer, exempelvis genom att läsa och skriva information till databaser.
Autentisering	Kontroll av uppgiven identitet, till exempel vid inloggning, vid kommunikation mellan två system eller vid utväxling av meddelanden mellan användare
Captcha	Ordet **captcha** är en alternativ, ljudenlig stavning av det engelska ordet *capture* (fångst). Det är alltså ett slags fälla. Ibland förklaras ordet captcha som en förkortning för "**C**ompletely **A**utomated **P**ublic **T**uring-test to tell **C**omputers and **H**umans **A**part", "Helt automatiskt Turingtest för att skilja på datorer och människor", men det är en efterhandskonstruktion (s.k. "backronym").
DNS	**DNS**, *Domain Name System* eller **domännamnssystemet**, är ett system för att förenkla adressering av datorer på IP-nätverk som till exempel Internet. Det uppfanns år 1983 av Paul Mockapetris. DNS är ett distribuerat system, där så kallade zoner av IP-nummer delegeras till ansvariga företag och organisationer. DNS är ett klient–server-system där klienten vanligen finns inbyggd i system som kommunicerar genom IP-nätverk.
DOS	**DoS** står för *Denial of Service* vilket ungefär betyder *blockering av tjänst*. Termen används inom området datasäkerhet och avser en attack med en dator, ett nätverk el dyl med syfte att slå ut offret, oftast genom överbelastning.
GPL	**GNU General Public License**, vanligtvis förkortat till **GNU GPL** eller **GPL**, är en upphovsrättslicens för fri programvara som ursprungligen skrevs av Richard Stallman.
LGPL	**GNU Lesser General Public License**, *GNU LGPL*, (tidigare GNU Library Public License) är en licens för fri programvara. Den främsta skillnaden mot den mer kända GNU General Public License är att det är tillåtet att inkludera program licenserade under LGPL i ett nytt program, utan att det nya programmet omfattas av LGPL. Detta gör att man till exempel i en kommersiell programvara med sluten källkod, kan dra nytta av ett externt bibliotek som är tillgängligt under LGPL, utan att bryta mot licensregler.
HTTP	**Hypertext Transfer Protocol** är det kommunikationsprotokoll som används för att överföra webbsidor på informationsnätverket WWW, World Wide Web på Internet

Ord/förkortning	Beskrivning
IMAP	**IMAP** - *Internet Message Access Protocol* Med IMAP "tankar" man inte hem den e-post man avser läsa, till skillnad från POP3. Den finns lagrad på en central server. Tanken med IMAP är att man på så vis får en effektivare hantering av e-post, då man bla kan hantera flera e-postkonton/användare på ett mer effektivit sätt. Övriga funktioner är bland annat centrala publika brevlådor som flera har åtkomst till.
Internet mail	**E-post** eller **e-brev** (för meddelandet)(**elektronisk post**, även kallat **e-mail**, **mail** eller **mejl**) är en av de ursprungliga typerna av meddelandebefordran över Internet
ISP	**Internetleverantör** eller **ISP** (*Internet Service Provider*) är en aktör som tillhandahåller en uppkoppling till Internet. Kunden - ett privathushåll, en förening eller ett företag - kontaktar via modem, DSL, ISDN eller olika typer av bredband en av leverantörens accesspunkter så att en förbindelse med internet kan upprättas. I de flesta fall är internetleverantörer företag, men ibland kan det vara en förening.
Java	**Java**, eller **JAVA**, är ett objektorienterat programspråk som konstruerades av bland andra James Gosling på Sun Microsystems 1991-1995. Ursprungligen kallades språket *Oak*, men Sun bytte namn innan Java presenterades för världen 23 maj 1995. Anledningen till namnbytet var att 'Oak' var upptaget.
JPA	**Java Persistence API**, Java standard för att mappa javaobjekt till tabeller I en relationsdatabas.
JSP	**JSP** eller **JavaServer Pages** är en Java-teknologi som används för att dynamiskt skapa svar från en webbserver. Oftast skapas HTML, XHTML eller andra typer av webbsidor.
JVM	**Java Virtual Machine**, *JVM*, är en programvara utvecklad Sun Microsystems. JVM är det program som "kör" program skrivna i Java. JVM är, som namnet antyder, en virtuell maskin. Den är alltså inte en maskin (dator) som sådan, utan är en simulerad dator som körs i systemet. JVM:en porteras till många OS och på så sätt kan Java-program köras på många olika plattformar.
LAN	**Local Area Network** förkortat **LAN** är ett lokalt nätverk och ett begrepp inom datorkommunikation. Med LAN avses ett nätverk begränsat till en byggnad, eller möjligen en grupp av byggnader.
NDR	**Non-Delivery Report**, meddelande som ”stutsar” tillbaka till en avsändare av ett mejl som inte kunde levereras till mottagaren. Orsaken kan vara att avsändaren inte finns eller att avsändarens SMTP-server är nere.
phishing	**Nätfiske** eller **phishing** (efter engelskans *fishing*, ’fiske’) är en olaglig metod att lura innehavare till bankkonton och andra elektroniska resurser att delge kreditkortsnummer, lösenord eller annan känslig information. Ett vanligt mål är Ebays kunder. Nätfiske är oftast utformat som ett e-brev som ser ut att komma från en bank eller ett kreditkortsbolag, och som innehåller en uppmaning att logga in snarast möjligt och en länk till en falsk webbsida med inloggningsformulär.

Ord/förkortning	Beskrivning
PKI	**Public Key infrastructure**, PKI. Den dominerande standarden för att hantera publika krypteringsnycklar. Möjliggör för användare av ett i grunden osäkert publikt nätverk som till exempel internet, att säkert utbyta data genom att använda ett kryptonyckelpar, som består en publik och en privat nyckel. Dessa tillhandahålls av något godkänt företag, som till exempel VeriSign.
POP3	**POP3** är version **3** av **Post Office Protocol**. POP3 är det vanligaste kommunikationsprotokollet för att hämta e-postmeddelanden från en server till ett e-postprogram, även om det mer och mer håller på att konkurreras ut av IMAP som är ett mer kapabelt protokoll.
RUP	**Rational Unified Process** (RUP) är en systemutvecklingsmodell för design och implementering av IT-system. RUP grundar sig på utvecklade och testade s k "bästa tillämpningar" i en iterativ utvecklingscykel. Tanken är att RUP ska skräddarsys så att det passar den enskilda organisationen och projektet. Utvecklingsmodellen ägs numera av IBM och utvecklas av IBM Rational Software.
SMTP	**Simple Mail Transfer Protocol** (**SMTP**) är det vanligaste kommunikationsprotokollet för att leverera elektronisk post
SPAM	**Skräppost** eller **spam** är oönskade reklamutskick, bedrägeriförsök eller utskick av datorvirus via elektroniska medier, vanligast förekommande i e-post. På engelska förekommer även synonymerna UCE (*unsolicited commerical e-mail*, "oönskad kommersiell e-post") och UBE (*unsolicited bulk e-mail*, "oönskade massutskick av e-post"). Svenska datatermgruppen rekommenderar *skräppost*, men accepterar även *spam* som lånord.
Spyware	**Spionprogram** (engelska: *Spyware*) är program som följer med lite "skummare" program. Dessa program verkar ofta bra genom att de innehåller någon nyttig funktion. Dessutom är de ofta gratis s.k. freeware. Spionprogram kan till exempel ta användares lösenord och skicka det till upphovsmakaren (eller någon annan) eller ge riktad reklam (kollar vilka sidor användaren surfar på o.s.v.).
Trojan	Inom datorvärlden betecknar en **trojansk häst** ett datorprogram som utger sig för att vara till nytta, men som orsakar skada när det lurat en användare att installera det. Vanliga trojanska hästar installerar någon form av bakdörr eller spionprogram i datorn eller förstör och ändrar andra filer.
Virus	Ett **datorvirus** är i dagligt tal ett datorprogram som sprids genom att lura oförsiktiga användare att starta det. Vanligen sprids virus genom en bifogad fil till e-brev med någon rubrik som gör den intet ont anande användaren mycket nyfiken. En besläktad företeelse är "maskar", som inte behöver någon hjälp från användaren för att spridas. Datorvirus motverkas med antivirusprogram.
WAN	**Wide Area Network** förkortat **WAN** är en inom datorkommunikationen ett datornätverk som är så stora att de omfattar exempelvis ett land eller globala landmassor. Det huvudsakliga exemplet är Internet, men även det svenska universitetsdatanätet Sunet är ett WAN.

källor Wikipedia och susning.nu

10.4 Revisionshistorik

Här redovisas dokumentets revisionshistorik.

Datum	Version	Beskrivning	Signatur
2007-04-26	0.1	Första utkast med bakgrund, problembeskrivning, avgränsning, målgrupp och metod.	Toni Thomsson
2007-04-30	0.2	Beskrivning av filtreringstekniker. Förändring enligt kommentarer från KS[1] på 0.1	Toni Thomsson
2007-09-10	0.3	Sammanställning av enkätsvar, början till rubriker och struktur på arkitekturförslaget.	Toni Thomsson
2007-09-19	0.4	Uppdatering av avsnittet ”Intressenterna…”. Förändring enligt kommentarer från KS på 0.3	Toni Thomsson
2007-10-01	0.5	Nytt kapitel för analys och lösningsförslag. Beskrivning av perspektiv och vyer.	Toni Thomsson
2007-10-28	0.6	Flyttat ”Systemets trådar” till Egineering vyn. Val av programmeringsspråk. Beskrivning av ”Kritiska användningsfall”. Beskrivning av vyerna. Beskrivning av ”Processer och nätverksnoder”. Utveckling av ”Computational vyn”. Rättning av diverse små felaktigheter i text och modeller. Förändring av definition av skräppost. Abstract	Toni Thomsson
2007-11-02	0.7	”Sammanfattning” , ”Motivering av arkitekturen”, ”Beskrivning av Transaktioner”, ”Säkerhet”, ”Prestanda och skalbarhet” samt ”Fail-over” i Engineering vyn. Kategorisering av krav i kravlistan.	Toni Thomsson
2007-11-05	0.8	”Slutsatser och rekommendationer” och ”Diskussion”	Toni Thomsson
2007-11-19	0.9	Release kandidat 1. Version till opponenter[2] och handledare. Förändring enligt kommentarer från KS på 0.7.	Toni Thomsson
2007-11-28	1.0	Förändringar enligt kommentarer från handledare, opponenter och MÖ[3]. Slutlig version.	Toni Thomsson

Tabell 10.3 Revisionshistorik

[1] KS – Karin Stenberg (Handledare)

[2] Opponenter – Jan Broman (Astrazeneca), Mats Kempe (Modul-1)

[3] MÖ – Maria Öberg (Skatteverket)

11. Bilagor

11.1 Enkät till E-post användare

Nedan finns enkäten som användes för att undersöka vilka problem E-post användare upplever med sin E-post.

Svaren finns sammanställda under rubriken *11.2 Sammanställning av enkät*

KORT UNDERSÖKNING AV PROBLEM MED SKRÄPPOST (SPAM) PÅ INTERNET

1. Svar på denna enkät skall göras ur ett av två perspektiv, välj vilket (du får gärna fylla i två blanketter om du vill svara ur båda perspektiven)
   - _ Min privata e-post
   - _ Min arbetsrelaterade e-post

2. Kön?
   - _ Kvinna
   - _ Man

3. Hur har du fått din e-post-adress?
   - _ Har egen domän
   - _ Gratis e-post-tjänst (hotmail, Gmail m fl)
   - _ ISP (Telia, Bredbandsbolaget, m fl)
   - _ Arbetsgivaren

4. Finns din e-post-adress publicerad på Internet? (t ex på egen hemsida, CV e dyl)
   - _ Ja
   - _ Nej
   - _ Vet ej

5. Har du använt din e-post-adress när du registrerat ett konto av något slag på Internet?
   - _ Ja
   - _ Nej

6. Hur läser du oftast din e-post?
   - _ Vanlig webläsare (Internet Explorer, Firefox, e dyl.)
   - _ Vanlig e-post-klient (Outlook, Thunderbird, Lotus Notes e dyl.)
   - _ Mobiltelefon

7. Har du ett ”SPAM-filter”?
   - _ Ja
   - _ Nej
   - _ Vet ej

8. Vilken typ av SPAM-filter du har? (Du kan kryssa flera)

_ Bayesian
_ Blacklist (SpamCop, ORDB e dyl)
_ SPF (Sender policy framework)
_ Sender-ID
_ Voting
_ Manuellt (Klickar på ”skräp” när du fått något som du uppfattar som skräp)
_ Vet ej

9. Får du ofta skräppost?
   _ Ja, varje dag
   _ Ja, någon gång i veckan
   _ Nej, endast någon gång i månaden
   _ Nej, aldrig
   _ Vet ej

10. Vad tycker du är det största problemet som orsakas av SPAM eller av ditt skydd mot SPAM?
   _ Skräppost hamnar i inkorgen
   _ Ren post hamnar i ”skräpkorgen”
   _ Post ”försvinner”
   _ Datorn smittas med virus eller annan skadlig programvara
   _ Upplever inga problem

VÄND!

11. Hur bedömer du konsekvensen av att nedanstående brev inte hamnar i din inkorg:
    0 – Allvarligt, 1 – Inte bra, 2 – Bra, 3 - Viktigt

   _ Brev innehållande virus, spyware eller trojan
   _ Phishingförsök (T ex falsk uppmaning om att ange bankkoder)
   _ Kedjebrev, Afrikabrev e dyl.
   _ Reklamutskick från företag från vilket du aldrig handlar
   _ Reklamutskick från företag där du är eller har varit kund
   _ Utskick från nyhetsgrupp eller tjänst där jag har registrerat min e-post-adress
   _ Utskick från nyhetsgrupp eller tjänst där jag inte registrerat min e-post-adress
   _ Brev innehållande hot eller mobbing
   _ Brev från vän eller släkting

12. Hur bedömer du konsekvensen av att nedanstående brev hamnar i din inkorg:
    0 – Allvarligt, 1 – Inte bra, 2 – Bra, 3 - Viktigt

   _ Brev innehållande virus, spyware eller trojan
   _ Phishingförsök (T ex falsk uppmaning om att ange bankkoder)
   _ Kedjebrev, Afrikabrev e dyl.
   _ Reklamutskick från företag från vilket du aldrig handlar

_ Reklamutskick från företag där du är elle r har varit kund

_ Utskick från nyhetsgrupp eller tjänst där jag har registrerat min e-post-adress

_ Utskick från nyhetsgrupp eller tjänst där jag inte registrerat min e-post-adress

_ Brev innehållande hot eller mobbing

_ Brev från vän eller släkting

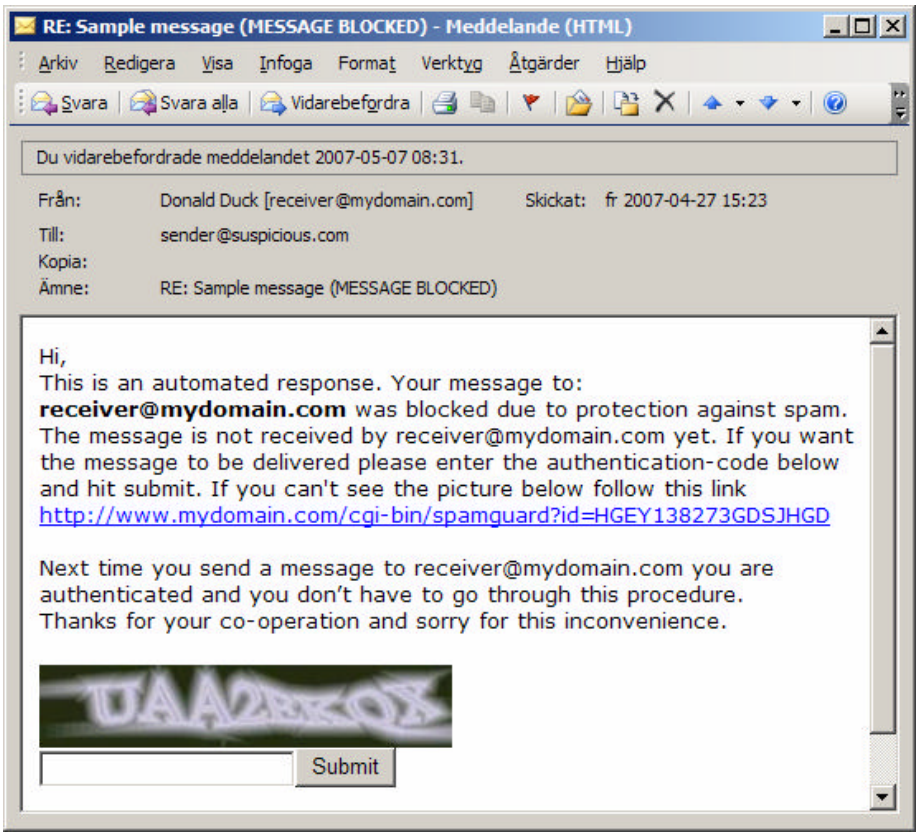
13. Skulle du kunna tänka dig att svara på nedanstående uppmaning första gången du, sender@suspicious.com, skickar ett brev ”Sample message” till en person, receiver@mydomain.com?

_ Ja, men jag förstår inte varför

_ Ja, eftersom denna uppmaning även skulle kunna hjälpa mig

_ Nej, helst inte

_ Nej, aldrig, skulle betrakta brevet som SPAM



11.2 Sammanställning av enkät

Enkäten skickades till ca 30 personer varav 13 personer svarade. Enkäten kunde svaras på ur två perspektiv: ”Privat e-post” och ”E-post på arbetet”. De flesta svarade ur det privata perspektivet.

	Fråga	Sammanställning														
Total	1	Privat/arbete	9	4												
Privat	2	Man/kvinna	12	1												
Arbete	3	Kontotyp	2	7	3	0				0	0	0	4			
	4	Publicerad adress	3	5	1					2	2	0				
	5	Registrerad adress	9	0	0					2	2	0				
	6	Klienttyp	4	5	0					0	4	0				
	7	Filter	7	1	1					4	0	0				
	8	Filtertyp	1	1	0	1	0	4	5	0	0	0	0	0	0	4
	9	Skräppostfrekvens	2	6	1	0	0			0	1	1	2	0		
	10	Största problem	11	2	1	2	1									
	11	Ej inkorg	31	27	35	25	15	8	22	17	3					
	12	Till inkorg	6	9	10	11	22	29	12	19	33					
	13	Blockskydd	0	6	3	4										

Tabell 11.1. Sammanställning av enkätsvar

1) De flesta är restriktiv med att lämna ut sin arbetsmejladress på internet. Den privata mejladressen lämnas däremot ut ganska flitigt!
2) De flesta har mindre problem med skräppost på arbetet.
3) De flesta har ett SPAM-filter som skydd. Trots detta får de flesta ganska ofta skräppost (i alla fall privat).
4) De flesta använder en vanlig mail-klient när man läser sin mejl, men vanligt är även webläsare.
5) De flesta tycker att det största problemet med skräppost är att den når inkorgen. Största problemet är alltså inte hotet: virus, fiske osv.
6) De flesta anser att det är större problem med att skräppost når inkorgen än att "ren" post fastnar i ett filter.
7) De flesta anser att reklam innebär mest problem följt av virus, nätfiske och mobbing.
8) De flesta är positiva till en blockeringsfunktion som skydd för sin inkorg.

Tabell 11.2. Slutsatser (påståenden) av enkätundersökningen