A Methodology and Architecture for a Dependable and Secure E-mail Service

Marcelo Maraboli
Chief Network & Systems Engineer
MSc Student in Electronic Engineering
Universidad Técnica Federico Santa María, Valparaíso, Chile
+56-32-2654071, marcelo.maraboli@usm.cl

Reinaldo Vallejos
Ph.D, Professor of Electronic Engineering Department
Universidad Técnica Federico Santa María, Valparaíso, Chile
+56-32-2654207, reinaldo.vallejos@usm.cl

***Abstract*—A novel methodology and architecture of the Internet e-mail service (SMTP standard) based on the concepts of dependability and secure computing is presented. The goal of this work is four folded: firstly, to raise awareness about all the existing faults and threats of today's e-mail service. Secondly, to provide an exhaustive and systematic classification of faults, threats, failures that can affect E-mail services and their corresponding defense mechanisms. Thirdly, to provide a methodology that will enable IT staff to redesign and deploy a secure and dependable e-mail infrastructure in a proactive and preventive process. Finally, we propose a new dependable and secure E-mail architecture with protections against every classified fault and threat using a corresponding defense mechanism.**

***Index Terms*— Dependability, Electronic mail, Internet.**

I. Introduction

E-MAIL is today the most popular Internet application for asynchronic communications. According to [34], in the first quarter of 2006 there are about 1.1 billion email users worldwide, about one in every six persons on the earth use email. The estimate the number of emails sent per day is around 171 billion meaning almost 2 million emails are sent every second. E-mail is widely used for private use as well as for business's internal and customer/supplier relationships. Because of this, users demand a high availability of e-mail systems. This leads IT managers to permanently try to increase their e-mail infrastructure's security and dependability, defined as the ability of a system to avoid service failures that are more frequent and more severe than is acceptable [1].

If email is such an important application, then every e-mail architecture should have appropriate defense mechanisms against every possible fault and threat. Most people might think that securing an E-mail infrastructure is a common practice, however, it is common to see that IT staff modify their infrastructure on a reaction basis; by applying solutions to problems as they occur. Typically, they apply protections in advance only for the most common problems, like an anti-virus or an anti-SPAM appliance, which do not *solve* the problem by themselves, but fail to protect the system from far more catastrophic threats like a slammer attack (denial of service) [11] or a phishing attack [31] that will significantly affect the company's credibility on customers.

Additionally, there is little detailed knowledge on how SMTP (Simple Mail Transfer Protocol) [2] works, specifically its design flaws and present threats are not widely known. Few IT e-mail administrators have a complete understanding of these faults, and therefore cannot protect their e-mail systems proactively in a timely manner. This superficial knowledge is, in part, due to the lack of a complete taxonomy of faults, threats and failures of an Internet e-mail system based on the SMTP protocol. Thus, it is of fundamental importance to raise awareness about all the faults and threats existing nowadays. Furthermore, there is no common knowledge of which system functionality is affected by which fault or threat, and subsequently what defense mechanism to deploy. Finally, there is no detailed updated taxonomy of available defense mechanisms, mitigations or solutions to each of these faults and threats. Protecting an e-mail system is not expensive, nor extremely difficult, but it is not possible to protect a system from failures that IT staff are unaware of.

Since SMTP lacks many features and has some known security issues, many other protocols have been standardized to complement SMTP [3]. There are proposals to replace SMTP with new architectures, such as "sender storage" systems like IM2000 [4], Peer-to-Peer Email [5] and Multimedia e-mail [6]. However, both proposals are unlikely to be deployed since evolution seems to be the consensus. On the other hand, there have been various proposals regarding the dependability of E-mail systems, such as in [7]; the dependability of free public e-mail providers like Hotmail and Yahoo is studied from a user's point of view (delays and silent discards of electronic mails). In [8], the Berkeley/Stanford Recovery-Oriented Computing (ROC) Project proposes novel techniques for building highly-dependable Internet services and designs new methods to test its dependability. Dependability software tools for E-mail systems are not common, most of these tools are designed for Web Services. There are few major email benchmarks in production use today (i.e. Netscape's Mailstone [32] and SPEC's SPECmail2001 [33]). They focus mainly on performance and only measure dependability as a simple count of dropped client connections, which is in fact a conceptual mistake with the definition of Availability [1].

Until now, most e-mail infrastructures have been designed by focusing mainly on performance [9] (e.g. email transfer

Manuscript received October 15, 2006. Financial support from Fondecyt Projects 1000055 (Chilean Government), USM Projects 23.04.27 (Universidad Santa María), and Fundación Andes Chile, is gratefully acknowledged

rate, storage capacity), scalability, hardware fault tolerance through redundancy [10] and some security issues [11] [12] (password and content protection, cyber attacks). All these improvements are aiming in the right direction, but are insufficient to respond to all the attributes required by a dependability and security analysis [1], which include concepts like availability, reliability, safety, confidentiality, integrity and maintainability, accountability, authenticity and non-repudiability. In fact, despite the many protocols, standards and proposals to complement or replace SMTP as a de-facto standard, the problem of dependability and security in e-mail systems persists until today. To improve the dependability of the e-mail system, defense mechanisms should be deployed before problems arise. Therefore, existing systems should be redesigned or modified with dependability and secure computing attributes as their primary focus.

Our goal is not to *change or improve* how SMTP works or to propose a *replacement* of SMTP, because these kinds of proposals [4], [5], [6] have not been implemented due to economical and practical deployment problems at an Internet wide basis. We therefore focus at the E-Mail SMTP Service (Application Level), specifically on how to design a Dependable and Secure SMTP service in today's environment. (Hence, hardware and operating systems faults and fault tolerance solutions (Physical and OS Level) are out of the scope of this paper and are a complement to this work.)

This paper has a theoretical and practical contribution. The theoretical contribution consists in a complete dependability and secure computing classification of the Internet e-mail standard SMTP based on the Avizienis Taxonomy [1]. This classification consists in an exhaustive and systematic categorization and matrix representation of the Internet SMTP e-mail faults, threats and failures. To do so, SMTP faults and threats are explained and classified along with describing the state of the art defense mechanisms. The classification is flexible enough to allow for future -yet unknown- threats and their corresponding defense mechanisms to be incorporated. It is expected that its systematic use will enable IT staff to redesign, modify and plan future enhancements to their e-mail infrastructure in a proactive and preventive process.

On the other hand, the practical contribution of this work consists in proposing a secure and dependable E-mail architecture based on the classification, which will allow IT staff to protect their e-mail infrastructure.

The rest of this paper is structured in a top-down approach. We first explain the methodology in Section 2 and a brief description of the e-mail standard (SMTP) in Section 3. Later we present the classification of faults, failures and defense mechanisms in Sections 4, 5, 6 and 7. Finally, in Section 8, we present the proposed E-mail architecture.

II. Methodology

The methodology composed of 2 parts. The first part is an application of the Avizienis Taxonomy [1] to the E-mail service, which provides the basis for the classification of faults, threats and failures.

The second part is a 4 step periodic procedure:

1.- **Review** the e-mail infrastructure by making a detailed itemized list of its capabilities, enabled features and defense mechanisms (Section 7, figure 5).

2.- **Fault Detection**. A set of tests that should be conducted to detect if the faults and weaknesses described in Section 4 (see figure 2) are properly protected. These tests will show which faults have a working defense mechanism, a non-working defense mechanism or no defense mechanism at all (see figure 5).

3.- **Failure Evaluation**. The detected faults may produce a system failure, as classified in Section 5 (see Figure 3), so step 3 is an evaluation of the damage of possible failures, either to the e-mail system or to other systems (Section 6, figure 4). User's perception of a failure is also inferred by Section 6, since they will perceive a problem of the system in relation to a dependability or security attribute (see figure 4).

4.- **Defense Mechanism Deployment**. The assessment of failures will lead to a deployment plan of multiple defense mechanisms in the E-mail architecture (Section 8) for fault removal or mitigation (Section 7, figure 5).

III. The Simple Mail Transfer Protocol - SMTP

The most important protocol of the Internet E-Mail system is without a doubt the Simple Mail Transfer Protocol (SMTP) [2]. Its primary objective is to transfer mail reliably and efficiently across the Internet. SMTP has evolved from RFC 821 to the latest specification in RFC 2821 [2] and is complemented by many other RFC standards like POP3, IMAP4, ESMTP, LMTP and MIME [3].

The SMTP basic structure consists of 3 components: 1) a SMTP client called Mail User Agent – MUA, from which a user may send and receive email, 2) several SMTP servers called Mail Transfer Agents – MTA, in which emails are transferred from one MTA to another and 3) a Message Store – MS server, where email reaches its final destination and is deposited in a user's mailbox. For successful operation, at least 2 MTA servers are needed: one for the sending domain and one for the receiving domain. Intermediate MTAs called "SMTP relays", may be added for filtering emails, antivirus checking or translation purposes.

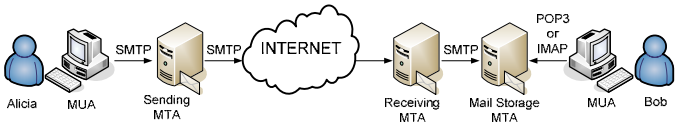


Figure 1- Basic structure of an SMTP system

The process is shown in Figure 1: Alice's MUA sends email to its configured SMTP server. This server looks up the "Mail Exchanger" MX record of the recipient's domain, which is in fact an MTA server. This latter MTA server will receive the email and optionally send it to another internal MTA server, and so on until it reaches the MS server. The receiving user (Bob) checks for new email with a POP, IMAP or Webmail MUA

IV. Faults

The relationship between faults, failures and defense

mechanisms is quite simple. A “fault” is the cause of an “error”. This fault must be activated (internally or externally) to cause the error. A threat is an unwanted (deliberate or accidental) event that may result in harm to an asset. Often, a threat is exploiting a known vulnerability(ies), in other words, a “threat” is generally the activation (exploitation) of a fault (vulnerability).

The generated error may lead to a system “failure”, which is an event that occurs when the delivered service of a system deviates from a correct service state, in other words, the system delivers an incorrect service. On the other hand, a defense mechanism is a mean to increase the dependability and/or security of a system by tolerating (continue to provide correct service), removing (eliminating) or mitigating (decrease the effect) the activation of faults.

Most of the Internet E-mail service faults are well known and many working defense mechanisms are available today, but these faults are not widely understood nor defense mechanisms deployed for each case. More importantly, most IT staff members are not aware if there are more faults.

This taxomomy of Faults reviews all existing SMTP faults, either attributed to a design flaw or to an interaction abuse or threat. These faults are revised and classified according to the Avizienis Taxonomy [1] and are classified into two categories: Design flaws and Interaction faults.

Design flaws are faults related specifically to the SMTP standard [2] without considering the numerous optional or additional protocols and standards [3]. Interaction faults are meant to address the common threats and attacks to the e-mail system as well as faults derived from any type of user interaction such as input mistakes, intrusion attempts and any form of malicious activity. Most of them could not have been foreseen at the initial design stage of SMTP in 1981. Due to space restrictions, we only list each of the 14 faults of the 2 categories. Further details can be found in [37].

The Design Flaws are:

- SMTP is an “any MUA to any MTA” system
- Spoofed/forged Email
- Open Relay
- Anonymous messages
- Data size is set by sender
- A sender may amplify its traffic
- No negative feedback
- No content privacy

The Interaction Faults are:

- SPAM or Unsolicited Bulk Email UBE [13][14][15][16]
- Virus (via email)
- Hoax – Fraud
- Phishing Attacks [31]
- Slammer Attack [11]
- Mail bombing [11]

Next we show a classification of faults that can affect E-mail SMTP protocol. This classification is an application of [1] to the E-mail service as shown in Figure 2. This classification shows a clear distinction between the Design Flaws and Interaction Faults, since the former are associated to the Development stage of the SMTP protocol and the later to Operational faults [1]. This classification focuses on the Software related faults, since any Hardware Faults are a non SMTP matter.

Future faults may easily be incorporated into this matrix and grouped with similar faults. This may give system administrators an insight on how to defend the E-mail service.

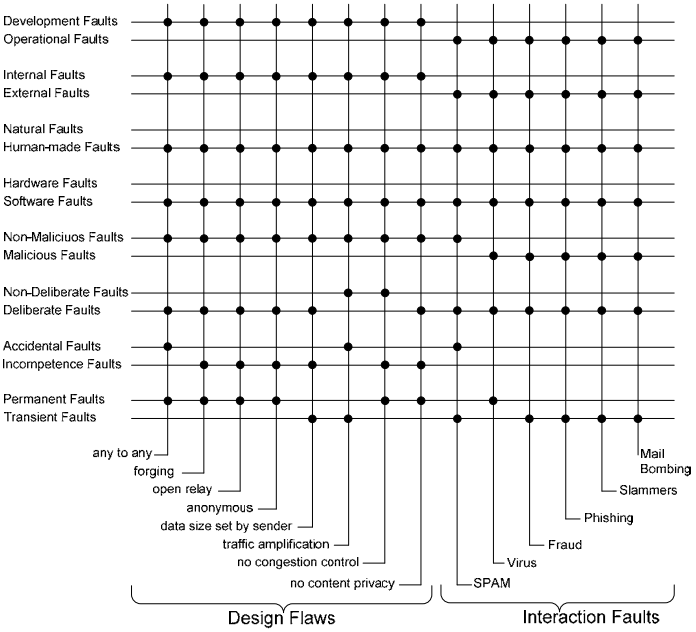


Figure 2- SMTP Fault classification matrix.

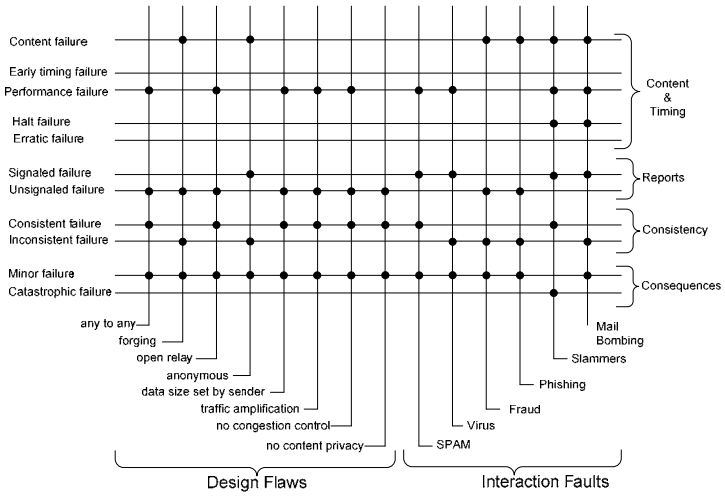


Figure 3- SMTP Failure classification matrix.

V. FAILURES

If activated, each of the 14 faults will produce an error. This error may lead to a service failure (deviation from the correct service). Figure 3 shows a failure matrix classification of each of the described SMTP faults.

An important remark is that there is only 1 catastrophic failure due to a Slammer Attack, since this will produce a halt failure for all system users of an E-Mail system. The Mail

Bombing attack will normally produce a halt failure for a specific user and may cause a halt failure for all system users, if there is no individual user quotas are enabled, a rare situation.

Security failures are mostly due to secondary attribute failures, i.e. accountability, authenticity and non-repudiation. Another interesting aspect is that most of the failures are not reported (unsignaled), meaning that special detection tools (scripts, log analyzers, etc.) are to be installed to detect these failures.

VI. Dependability Attributes Affected by Faults and Threats

The relationship between Faults and dependability/security attributes [1] is not easily inferred. To clarify this issue, this relationship is shown in Figure 4 and in addition it indicates if another computer system, other than a single e-mail system, may be affected. The purpose of this figure is to show which attribute is affected if each of the 14 faults is activated and causes a system failure. Understanding this relationship helps to understand the severity of each fault and therefore how a dependable architecture should be constructed.

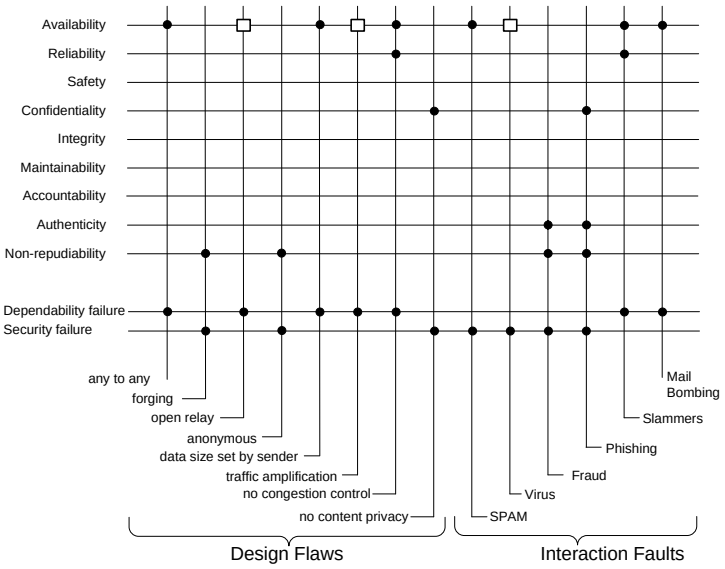


Figure 4- Dependability Attributes affected by Faults and Threats.

In Figure 4, a circle indicates no other computer system is likely to be affected and a square the opposite. We have added the last two rows which indicate if the fault is dependability related or secure computing related.

It should be noticed that Availability is the most affected attribute (9 possible failures), and should therefore demand the most attention, since the uptime will be seriously harmed if appropriate defenses are not implemented.

VII. Available Defense Mechanisms

E-mail service defense mechanisms are explained and classified in this section. They should be understood mainly as "Fault Removal during system use" mechanisms [1]. However, we will use the word "defense" and not "Fault removal" since not all mechanisms remove the fault, but only provide mitigation. Few mechanisms are IETF standards [17] and most of them are MTA software features not enabled by default. In addition, because of the high diversity of MTA software, each of them with different configuration options and third party software, IT staff is typically not aware of the multiple available defense mechanisms for each affected dependability attribute.

Figure 5 shows the effect of the defense mechanisms against each of the 14 Faults. Mechanisms that provide a total solution are depicted with a square icon. Those who only provide mitigation are shown with a circle and a rhomboid indicates a group of mechanisms should be used to provide a full solution. Due to space restrictions, further details can be found in [37].

It is important to notice that 9 of the 14 Faults have a proven solution and the rest have mitigations and await a final solution.

VIII. Proposed E-Mail Architecture

In this section we will start by describing an EA as a simple input/output system and progressively incorporate challenging design issues that a modern EA face in a real internet environment.

A. *E-mail Architecture Basics*

An E-mail Architecture (EA) is a group of servers whose purpose is to process e-mail messages. An EA may consist of a single server or a large geographically distributed group of servers, i.e. small office or a worldwide company respectively. Deploying all of the defense mechanisms detailed in section 4 in a single server has many drawbacks: (1) can cause the configuration to be very complex, (2) the defense mechanisms may not always be compatible with each other and (3) the server represents a single point of failure. In contrast, in a multiple server E-mail architecture it is not always clear where the defense mechanisms should be deployed.

An EA may be seen as an input/output system as shown in Figure 6, in which the EA node transfers email from/into the Internet (email flows 1 and 2) and from/into internal company users (email flows 3 and 4). Numbers represent external email flows (outside the EA) and letters represent internal email flows (inside the EA). A company user is defined as a valid user within the EA, i.e. has valid company user credentials.

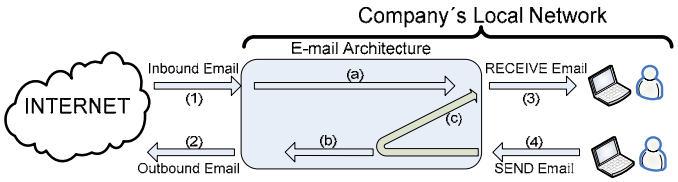


Figure 6- Basic diagram of an E-mail architecture.

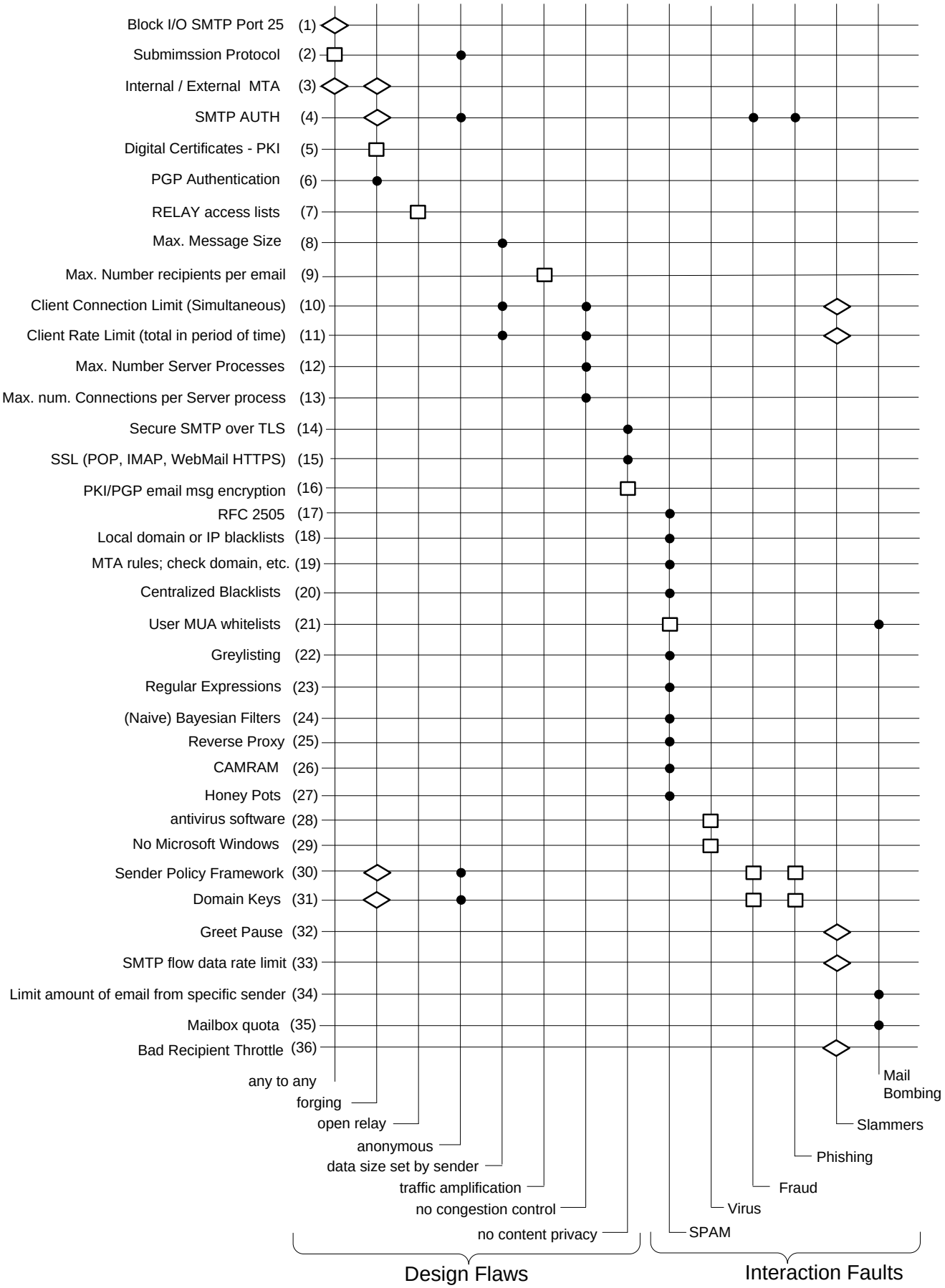


Figure 5- Effect of Defense mechanisms.

From the EA's point of view, incoming email flows are (1) and (4) and outgoing email flows are (2) and (3). If E-mail is not retrieved by users, then accumulation will occur and either the user's mailbox or the server's storage will reach its maximum capacity (Faults probably caused by SPAM, Slammer attach or mail bombing).

Inbound (external) e-mail (1) should always be directed to a local user, otherwise the architecture will be used as an Open-Relay (Section 4); therefore flow (a) is equal to flow (1). Flow (b) represents company user's outbound email (flow b equals flow 2), while Flow (c) shows email from a company user to another company user. Usually, company users send (4) and receive (3) email from within the company network into a separate internal server (inside the EA) and may be subject to special access rules based on user authentication or IP address validation.

It is clear that most faults occur in the internet inbound email flow (1), so many defense mechanisms are usually deployed at this point, leaving the other flows less protected or in some cases unprotected. Usually administrators stop deploying additional defense mechanisms at this stage since a sense of "enough security" is reached.

A more realistic E-mail system is shown in figure 7, since EAs are not quite as simple as described above. This is due to many reasons: company users are often outside the company´s local network (traveling salesman or an employee at home) and users also use third party E-mail systems through webmail (Yahoo, Gmail, etc) (flows 7 and 8). The latter let Virus and SPAM penetrate the inside network bypassing every defense mechanism in the EA.

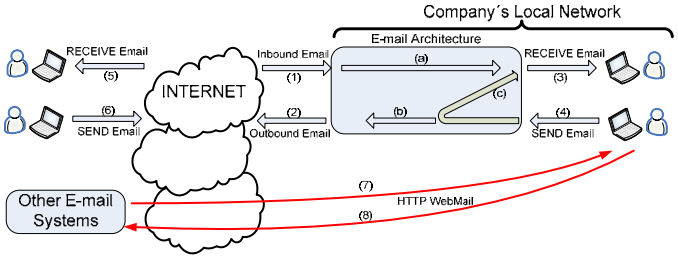


Figure 7- More realistic diagram of an E-mail architecture.

Figure 7 shows that inbound E-mail traffic (1) includes emails from legitimate company users located outside the company (users on the left), so now the EA's defense mechanisms must distinguish between incoming employee email (6) and non-employee incoming email. This complicates every defense mechanism inside the EA since it has to apply strict rules to non-company email and looser rules to company user's email. This would only be possible if each defense mechanism had a built-in user authentication module, which is currently not available.

Large companies have found a not quite elegant solution to this problem shown in figure 8. Company users located outside the company's local network must connect to the company network using an encrypted IP tunnel (9) (Virtual Private Network-VPN) and then send email just as if it were inside. Connections to other E-mail systems from within the company's local network are forbidden (8).

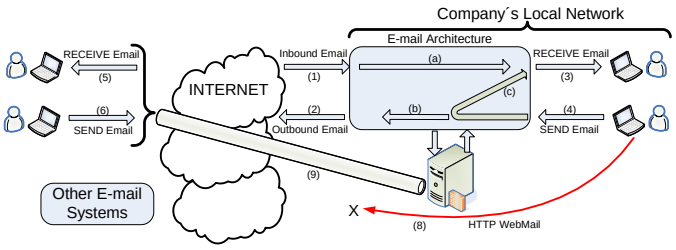


Figure 8- Large Company approach: VPN and Firewalls.

This approach is expensive since a VPN system normally requires a specialized central hardware, software licenses for each user, requires users to perform additional steps and does not guarantee a final solution, since new software like Hopster [29] now promises to bypass any firewall restrictions.

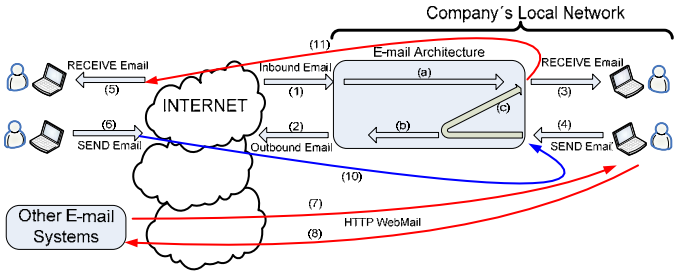


Figure 9- Ideal E-mail Architecture (conceptual).

The ideal solution is to be able to apply a specific set of defense mechanisms only to external email and another set of mechanisms to company users no matter where they are located. Figure 9 shows this idea, where outside company users (MUA) send (email flows 6,10) and receive (email flows 5,11) email using the same process as an inside company user. Mechanisms apply to them even if they access other E-mail systems.

A far more important conclusion is that there is a clear need for a dependable and secure Email Architecture design, that may allow IT staff to deploy the most amount of defense mechanisms to protect itself from each fault. Furthermore, this EA must also be able to accommodate demanding requirements like fault-tolerance through redundancy, scalability through clustering, etc.

Finally, users must be able to access their email in the same way (MUA, WebMail) no matter where they are physically located and user's credentials must be sent through an encrypted connection at all times.

B. Proposed E-Mail Architecture

The proposed EA (Figure 10) may implement all of the defense mechanisms shown in section 7 and complies with the EA shown in figure 9 (ideal solution). The EA is represented by the block in the center of figure 10 and consists of several stages, each identified with a capital letter. Each external flow is numbered from 1 to 6 according to figure 9. A company user may be located inside or outside the company.

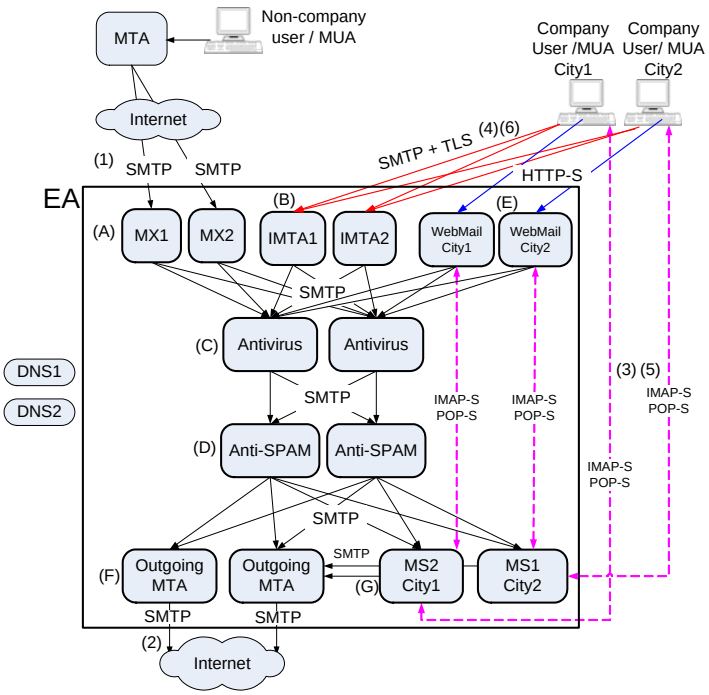


Figure 10- Proposed E-mail Architecture

Each stage (A to G) is a group of SMTP servers performing a specific task and they later forward email to the next stage. The location and function of each stage has been carefully studied; each stage either deploys a specific function and/or a specific defense mechanism. It is important to notice that some stages are not accessible from outside the EA and the rest only allow specific inbound/outbound connections.

Three major features have been incorporated into the proposed EA: 1) For scalability (The potential for a system to continue to function effectively as its size increases) and hardware fault tolerance issues, we represent each stage with 2 servers in an active-backup, load balancing or cluster configuration, 2) Corporate webmail servers (E), and 3) 2 different company user profiles; City1 users and City2 users (this may also be student and professor profiles in a university environment). These features make this architecture a generalization of today's existing architectures.

The EA internal stages are:

(A) **MTA MX (Mail Exchanger):** SMTP servers that receive external incoming emails sent to company users. These servers should represent the only entry point for this type of email.

(B) **MSA (Message Submission Agent)** or IMTA (Internal MTA): SMTP servers that receive incoming email from authenticated company users. The recipient maybe another company user or an external email address.

(C) **Anti-Virus servers**: What to do with an infected email is subject to corporate E-mail policy. It is important to filter or quarantine infected emails before further processing since it may waste other server's CPU processing time.

(D) **Anti SPAM servers**. What to do with an SPAM email is subject to corporate E-mail policy.

(E) **Corporate Webmail servers**. These servers should accept only HTTPS traffic from outside the EA. Some systems only force https connections at the authentication stage and then switch to unencrypted http (Gmail, Yahoo, etc.) to lower CPU load. This allows any intermediate user to sniff (spy) the http traffic and therefore every email.

(F) **Outgoing MTA servers**. Only these servers should be allowed outgoing SMTP traffic to the internet to comply with the Sender ID Framework [23] or Domain Keys [24] defense mechanisms.

(G) **Message Store servers** for each company profile. Authenticated users retrieve their email from these servers using standard protocols like POP3 or IMAP only with SSL encryption enabled.

DNS servers are essential for the EA to work. For security reasons, these DNS servers should not be accessible from outside the company and only be used as DNS resolvers by the EA servers since they are prune to several attacks [30].

Inbound external emails (1) is received by the Mail Exchanger SMTP servers (A), later checked for Virus (C), then for SPAM (D) and finally stored in the corresponding Message Store Server (G). Company users will then retrieve this email through a Secure IMAP/POP connection (flows 3,5) to its corresponding MS Server (G) or through a Secure HTTP connection to its corresponding WebMail server (E).

Company Users (City1 or City2) send emails using the SMTP Protocol with TLS encryption (flows 4,6) to the Internal (company user only) MTA stage (B) with a required authentication mechanism like SMTP AUTH. Then the email will be checked for Virus (C) and SPAM (D) and later either sent to the Internet through the F stage or to the corresponding MS server (G). Stage F servers are the only outgoing SMTP connections allowed to the Internet.

The difference between stages A and B is that different defense mechanisms apply to external users and internal users respectively. Each email is checked for Virus and SPAM no matter who originated it.

Every EA design that has user accounts defined in one stage (MS servers stage G or in a centrally located LDAP server) and the receiving SMTP servers in another stage (stage A and B) are eligible to the Backscattering problem [35]. Spammers deliberately send email to an obviously non-existing email address (i.e. hjjhsdkjhsd@yourcompany.com) with a forged sender email address (the victim). This will cause your EA to send thousands of "User Unknown" notifications with SPAM content to external users (the victims). The solution is that servers in stages A and B must be able to check if each recipient is a valid Company user email address, therefore rejecting email for "unknown" users. This may sound trivial, but 30% of Fortune 500 companies are sending backscatter email [36].

TABLE I- PROPOSED LOCATION OF DEFENSE MECHANISMS

Defense Mechanism	Location
1) Block Incoming/Outgoing SMTP Port 25	network perimeter
2) Submission Protocol [18]	B
3) Internal / External MTA	A, B
4) SMTP AUTH [3]	B
5) Digital Certificates – PKI [19]	B, E, G
6) PGP [20]	B, E, G
7) RELAY access lists	A, B
8) Max. Size per email	A, B, E
9) Max. Number recipients per email	A, B, E
10) Client Connection Limit (Simultaneous) [21]	A, B
11) Client Rate Limit (total) [21]	A, B
12) Max. number Server processes	A,B,C,D,F, G
13) Max. number of connections per Server process	A,B,C,D,F, G
14) Secure SMTP over TLS [3]	B
15) SSL (POP, IMAP, WebMail HTTPS)	B,E,G
16) PKI/PGP email message encryption [19][20]	B
17) RFC 2505 [17]	A, B
18) Local domain or IP blacklists	A
19) MTA rules; check domain, etc.	A, B
20) DNSBL (Centralized BL)	A
21) User MUA whitelists	A, MUA
22) Greylisting [26]	A
23) Regular Expressions	A, B, D, MUA
24) Bayesian Filters [22]	D, MUA
25) Reverse Proxy	D
26) CAMRAM [27]	A, D
27) Honey Pots [28]	A
28) antivirus software	C, MUA
29) not use Microsoft Windows	A-G, MUA
30) Sender Policy Framework SPF [23]	A, F
31) Domain Keys [24]	A, F
32) Greet Pause	A, B
33) Data rate (bps) limit of each SMTP flow	A, B
34) Limit amount of email from specific sender [25]	A, B
35) Mailbox quota	G
36) Bad RCPT Throttle	A,B

We emphasize that the proposed EA can implement all the defense mechanisms available today and since the design is based on stages and standard IETF protocols, it is easy to add a new stage in between to deploy a new defense mechanism; hence this EA is flexible and allows future enhancements. In contrast, this EA requires that the IT Staff work with complete knowledge of each component and therefore keep each server and software documented and updated. Maintenance is quite simple since each stage is in server pairs, so each server may be maintained simply by disabling it from the pair. The exact procedure depends on how you build this pair (cluster, software load balancing, etc.)

C. *Location of Defense Mechanisms*

The defense mechanisms described in section 7 are applied to one or several servers inside the EA. A common mistake is to just place a group of anti-virus, anti-spam and firewalls in front of the corporate SMTP server. This may work for some faults and threats, but those systems fail today because it is not seen as a whole architecture, but just a set of *hardware patches* for SMTP. As an example, it is not the same to place an anti-virus before the anti-spam appliance than the other way around, since E-mail delivery may be delayed significantly, because there is no point in checking a virus infected email for SPAM if that email will be later discarded on the anti-virus stage. It must be understood that these defense mechanisms (countermeasures) should be deployed as part of a global strategy within a systematic and periodic process along with selecting the best location(s) for that defense mechanism inside the EA.

Since there are many defense mechanisms for each of the 14 Faults, IT staff must choose the best suitable combination according to their company policy. Table I shows the exact location, inside the EA (figure 10), of each of these defense mechanisms.

D. *Applied Example: "F" University*

As an example, we modified the EA of the "F" University from a classic infrastructure (figure 11A) to the proposed EA in figure 10. This migration process is still in course, so we will present the current EA at the time being (figure 11B). This EA was substantially modified in 2004, so we will present data before and after this year, namely 2002 and 2006. This EA processes 68.000+ emails per day for 1.200 faculty/staff and 17.000 student accounts.

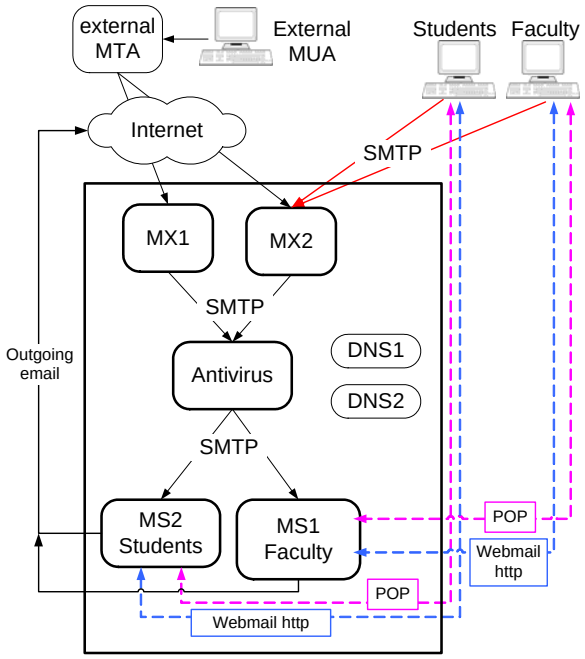


Figure 11A- "F"'s EA in 2002

Table II shows the deployed defense mechanisms in both EAs. EA-2002 has only 2 solved and 1 partial solution to 3

faults with 3 defense mechanisms. In contrast, the EA-2006 has 7 full and 7 partial solutions with 21 defense mechanisms, therefore having solutions/mitigations for each fault. Every single component uses free open source software, so the difference in cost is only 2 medium range UNIX servers, roughly an increase of 30% of infraestructure. The final result is a stable and dependable E-mail service with a higher user confidence level, therefore the benefit/cost is highly atractive.

TABLE II- COMPARISON OF DEPLOYED DEFENSE MECHANISMS

Fault	EA in 2002	EA in 2006
any to any	none	**solved**: Block Port 25 (DM1[1]), "Internal/ External MTA" (DM3)
forging	none	**solved**: "Internal/ External MTA" (DM3), SMTP AUTH (DM4), SPF (DM30)
open relay	**solved**: Access Lists (DM7)	**solved**: Access Lists (DM7)
anonymous	none	partial: SMTP AUTH (DM4), SPF (DM30)
data size set by sender	none	partial: Max size per email (DM8), Client Connection (DM10) and Rate Limit (DM11)
traffic amplificatio n	none	**solved**: Max # recipients per email (DM9)
no congestion control	none	partial: Client Connection (DM10), Client Rate (DM11), Max Processes (DM12), Max. Connections per Process (DM13), Bad RCPT Throttle (DM36)
no privacy	none	partial: SMTP+TLS (DM14), SSL (POP, IMAP, WebMail HTTPS) (DM15)
SPAM	none	partial: RFC 2505 (DM13), MTA rules (DM19), Greylist (DM22), Bayesian Filter (DM24)
virus	**solved**: SMTP Antivirus (DM28)	**solved**: SMTP Antivirus, MUA AV (DM28)
fraud	none	**solved**: SMTP AUTH (DM4), SPF (DM30)
phishing	none	**solved**: SMTP AUTH (DM4), SPF (DM30)
slammer	none	partial: Client Connection (DM10), Client Rate (DM11), Greet Pause (DM32)
mail bombing	partial: Mailbox quota (DM35)	partial: Mailbox quota (DM35)

Complete numerical results of the deployed defense mechanisms in the "EA in 2006" are documented in [37]. Here, due to space limitations, we limit ourselves to present only the interesting effect obtained when we added a Backscattering Defense Mechanism (DM19) to the EA on May 15th 2006.

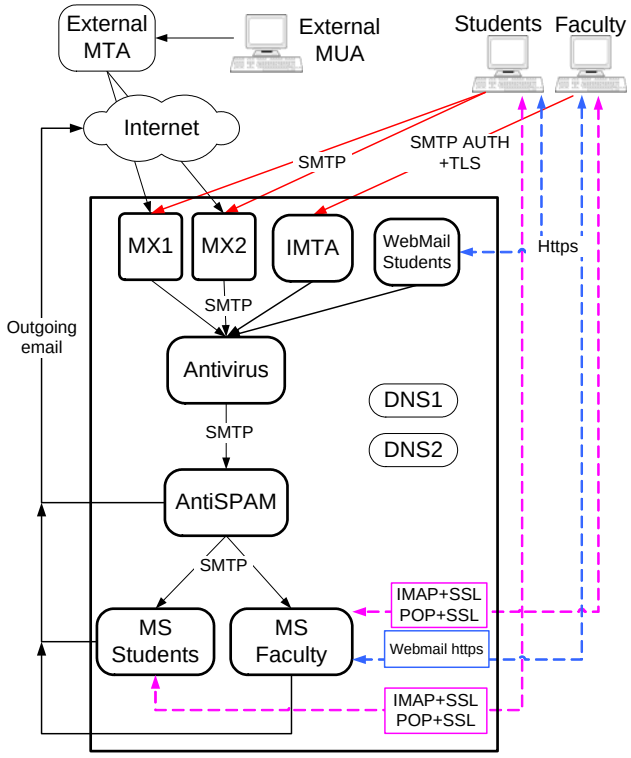


Figure 11B- "F"'s EA in 2006

A basic MTA rule is to reject email for unknown users, but in most cases, the receiving MTA is only a relay (MX server) for the Mail Storage Server and therefore does not have knowledge of the valid users, this MTA cannot perform this check and will accept email TO any user within its domain, i.e. will accept email to ksjhfskjdfh@yourdomain.com. As previously explained, this is likely to be abused by spammers through Backscattering [35]. This is exactly what happened in "F" University's EA before May 15th 2006 as clearly shown in Figure 12. From this day on, the average rejection rate is approximately. 74,939 emails per day.

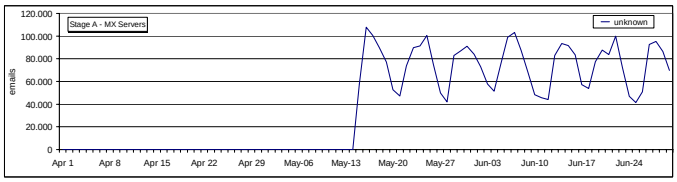


Figure 12- Defense Mechanism (19): MTA Rules. MX Servers (A).

The backscattering problem was solved by simply copying a list of valid users from each MS server (Stage G) to the MX Servers (Stage A) each time there is a modification to the user database in an automated process, and therefore rejecting emails to unknown users (DM19).

This deployment caused other defense mechanisms to significantly reduce their workload (Fig. 14, 15) and the amount of SPAM significantly reduced, as shown in figure 13.

Figure 13 shows the amount of SPAM and HAM (not SPAM) email. The SPAM peak is 46,508 emails on May 12th. Since the Backscattering problem was solved at May 15th, the daily average amount of SPAM significantly was reduced from 12,679 to 4,067. The amount of HAM is a periodic curve denoting Monday-Friday normal activity similar to the internal telephone activity.

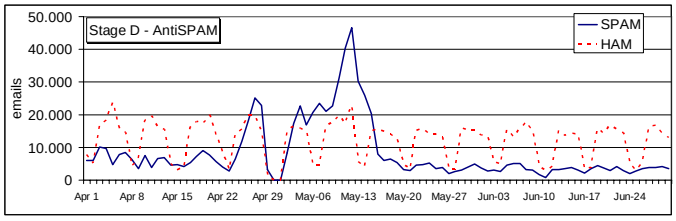


Figure 13- Defense Mechanism (23, 24): SPAM general statistics. SpamAssassin. Stage D

Figure 14 shows the amount of temporarily rejected connections due to the Greylisting mehanism. This workload was also significantly reduced from 149,566 to 46,487 average daily rejections, because of the Backscattering solution applied on May 15th. Note that this Greylisting feature is configured to temporarily reject email from all domains except if they are SPF complaint (SPF_PASS), therefore "well behaved" domains are not subject to multiple defense mechanisms.

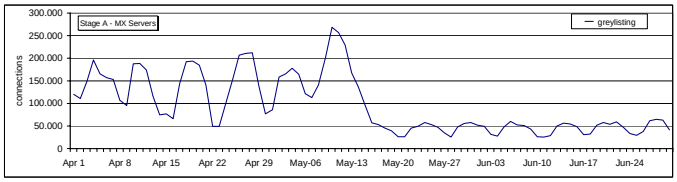


Figure 14- Defense Mechanism (22): Greylist

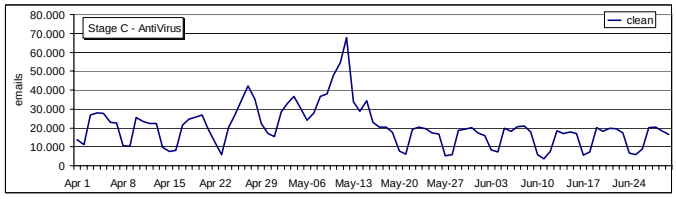


Figure 15- Defense Mechanism (28): SMTP Antivirus

Figure 15 shows the amount of non virus-infected emails processed at the Antivirus MTA - Stage C. This workload was also significantly reduced from 25,705 to 15,568 average daily emails, because of the Backscattering solution applied on May 15th.

Another 2 examples of the application of this methodology to working (in actual use) E-mail architectures, a private company and another university, is shown in [37].

IX. Conclusion

Until now, e-mail architectures are mainly designed based on performance issues and/or budget constraints. Security is incorporated into the design, but mainly for user/password encryption. Due to high work load, IT staff usually applies defense mechanisms after a failure has occurred and in most cases, they simply add a new appliance or server in front of their mailbox servers. Only few E-mail architectures provide a broader range of defense mechanisms for both Dependability and Secure Computing attributes.

In this work a complete review of the current SMTP weaknesses and threats to the E-mail service is presented, classifying them according to the most recent dependability taxonomy [1], along with a list of state of the art defense mechanisms for each weakness. This classification of faults, failures and defense mechanisms has demonstrated to be complete enough to provide a useful tool when designing and deploying an E-mail architecture.

Finally, using this classification, a secure and dependable E-mail Architecture that implements the most efficient defense mechanisms available today is proposed. It is flexible enough to add new stages or defense mechanisms in the future. Practical examples and measurements of the effectiveness of several defense mechanisms are shown along with their interaction.

This methodology and proposed architecture is very useful and of key importance to e-mail IT staff, because it will help them redesign, modify and plan future enhancements to their e-mail infrastructure in a proactive, preventive and systematic process.

References

[1] Algirdas Avizienis, Jean-Claude Laprie, Brian Randell, Carl Landwehr.: Basic Concepts and Taxonomy of Dependable and Secure Computing. IEEE Transactions on Dependable and Secure Computing, Vol. 1, No. 1, Jan-Mar 2004. page 11.

[2] J. Klensin "Simple Mail Transfer Protocol" RFC 2821, April 2001. http://www.ietf.org/rfc/rfc2821.txt

[3] SMTP Service Extensions, http://www.imc.org/rfcs.html

[4] Internet Mail 2000. http://www.im2000.org/

[5] Jussi Kangasharju, Keith W. Ross, David A. Turner. "Secure and Resilient Peer-to-Peer E-Mail: Design and Implementation," IEEE p2p, vol. 00, no. , p. 184, Third 2003.

[6] David A. Turner, Keith W. Ross. A Comprehensive Architecture for Continuous Media e-mail. IEEE MultiMedia, Volume 8 , Issue 2 (April 2001), Pages: 88 – 98

[7] Anthony Lang. Email Dependability. Bachelor of Engineering (Telecommunications) Thesis, November 2004, Supervisor: Dr Tim Moors. http://uluru.ee.unsw.edu.au/~tim/projects/dependable_email/

[8] Benchmarking e-mail Dependability. The Berkeley/Stanford Recovery-Oriented Computing Project (ROC). http://roc.cs.berkeley.edu/retreats/spring_02/poster_slides/mailbench-v2.ppt

[9] Cardellini, V., Colajanni, M., Yu, P.S. A High Performance System Prototype for Large-scale SMTP Services. Internet Computing, IEEE , Volume: 3 Issue: 3 , May/Jun 1999

[10] Brad Knowles, Nick Christenson. Design and Implementation of Highly Scalable e-mail Systems. LISA XIV, 8 Dec 2000

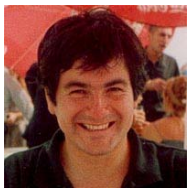
[11] Tim Bass, Alfredo Freyre. Email bombs and Countermeasures: Cyber Attacks on Availability and Brand Integrity. IEEE Network, Vol. 12, No. 2, pp. 10-17, March/April 1998

[12] J. Robert von Behren, ByungHoon Kang. A Secure Email Infrastructure for Computationally Weak Clients

[13] The Spamhaus Project, http://www.spamhaus.org/definition.html

[14] Spam Statistics 2004, http://spam-filter-review.toptenreviews.com/spam-statistics.html
[15] CAUCE, Coalition Against Unsolicited Commercial Email, http://www.cauce.org/
[16] Unsolicited Bulk Email: Definitions and Problems http://www.imc.org/ube-def.html
[17] Internet Engineering Task Force, http://www.ietf.org
[18] R. Gellens (Qualcomm), J. Klensin (MCI), “Message Submission”, RFC 2476, Diciembre 1998, http://www.ietf.org/rfc/rfc2476.txt
[19] “Public Key Infraestructure” and SMTP. http://www.imc.org/rfcs.html#security
[20] “OpenPGP y PGP”. http://www.imc.org/rfcs.html#security
[21] Sendmail, http://www.sendmail.org/
[22] Paul Graham, “A plan for Spam”, August 2002, http://www.paulgraham.com/spam.html
[23] Sender Policy Framework, http://www.openspf.org/
[24] Yahoo! Domain Keys, http://antispam.yahoo.com/domainkeys
[25] Milter-Limit, http://www.snertsoft.com/sendmail/milter-limit/
[26] Greylist Milter, http://hcpnet.free.fr/milter-greylist/
[27] CAMRAM Hybrid Antispam System “Campaign for Real Mail”, http://www.camram.org/
[28] Honey Pots. http://www.projecthoneypot.org/
[29] Hopster, http://www.hopster.com/
[30] Securing an Internet Name Server, http://www.cert.org/archive/pdf/dns.pdf
[31] Phishing Attacks, Anti-Phishing Working Group, http://www.antiphishing.org/
[32] Netscape, Inc. *Mailstone Utility*. http://docs.sun.com/source/816-6036-10/index.html
[33] Standard Performance Evaluation Corporation. SPECmail2001, http://www.spec.org/osg/mail2001/
[34] The Radicati Group, http://www.radicati.com/index.asp
[35] Backscatter http://spamlinks.net/prevent-secure-backscatter.htm
[36] Backscatter Fortune 500 victims: http://www.techzoom.net/paper-mailbomb.asp
[37] Full Paper version: http://www.maraboli.cl/papers/M_Arch_DSES.pdf

Marcelo A. Maraboli received the Professional title of Electronic Engineer (1997) from Universidad Técnica Federico Santa Maria <http://www.usm.cl> (UTFSM, Valparaiso, Chile) and is a MSc (C) of Electronic Engineering at the same University.
He currently works as the CHIEF NETWORK AND SYSTEMS ENGINEER of the Universidad Técnica Federico Santa Maria. His main research areas include Fault Tolerant Internet Services and Network Security.
Maraboli is also a Certified Information Systems Security Professional (CISSP), Washington D.C., 2002.



Reinaldo Vallejos received the BEng in Electronic Engineering (1975) and the Professional title of Electronic Engineer (1976) from Universidad Técnica Federico Santa Maria <http://www.usm.cl> (UTFSM, Valparaiso, Chile) and the degrees of MSc in Computer Science (1990) from the Pontificia Universidade Católica do Río de Janeiro (Brasil) and PhD in Computer Science (1993) from the Universidade Federal do Río de Janeiro (Brasil). Currently, he is a Professor in the Electronic Engineering Department of UTFSM.