Global Information Assurance Certification Paper

Spam and Secure MTAs: an Overview of Fedora, Sendmail and MailScanner

GIAC Security Essentials
Certification (GSEC)
Practical Assignment
Version 1.4b
Option 1 - Research on Topics
in Information Security

Submitted by: Andrew Lord
Location: SANS Orlando 2004

Paper Abstract: Building a Secure Antispam MTA using Fedora Core 2, Sendmail and Mailscanner

Table of Contents

List of Figures

Abstract/Summary

It would be impossible to discuss e-mail without addressing the biggest threat to e-mail communication today – spam. There are many technologies, including future ones that promise to alleviate e-mail vulnerabilities.

This paper focuses on an overview a secure MTA using Fedora Core 2, Sendmail and Mailscanner along with several complementary open source projects such as SpamAssassin, ClamAV, Mailwatch and Sendmail's milter interface. This paper also examines the application of modern information security concepts to real-world technical implementation and ends with a brief glimpse of the future of e-mail: sender authentication and reputation systems.

Introduction

Spam. In and of itself a fairly innocuous-sounding word. But why not? Certainly when Kenneth Daigneau coined the term to describe Hormel's Spiced Ham[1], he had no inclination of the frequency with which it was to appear in today's headlines, or of its association with cybercrime nearly thirty years later.

What is spam? The Merriam-Webster Online Dictionary defines e-mail spam as "unsolicited usually commercial e-mail sent to a large number of addresses"[2]; the origins of spam seem to be untraceable to a specific moment in time, but rather evolved from a series of similar attempts to get everyone on the Internet's, or the ARPANET's, attention.[3] One thing is clear: the possibility of receiving unwanted messages was known to the developers of e-mail before the protocol was fully developed[4]. Had they considered the widespread usage of e-mail as the medium for business communication that it is now, the required security would have probably been architected into the original design.

Today's Message Transfer Agents (MTAs) need to be designed to protect us from the more than 600 million spam messages scanned by MessageLabs which represents a small fraction of the more than 60% of e-mail traffic that is spam on the Internet today[5]. The business requirements are almost oxymoronic; security must be architected into the system from conception, it must be fast and efficient and the components must be updated frequently to keep pace with the times. It should be cost effective and easy to maintain and most of all, simple to deploy.

These complex business requirements of today's world have given rise to the popularity of the open source movement in general and Linux in particular. Linux has risen with the growth of the Internet itself, with higher and more freely available bandwidth, it is impossible to count the number of copies of this freely available operating system in use. The 2.6 kernel was largely developed as the culmination of enterprise-ready Linux. The business benefits of Linux are clear; it is free (although there can be significant cost associated with installation, maintenance, support and training), most of the core components are maintained daily by volunteer developers around the world, the source code is available and its upgrades are feature and quality-driven, as opposed to revenue or marketing driven. These elements combine to form the most important benefits to businesses – choice and flexibility.

The purpose of any MTA is to transfer messages for delivery to the internet and/or to receive messages from the Internet.

---

[1] http://www.marketingvox.com/archives/2003/06/03/a_brief_history_of_spam_lunch_meat_monty_python_junk_email/
[2] http://www.webster.com/cgi-bin/dictionary?book=Dictionary&va=spam&x=12&y=11
[3] http://www.templetons.com/brad/spamterm.html
[4] http://community.roxen.com/developers/idocs/rfc/rfc706.html
[5] http://www.messagelabs.com/emailthreats/default.asp

The threat to e-mail infrastructure is hybrid: abuses, viruses, worms, social engineering attacks/phishing scams, spam, intrusion, denial-of-service or any combination of these things represent a threat. The threat agents are just as diverse; hackers, users, malware, acts of God and more. All of these threats when coupled with unmitigated vulnerabilities in our infrastructure represent a risk. A *threat matrix* can help us identify risk and proactively improve the security posture of our infrastructure. It helps to keep the bedrock principles of security foremost in our mind: Confidentiality, Integrity and Availability. Most of e-mail's vulnerabilities center on integrity and availability. Sending e-mail is like sending a postcard*, its store-and-forward mechanism and reliance on headers in the message for routing and authentication mean that it was not designed to guarantee confidentiality (or integrity for that matter) by itself. Encryption and digital signatures, respectively, would need to be layered on to provide this.

The Evolution of the Spam Threat

Security Administrators are not the only ones studying information security trends. The use of increasingly sophisticated techniques to disguise spam is becoming more and more prevalent today. CipherTrust, the maker of the IronMail e-mail security appliance outlines the basic methods of spam-filtering, and how spammers fool spam filters[6]. The e-mail security vendors and spammers have played out a one-upmanship game with technology: signature-based filtering, rule-based filtering, blacklisting, whitelisting and Bayesian filtering have all met their come-uppance in the antispam wars. CMP's EETimes even suggests that open source may be part of the problem, "Whenever a new version of such open-source software is released, spammers simply download it and learn all the ways to escape detection by it. After all, it's free."[7] They go on to highlight several more techniques, including obfuscation, intentional misspelling, embedded and (Base64 or other) encoded text, sometimes included on a same-colour background to be invisible to the reading recipient, but effectively confusing the spam filter.

These latest techniques are evolving into substitution and in some cases URL encoding. Spammers are using HTML and images in portions or whole images to spam. Another increasingly effective tactic is phishing, or using websites to impersonate corporate entities and to trick e-mail recipients into divulging personal or financial information. The ISC Handler's Diary Archive for August 2004 makes mention of four separate phishing-related logs, and the Anti-phishing Working Group website graphs both weekly phishing attacks and cumulative phishing attacks steadily increasing during the period April 2004 to June 2004[8], with recent phishing attacks occurring as often as every three days.[9]

[6] http://www.ciphertrust.com/resources/articles/articles/foolspam.html
[7] http://www.eet.com/in_focus/communications/showArticle.jhtml?articleID=23900564
[8] http://www.antiphishing.org/images/chart_08-04-04.gif
[9] http://www.antiphishing.org

Some flawed filtering software made assumptions about signed e-mail and its spam content which prompted the use of PGP signatures by spammers in an effort to reduce the spam score of the messages sent. Any PKI-based antispam measure is sure to result in spammers going out and purchasing certificates; the resulting encrypted spam would surely be overlooked by current antispam methods, or be too expensive for the average MTA to be able to decrypt, scan and forward in large volumes in a reasonable time.

Threats and Risk Mitigation – Defense In-Depth

In order to be successful at defending our MTA asset, we must not only make it secure, but we must create a secure environment around it. This layered approach to security is called defense in-depth.
To successfully protect information, defenses must be built at each stage of the information environment. A common classification for the layers looks at the flow of information from the network to the computer hardware, to operating system, to the application.

At the Network classification our defenses need to protect us against denial-of-service attacks and allow the legitimate transfer of data to and from our MTA. Routers provide us with connectivity to the Internet and basic ingress filtering allow us to filter packet-based attacks such as IP-spoofing. As with any Internet-accessible service, a firewall is an absolute necessity and is usually the first line of defense against any attack at the OSI Transport Layer up to the OSI Application Layer. All Internet "noise" is filtered at this level, and the only traffic that should be permitted should be that which is required to operate the services on our MTA. Inbound rules should be different to outbound rules; our MTA like all e-mail servers requires at a minimum DNS to route e-mail to Internet hosts, however all DNS queries from the Internet should be directed to a designated DNS server, and preferably not the MTA. E-mail servers should generally be isolated on a separate DMZ, disconnected from the internal network as well as not connected to the Internet directly.

Defenses: packet-based filtering, firewalling.

At the heart of hardware selection is performance and availability. Many organizations select desktop systems to perform dedicated server tasks to reduce cost. While this may be acceptable in the case of less critical applications where e-mail volume is minimal, one must carefully consider the worst-case cost of doing without a mail server for several hours due to hardware failure when e-mail communication to another business partner is critical. A quick lesson in quantitative risk analysis is in order here. If the cost to recreate or recover the MTA, plus the cost of not being able to communicate at the most crucial time multiplied by the probability of occurrence is more than the cost to implement hardware fault tolerance (known as Annualized Loss Expectancy or Annualized Loss Exposure), then purchasing dedicated server hardware is probably a wise idea.

Defenses: backup and recovery software, hardware fault-tolerance, high-availability, hardware-based monitoring.

Another lesser contemplated risk is that of using cheaper or less popular hardware, particularly with open source software. The consideration has to do with the way open source hardware support is developed and maintained and license considerations. At times, companies that sell low cost hardware implement reference designs from the chipset manufacturer. Often, there is little development of driver software beyond the reference software provided by the chipset manufacturer, and Linux developers may be limited to what documentation the manufacturer considers to be public domain as opposed to trade secret or proprietary license.
Developers may create a driver based on the reference design, the driver may contain binary-only code from the vendor or the driver may be generic to a family of chipsets. While these modules may work well under light loads, in a server environment with several hundred requests per second the code may be unstable, exhibit memory leaks or intermittently crash. In the open source world where software is constantly in flux and motivation is often non-monetary, code that exhibits this behavior that is unpopular is often either rewritten from scratch or no longer maintained by the developer and abandoned to the annals of history. Larger vendors and manufacturers on the other hand, will have a vested interest and larger resource base for developing and maintaining driver modules, even for mature hardware. Such was the case with the Broadcom NIC Linux driver module which was only stabilized in the 2.6 kernel series.

Finally, different types of hardware can afford you some minor additional layers in to add to your defenses. Server-class hardware BIOSes may include chassis tamper notification and alarms, BIOS reconfiguration alarms and the ability to filter hardware monitoring data to the operating system level to be remotely monitored by SNMP or other proprietary solutions.

A major attraction to open source servers is the wide range of features offered by the kernel itself. Linux's kernel development community has contributed a number of security features that can be turned on from within the kernel, not the least of which is the kernel-level stateful-inspection firewall, iptables. Among the more useful features are kernel-level logging, source-NATing, stateful filtering (connection tracking) and rate-limiting. There are also extensions merged into the kernel for file-system ACLs and to implement a role-based access control (RBAC) mechanism, a type of Mandatory Access Control (MAC) within the kernel with built-in auditing (SELinux). Unfortunately, the design of this secure MTA requires a number of open source components to interact with each other, and this is suboptimal to the current SELinux implementation in Fedora Core[10]. Additionally, while these controls can create extremely secure implementations of Linux, there is significant effort required to maintain this in a dynamic environment which reduces the feasibility of regular updates. Patching becomes

[10] http://fedora.redhat.com/docs/selinux-faq-fc2/

more difficult to implement and verify which affects availability and therefore overall security may be reduced.

Defenses: kernel-based firewalling, host-based intrusion detection, operating-system hardening, vulnerability assessment, penetration testing

Software Configuration

Fedora Core

The Fedora Core project is the direct descendant and replacement for Red Hat Linux 9. It includes all of Red Hat's technology and according to the website, "is also a proving ground for new technology that may eventually make its way into Red Hat products."[11] Fedora Core 2 was released in May 2004 and has undergone sufficient testing to be considered a stable Linux distribution. Fedora Core was selected on the basis of its popularity as a distribution, its compatibility with Red Hat and its overall support within the community.

Sendmail

Sendmail is one of the most popular MTAs on the Internet. Sendmail is also Fedora's default MTA but Fedora is configured to be able to be switched to Postfix, a more recent Sendmail-compatible alternative.
Mailscanner however is not fully compatible with Postfix, and these issues are noted on the Postfix Add-on Software website[12].

Mailscanner with SpamAssassin, ClamAV, Mailwatch

Mailscanner is billed as an open source e-mail security system. This software is used to provide the antispam and antivirus management functionality of our MTA.

[11] http://fedora.redhat.com/
[12] http://www.postfix.org/addon.html#content

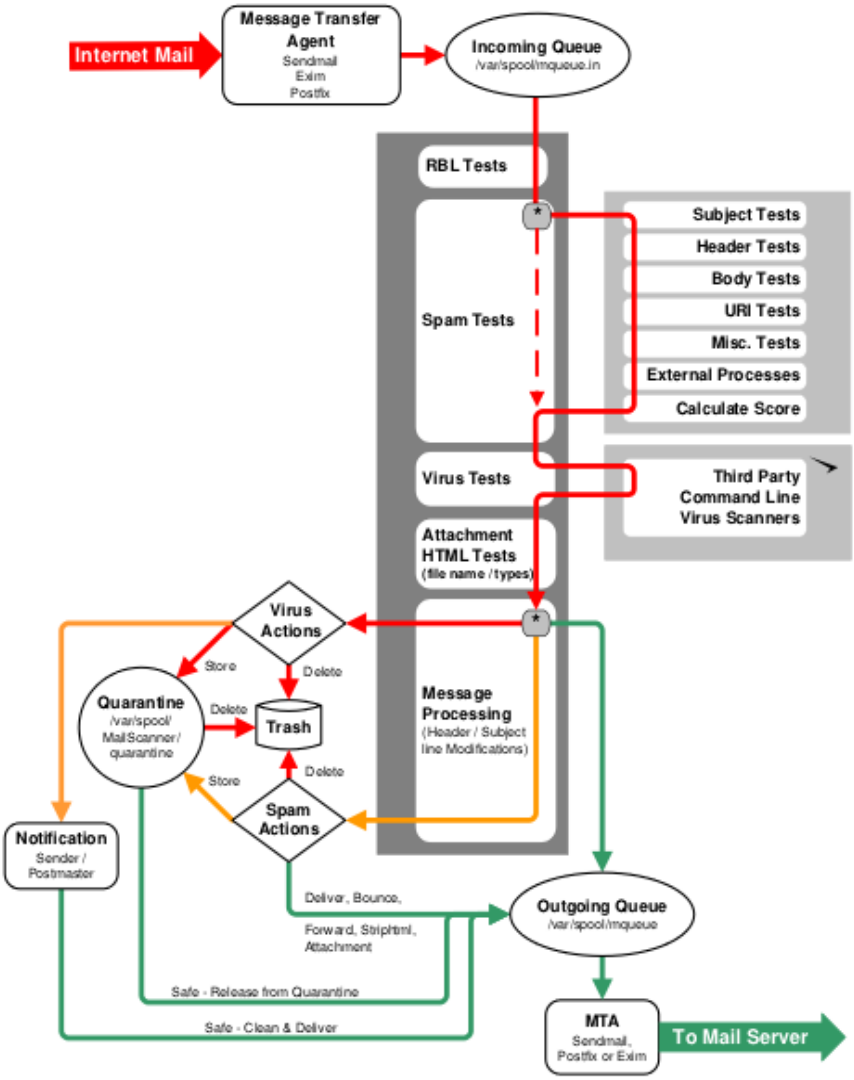

Figure 1 - Mailscanner Process Flow[13]

The overall GNU/Linux philosophy is that of many small, single-purpose software utilities that are combined together to provide more sophisticated functionality; Mailscanner manages two sendmail processes and two sendmail queues in series. One is used to collect messages for scanning and the other is used to delivery messages after being filtered by Mailscanner. Mailscanner spawns a number of child processes to perform the scanning and filtering functions that take place between the two queues, and restarts these processes on a schedule specified in its configuration file. These child processes are SpamAssassin and

[13] http://www.fsl.com/whitepapers/Fortress_SMGateway_Architecture_Diagram.pdf (page 3, Figure 2)

ClamAV (or any other supported 3rd party virus scanner) processes. MailScanner 4.34 also includes protection against all known Outlook, Outlook Express, Internet Explorer and Eudora security vulnerabilities, does HTML stripping of messages and can archive messages or forward mail messages[14]. Another important feature is the ability to decode MIME and to filter Base64 encoded messages[15]

SpamAssassin is an extensible email filter which is used to identify spam[16]. It is now maintained by the Apache Software Foundation and as such version 3.0.0 and later are licensed under the Apache Software License version 2. (This license is approved by the Open Source Initiative[17] and the change is transparent to most end users.) SpamAssassin uses many of the tricks in the book to help identify spam: header analysis, text analysis, blacklisting, Bayesian learning classifier and distributed (spam) hash databases such as Pyzor, Razor and DCC. Version 3's feature list includes the denial-of-service countermeasure hashcash which is basically a white list mechanism for individual e-mail messages and an updated Perl API which is now supported by Mailscanner.

Clam Antivirus, or ClamAV as it is known, is a GPL command-line scanner for UNIX and Linux. Its native support for RAR, Zip, Gzip, Bzip2, Mbox, Maildir, raw mail files, detection of over 20,000 viruses, worms and trojans, and free, up-to-date virus database[18] make it ideal for use in an e-mail filter environment.

MailWatch for Mailscanner is a monitoring and reporting solution with some minor management capabilities. It is written in PHP and therefore requires Apache (or a PHP-compliant web server) to be running on the MTA.

Bastille

An important layer of defense, Bastille is used for hardening the operating system of the host it is applied to. Bastille is a series of shell scripts and Perl modules that implement and test the security of a Linux host: a software-based checklist of best practices that a sysadmin might use to lock down a production environment before deployment.

Implementation Gotchas and Maintenance

Fedora Core 2 was installed with the following features:

- A Custom installation was selected.
- A bootloader (GruB) password was applied.
- The firewall was enabled and only SSH and Mail (SMTP) allowed through.

[14] http://www.sng.ecs.soton.ac.uk/mailscanner/newinv4.shtml
[15] http://www.sng.ecs.soton.ac.uk/mailscanner/#news
[16] http://spamassassin.apache.org/
[17] http://opensource.org/licenses/
[18] http://www.clamav.net/abstract.html#pagestart

Mailscanner required the turning off of the Fedora Core Sendmail SysV initscript and the turning on of the MailScanner initscript.

```
#service sendmail stop
#chkconfig sendmail off
#chkconfig --level 2345 MailScanner on
#service MailScanner start
```

Mailscanner utilizes a script in order to create RPMs for the Perl modules it requires. It also prepackages the required modules together with its RPMs in a tarred and gzipped file. To manually install additional Perl modules for ClamAV and SpamAssassin, you would need to locate them, download and do a manual installation, or install them from the CPAN shell.

```
tar xzf <module>.tar.gz
# cd <moduledir>
# perl Makefile.PL
# make
# make test
# make install
```

OR

```
#perl -MCPAN -e shell

If you do not want to enter a dialog now, you can answer 'no' to this
question and I'll try to autoconfigure. (Note: you can revisit this
dialog anytime later by typing 'o conf init' at the cpan prompt.)

Are you ready for manual configuration? [yes] no
cpan> install <module>
cpan> quit
```

Updated LDAP libraries on the MTA prevented Sendmail from starting:

```
Starting MailScanner daemons:
        incoming sendmail: /usr/sbin/sendmail: error while loading shared libraries:
libldap.so.2: cannot open shared object file: No such file or directory
                                                   [  OK  ]
        outgoing sendmail: /usr/sbin/sendmail: error while loading shared libraries:
libldap.so.2: cannot open shared object file: No such file or directory
                                                   [  OK  ]
        MailScanner:                               [  OK  ]
```

The libraries had had their names changed. The problem was quickly resolved by locating the new libraries and creating symlinks to the old.

Testing with GTUBE, EICAR

To test the correct functioning of the MTA, the EICAR test string[19] was forwarded to MailScanner. To test the antispam functionality, a GTUBE[20] was used. Both of these trigger the functionality of the MailScanner filters by using test strings of characters that the software responds to and logs.

Thinking like a hacker

To prevent internal intrusions, a vulnerability assessment should be conducted. Nessus excels at this, although it can be very slow on a secured box. One of the keys to remaining secure is in reducing fingerprint the operating system and application fingerprint by disabling the default banner Sendmail displays. This is achieved by editing /etc/mail/sendmail.mc and running the following command from in the /etc/mail directory:

```
make -C /etc/mail
```

All intrusions begin with reconnaissance – information gathering. The less information presented, the more secure our MTA.

Patch management and software updating

Owing to the RPM architecture of Fedora Core 2, patch management fairly straightforward. Red Hat's inclusion of yum as the default tool for up2date means that yum update will update all components of the MTA registered with RPM. In order to enable yum nightly autoupdating, you need to have an Internet connection and to enable the service via

```
chkconfig yum on
```

To further secure the MTA, host integrity/host-IDS applications can be deployed. Some of the popular ones include Samhain, AIDE, integrit and OSIRIS.

Milters

The future of Sendmail lies in its milter interface. Milter stands for Mail Filter. There are several milters developed for sendmail but they are primarily in three groups.

Milters developed by Snert.com now hosted at Milter.info:
These include milters to check message content, to implement grey-listing and antispamming[21]. Grey-listing is a technique employed by MTAs that temporarily

[19] http://www.eicar.org/anti_virus_test_file.htm
[20] http://spamassassin.apache.org/gtube/
[21] http://www.milter.info/

rejects incoming e-mails from unknown sources. The temporary block rejects the e-mail and informs the MTA of the sender. In the case of spammers, the sender or the sending MTA is a fake so the message is never resent. For legitimate senders, the message is retried at the appropriate time by the sending MTA and is (hopefully) accepted the second time around. There is also a milter for a reputation service that is designed to work with emerging standards such as SPF, SenderID and DomainKeys. Additional draft IETF proposals include SRS, PRA, MailFrom and MARID, all being driven by open source.

URLS to the Future of E-mail

http://community.roxen.com/developers/idocs/drafts/draft-ietf-marid-submitter-03.html

http://community.roxen.com/developers/idocs/drafts/draft-ietf-marid-protocol-03.html

http://community.roxen.com/developers/idocs/drafts/draft-ietf-marid-core-03.html

http://community.roxen.com/developers/idocs/drafts/draft-ietf-marid-pra-00.html

http://community.roxen.com/developers/idocs/drafts/draft-hallambaker-accredit-00.html

http://community.roxen.com/developers/idocs/drafts/draft-irtf-asrg-iar-howe-siq-00.html

http://community.roxen.com/developers/idocs/drafts/draft-delany-domainkeys-base-01.html

Conclusion

Much of e-mail's future is to be decided in very soon. This could very well be in the final months of this year (2004) or in the years to follow.
Just as the development of an e-mail standard was the result of continuous debate with consensus in compromise[22], the MARID proposal for sender authorization looks to be no different. With the rejection of the use of PRA in the SenderID protocol, it remains to be seen whether an alternative will become the accepted standard or if the de facto gambit will result in victory for the vendors involved. When the dust settles, the result should be more secure e-mail and a solution to end spam for the foreseeable future.

[22] http://www.olografix.org/gubi/estate/libri/wizards/email.html

References

1. Hall, Steve. “A Brief History of Spam (Lunch Meat, Monty Python & Junk Email)” 3 Jun 2003 URL: http://www.marketingvox.com/archives/2003/06/03/a_brief_history_of_spam_lunch_meat_monty_python_junk_email/

2. Merriam-Webster Inc. Merriam-Webster Online Dictionary 2004. URL: http://www.webster.com/cgi-bin/dictionary?book=Dictionary&va=spam&x=12&y=11

3. Templeton, Brad. “Origin of the term "spam" to mean net abuse” URL: http://www.templetons.com/brad/spamterm.html

4. Postel, Jon.” On the Junk Mail Problem” NIC #33861. November 1975. URL: http://community.roxen.com/developers/idocs/rfc/rfc706.html

5. Message Labs 2004 Ltd. "Email Threats” URL: http://www.messagelabs.com/emailthreats/default.asp

6. 2004 Ciphertrust, Inc. “How Spammers Fool Spam Filters” Email Security Resources. URL: http://www.ciphertrust.com/resources/articles/articles/foolspam.html

7. Thorson, Bill. “How spammers bypass e-mail security” EE Times. July 19, 2004 (9:00 AM EDT) URL: http://www.eet.com/in_focus/communications/showArticle.jhtml?articleID=23900564

8. Anti-Phishing Work Group. “Unique Phishing Attack Trends Apr 2004 – June 2004”. What is Phishing? URL: http://www.antiphishing.org/images/chart_08-04-04.gif

9. Anti-Phishing Work Group. http://www.antiphishing.org

10. Wade, Karsten.” Fedora Core 2 SELinux FAQ” URL: http://fedora.redhat.com/docs/selinux-faq-fc2/

11. Field, Julian. “What’s New in Version 4”. MailScanner: A User Guide And Training Manual. 1st September 2004. URL: http://www.sng.ecs.soton.ac.uk/mailscanner/newinv4.shtml

12. Field, Julian. Mailscanner “News”. Version 4.33. 1st September 2004. URL: http://www.sng.ecs.soton.ac.uk/mailscanner/#news

13. SpamAssasin™. “Welcome to the Apache SpamAssassin Project website!” Latest (non-ASF) release: 2.64. URL: http://spamassassin.apache.org/

14. Open Source Initiative. Open Source TM. “Licenses” URL: http://opensource.org/licenses/

15. Clam Anti-virus. “Abstract” URL:http://www.clamav.net/abstract.html#pagestart

16. “Postfix Add-on software” URL:http://www.postfix.org/addon.html#content

17. Hafner, Katie and Lyon, Matthew. "Talking Headers”. Wizards. August 4, 1996. URL:http://www.olografix.org/gubi/estate/libri/wizards/email.html

18. Fortress Systems Ltd. “Fortress SMGateway Architecture Diagram”. 2004. URL:http://www.fsl.com/whitepapers/Fortress_SMGateway_Architecture_Diagram.pdf