A Company in Chapter Eleven Doesnt Have to Eat SPAM

Bob Olson

A Company in Chapter 11 Doesn't Have to Eat Spam

GIAC Security Essentials Certification (GSEC)
Practical Assignment Version1.4b Option 2
Case study in an Open Source Software Replacement System for a Commercial E-Mail Filtering System.

By

Bob Olson

March 22, 2004

Special thanks to my Co-Workers and to the Open Source Community.

Table of Contents

Abstract

This paper is a case study detailing the replacement of a commercial E-mail filtering system with one made up of all Open Source Software. The main goals were to reduce delivery time, increase accuracy of spam and Malware detection and to reduce operating costs.

The new system performed exceptionally well, catching a majority of E-mail borne viruses and spam with a total false positive rate not far off the target of less than one percent. The delivery times for inbound E-mail dropped considerably and were well below tolerable levels. Other than man hours spent researching and configuring the new system, there was no money spent implementing it.

In conclusion, the final implementation worked well and is a suitable E-mail content filtering system that meets the needs of a company such as ours. What more can you ask for when something is fast, flexible and free.

Background

Earlier in the year our company filed for Chapter 11, which among many things caused colossal budgetary hurdles to overcome. The primary mission statement of the IT department became to save money wherever possible. This prompted the review of every system in the enterprise to see where operating cost savings could be made.

As a large retail and catalog company, being able to correspond in a timely fashion with our customers is paramount. E-mail has long since been a convenient and cost effective method for staying in touch with our business partners and customers. So much so that is has almost become a double edged sword. Our company is so dependant on E-mail, that any disruption in the steady flow of E-mail into and out of our enterprise sends ripples of frustration deep into the ranks of the company and the depths of our customer base.

Our ability to process E-mail in a timely fashion has been slowly degrading to a point that incoming E-mail has been delayed for as much as 19 hours at its longest point. This is largely due to the fact that we are scanning every e-mail that comes in for an ever increasing number of E-mail born threats and craftier methods for slipping in messages offering free lunches, low interest mortgages, cheap Viagra prescriptions and the ever so interesting singles ads, with outdated software and a non-optimized hardware configuration.

As part of the data security team, I work hand in hand quite often with our organizations E-mail administrators and have shared in their frustration with the current E-mail system in place. In an effort to save the company money on software and hardware maintenance renewal fees and to improve performance and reliability, I decided to review our current system and propose a new system using existing hardware and open source software (OSS) to replace the old system.

Before - Analysis of Current System

Architecture

The current mail system consists of 4 Windows 2000 servers in the DMZ. Each server acts as a primary mail exchanger for 1 domain and a backup mail exchanger for 3 other domains. The mail exchanger records (MX) for each of the domains are maintained by an external DNS provider. Inbound SMTP traffic destined for these servers is controlled by a firewall. The firewall has two rules concerning these servers. The first is a rule to allow any external source to

communicate via SMTP with any these servers. The second rule allows these four servers to send mail to the internal Lotus Domino servers for final delivery.

Processes

Several discussions with the E-mail administrators provided me with the process in which inbound E-mail is handled.

1. Can the sender's IP address be resolved by DNS?
2. Does the sender's IP address have a reverse DNS record?
3. Is the mail message size less than 10 megabytes?
4. Is the senders address in the banned address list?
5. Is the senders address in the banned hosts list?
6. Is the recipient jobs@domain.com? If it is, reply to sender, forwarding them to http://careers.domain.com.
7. Perform a virus scan on the e-mail message.
8. If the E-mail has an attachment, scan the attachment for Malware.
9. If the attachment is Malware, can it be cleaned? If not, delete the message, if it can, strip the payload and place it in "dirty folder".
10. Insert legal disclaimer at the bottom of the e-mail message.
11. Does the message have a *.bat, *.com, *.scr, or *.pif attachment?
12. Does the message have an executable attachment?
13. Are there items in the E-mail that are in the "E-mail Bomb" reference list?
14. Is there content that is in the "Watch List"?
15. Is there porn related content?

If mail passes the above checks it will be forwarded to the internal Domino servers for further processing. These checks are also the minimum filtering requirements needed for the replacement system.

Problems Identified

The cost for maintaining the current system is in excess of $45,000.00 per year for hardware and software maintenance costs. This does not take into account the number of man hours being spent maintaining the current system.

The E-mail servers in the DMZ are not clustered or load balanced which allows for some servers to be underutilized. This is not an efficient use of hardware and causes some servers to sit idle while others are overloaded.

The virus scanning software being used does not run as a daemon. This causes a separate process to be spawned whenever a message is scanned for Malware and robs the system of valuable resources.

There is no intrusion detection measures installed on the servers. System logs are not sent to a central log server and auditing has not been configured on the servers.

The recipient of the incoming mail is not checked to see if it is a valid user on the system. If a mechanism where in place for the recipient to be checked when mail arrives these messages would not have to be processed further.

Recipients are not notified of messages that have been quarantined. It may be days before the end user is notified that they have mail in quarantine which usually causes the original sender to re-send the lost message.

Only the E-mail administrators are able to release messages for the end users. This creates extra work for the administrators. If the end users were given the ability to view, release and or delete messages in quarantine, it would allow the administrators to perform other tasks.

There is no easy method for releasing e-mail messages that have been quarantined. Rather than just selecting the quarantined message in a graphical user interface, the administrators have to log onto the server and manually release the message.

Approach Taken

Collect Business Requirements

The following requirements were deemed necessary by the business for any system that replaces the current system and are listed in order of priority:

- Ability to support at a minimum, the current E-mail filtering rules.
- Processing rate for E-mail should be at or above 3,000 messages per hour.
- Rule sets should be easy to configure and be able to be applied to groups where applicable.
- System should have a GUI front end or Browser based interface for administration and management of quarantined messages.
- The false positive rates for Spam should be <= 1%.
- The System needs to support LDAP queries to the internal Domino Servers.
- The ability to generate detailed reports of E-mail activity must be present in the replacement system.
- The virus signatures of the anti-virus software must be able to be automatically updated.
- Must fit into budget constraints.

Software Selection

One of the design goals of the new system is cost reduction. With this in mind all software reviewed and ultimately selected is Open Source Software.

For the new E-mail relay servers I chose Redhat 9 Linux[1] for the operating system. Not because of it's superiority over other Linux distributions but for it's familiarity to the non-UNIX admin. Ultimately this system will have to be maintained by administrators that don't have an extensive UNIX background so a distribution with a more mainstream presence was decided on here.

For the message transfer agent (MTA), I chose Postfix[2] as the MTA rather than the well known Sendmail . Postfix is an alternative to Sendmail that I have found to be much more user friendly and according to the author of the program, "very fast and secure".

In conjunction with Postfix's internal anti-spam features, SpamAssassin[3] will be used to check inbound messages for spam. SpamAssassin uses a rule-base to check the headers and text of incoming messages for spam and can tag them as such for further processing.

Virus checking will be performed with CLAM Antivirus[4]. It is also released on the GPL license and has a virus database of over 20,000 viruses, worms and Trojans which is updated regularly.

Between the MTA and content checking software is Amavisd-New[5] , a software package that will act as the middleman between the incoming and outgoing mail calling the appropriate scanning software as needed.

Maia Mailguard[6] is the front-end system that is needed for administrators and end-users alike. This software was chosen mainly for its feature that allows the end-user to manage their own quarantined E-mail as well as the feature that allows administrators to delegate administrative privileges to specific users.

During - Design, Build and Test Results

The new system will consist of three servers. Two servers will be the E-mail Relay servers and the third will be for the Web front end and database. Both mail servers will be configured to accept mail for all of our domains and will be load balanced via the firewall. All three of the servers will reside in the DMZ. (Figure. 1)

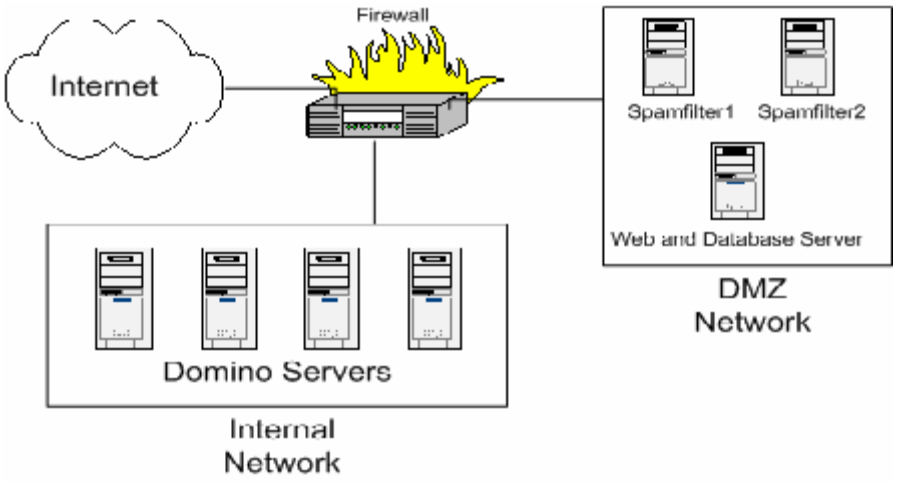

Figure 1

The firewall will be configured so that it will intercept all inbound SMTP traffic and check to see if the recipient's domain is one of ours. Mail received for domains other than ours will be automatically dropped; otherwise it is forwarded to either of the new mail relays in the DMZ. Having the firewall handle all communications with external mail servers provides three benefits. The mail servers will not communicate directly with the outside world, they will not have to process mail that is not destined for our internal domains decreasing system utilization and in the event that both E-mail relay servers are unavailable the firewall will queue the incoming e-mail until the servers are brought back online.

When it came time to build the systems I followed the guidelines as outlined in Lance Spitzner's whitepaper entitled "Armoring Linux: Preparing Your Linux Box for the Internet". Lance Spitzner is the founder of the Honeynet Project, co-author of "Know Your Enemy", author of "Honeypots: Tracking Hackers" and also author of several other whitepapers. He is considered a respected and trusted resource for security related information. In his whitepaper "Armoring Linux" he states;

> "The best place to start in armoring your system is at the beginning, OS installation. Since this is a production system, you cannot trust any previous installations. You want to start with a clean installation, where you can guarantee the system integrity. Place your system in an isolated network. At no time do you want to connect this box to neither an active network nor the Internet, exposing the system to a possible compromise. I personally witnessed a system hacked by a script kiddie within 15 minutes of connecting to the Internet."[7]

The first system I built was one that would be used for building all subsequent servers and would not be part of the production E-mail filtering system. This server resided on our internal network and had a separate connection to a private network where network installs would be performed. Procedures for setting up a Redhat Network Install server can be found in a number of places, I

used the instructions from the "Network-Install-Howto"[8] written by Graham White. Graham White is a graduate of The University of Exeter with a B.Sc. (Honors) degree in Computer Science and is a Red Hat Certified Engineer.

In preparation for creating the custom software packages to be installed on the production servers I had to download all the needed updates and source files for compiling the software. A possible caveat to downloading software from the internet is that it is difficult to ensure that you are downloading it from a valid source. I verified all downloaded software using GPG. GPG, the counterpart to PGP, is a tool for secure communication and software storage. More information on GPG can be found in the GPG How-to[9]. After all the software had been verified I proceeded to create the custom packages that I would need for the two different server configurations. Information on customizing the Redhat Kickstart installation process can be found in the "Redhat Kickstart How-to[10]" written and maintained by Martin Hamilton.

Now that I had the network install server ready to accept clients, I connected the servers for the new E-mail system to the private network reserved for network installs and started the install process for each server. For the two E-mail servers, I used a Kickstart config file created specifically for them and another Kickstart configuration file for the Web and Database server that installed packages specific to it. Once the servers were built, there was still more configuration that needed to be done to individualize the servers and to further secure the operating system, network and installed software. The following is a list of post-install steps performed that were not scripted during the install. Some of them were tips taken from the book, "Linux Security Cookbook"[11]

1. I setup Tripwire on each system – Tripwire is a system integrity checker that periodically inspects important system files for modifications. Once the snapshot of the system was taken I copied it to our internal security server for safekeeping.
2. I turned on source address verification in the Kernel that would drop packets that appear to come from our internal network but don't. This is important as our internal users will be accessing the Web server component of this system directly from our internal network.
3. I turned off the ability for Root to login in directly. If any maintenance needs to be done by administrators, they will use sudo to use privileged commands.
4. I set the hostname for each server to be unique and assigned the proper network configuration information that our telecom team provided me such as IP address, netmask and default gateway.
5. I had to edit the postfix configuration files to reflect the new hostname. I also had to change the server name configuration parameter in the httpd.conf file for the Apache web server to reflect its new hostname.
6. I went through the output of a 'chkconfig –list' command to verify what services were configured to start automatically and disabled all

unnecessary services with the command 'chkconfig *servicename* off' for each unnecessary service.

7. I had to configure our internal logging server to allow the three new servers to send their logs to. Having log files in a central location is not just for convenience, it is also to ensure log file integrity in case the servers in the DMZ were to be compromised and the log files altered.

Once the servers were ready, I requested, via our change management system, permission to move these servers to the DMZ. After the change request was approved, the servers were placed in the DMZ and tested for connectivity. The only issue that I ran into was the LDAP authentication process. It was a combination of firewall rules, or the absence of them, that would allow LDAP traffic between the DMZ and our internal network and a LDAP schema discrepancy that was resolved by our Mail Administrators.

Stage one of the testing process was now ready to begin. First I sent test e-mails from each of the servers to internal accounts and they were delivered fine. The CLAMAV antivirus software package comes with test virus signatures which I used for testing the antivirus daemon. Each time I sent either an infected file or embedded signature CLAMAV caught it successfully and quarantined the file and sent a notification message to virusadmin@domain.com. Spamassassin also has test files for testing and test file sent was identified as spam correctly. When I sent an e-mail to an unknown user such as asdf@domain.com the postfix daemon reported a '550' User unknown error in accordance with Section 3.1 in RFC 821[12].

Stage two consisted of testing the Web interface for administration and end-user quarantine management. I was able to log into the Maia Mailguard system as the super user and verified that the domains were setup correctly. I then logged out and logged in as a standard user. This allowed me to see my current quarantined E-mail and all mail that was possible spam. I had the ability to release quarantined e-mail has ham or delete it and mark it as spam. I could also create my own black and white lists for users or domains.

Stage three testing consisted of testing the server's ability to handle large amounts of e-mail. To perform this test, I used utilities from the postfix distribution, post-sink and post-source. One of the requirements of the new system was to be able to handle 3,000 messages per hour. I ran tests with 100, 500, 1000, 3000 and 10,000 messages against one server at a time. All messages were being checked for spam and being scanned for viruses. On the first server the times for processing the e-mails were very impressive, to process 3,000 messages it only took 33 minutes. It only took 110 minutes to process 10,000 messages. The results were somewhat slower on the second machine after checking the server; I found that the drive installed in that server was a 5400 RPM instead of a 7200 RPM drive like the first server. Also, there were messages in the system logs indicating that the network interface card was

possibly having duplex mismatches. I ended up changing out the drive with another 7200 RPM and our telecom group changed the speed and duplex on the port the server was using to 100/Full. After the changes were made, I re-ran the tests and this time the results were comparable only varying by a few minutes.

Stage four of the testing process consisted of having the Mail Administrators verify that the necessary mail filtering rules were in place and educating them on the use and configuration of the system. Once they were satisfied with the mail rules and were comfortable with editing the system configuration files it was time to test the system in production.

All that was needed to go live with the new system was to add the appropriate firewall objects and rules in the firewall and install the policy. A nice benefit of using the firewall to intercept inbound SMTP traffic was that we didn't have to change the MX records for any of our domains so the cutover was instantaneous. This also meant that in the event of catastrophic failure or irresolvable problems we could fall back to the original system in a matter of seconds. Once the cutover was made both mail servers came alive with activity and we kept a watchful eye on the system ensuring there were no issues. After three hours of monitoring it was decided that the system was stable enough to leave in place. We all breathed a sigh of relief and I kept my fingers crossed.

Summary - Conclusion

Over a weeks time the system was carefully monitored and metrics were being collected. Delivery time for incoming E-mail improved considerably. It would normally take at least an hour for a message sent from my home account to reach my office account. Now it was consistently under four minutes. We were averaging 115,000 E-mails per day for all domains with a little over two percent false positive rate. We attributed the high false positive rate to not enough end user education. Within Maia Mailguard if mail marked as spam is released, it affects the false positive rate and there were a number of users that were uncertain on what they should do with their quarantined messages. We did notice a small delay while the postfix mail daemon performed the local user lookup but nothing to be concerned about. The system resource utilization on the three systems were nominal with plenty of room to spare. The firewall showed an increase in CPU utilization from the extra task of intercepting and forwarding SMTP traffic but it too was negligible.

The response from the user community was mixed. Some where ecstatic that they finally had the ability to create their own spam rules and got weekly reminders about messages that had been quarantined. Others were frustrated that they were not given the proper training on the new procedures. The business, from initial reports, was very happy. The cost savings per year off the bottom line could attribute to that.

Even though the decision hasn't been made to stay with this new system, I am pleased with the outcome thus far. I can safely say that a system such as this can definitely serve a company such as ours.

References

[1] Redhat, Inc. “Redhat – Linux” URL: http://www.redhat.com (Jan 25, 2004)

[2] Venema, Weitse. “Postfix Overview” URL: http://www.postfix.org/motivation.html (Jan 25, 2004)

[3] SpamAssassin™ “SpamAssassin – Welcome to Spamassassin” URL: http://www.spamassassin.org/full/2.6x/dist/INSTALL

[4] ClamAV. “ Clamdoc” URL: http://www.clamav.net/doc/0.65

[5] Unkown “ Amavisd-new” URL: http://www.ijs.si/software/amavisd/#doc

[6] LeBlanc, Robert “Maia Mailguard Homepage” URL: http://www.renaissoft.com/projects/maia/index.php

[7] Spitzner, Lance. “Armoring Linux: Preparing Your Linux Box for the Internet” URL: http://rootprompt.org/article.php3?article=376 (Feb 18, 2004)

[8] White Graham, “Network Install HOWTO” URL: http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/pdf/Network-Install-HOWTO.pdf

[9] De Winter, Brenno. “Gnu Privacy Guard (English)” URL: http://webber.dewinter.com/gnupg_howto/english/GPGMiniHowto.html

[10] Hamilton, Martin “Redhat Linux Kickstart HOWTO” URL: http://www.ibiblio.org/pub/Linux/docs/HOWTO/other-formats/pdf/KickStart-HOWTO.pdf

[11] Barret, Daniel; Byrnes, Robert and Silverman, Richard. Linux Security Cookbook. Reading: O’Reilly, 2003

[12] Postel, Jonathan B., “SIMPLE MAIL TRANSFER PROTOCOL” URL: http://www.ietf.org/rfc/rfc0821.txt

Anonymous, Maximum Linux Security, Sams Publishing, 2000