A New Architecture for Enhancing the Security of E-mail System

Jangbok Kim
Graduate School of Information and Communication, Ajou University, Suwon Korea

Hosung Lim
Graduate School of Information and Communication, Ajou University, Suwon Korea

Kyunghee Choi
Graduate School of Information and Communication, Ajou University, Suwon Korea

Gihyun Jung
Division of Electronics Engineering, Ajou University, Suwon Korea

Abstract - In this paper, we propose a bridge-type e-mail proxy architecture to release the bottlenecks of the two popular mail security architectures: software mail filter and e-mail gateway. In the proposed architecture, a bridge-type e-mail proxy is located in front of mail server and sniffs all flowing packets on the network. If sniffed packets belong to one of mail protocols, the proposed system filters the packets just as an e-mail proxy system does. But if a sniffed packet does not belong to mail packets, it is forwarded just as a normal network bridge does. The proposed architecture has several merits.

Keyword: e-mail, proxy, spam, bridge

I. Introduction

Now days, it is nearly impossible to work in office without e-mail. But the convenient and popular e-mail is also utilized as a method to impose severe damage to its clients. The possibility of damage introduced through e-mail increases along with the increase in population of e-mail users. The dark side of e-mail system may be summarized into five categories. First, e-mail is utilized as a path to spread virus. Second, spam mails flowing into a system through e-mail waste the system resource. Third, many recent hacking techniques utilize e-mail. Fourth, important information may be leaked easily through e-mail. Lastly, e-mail text itself may be sniffed by other unauthorized clients and used for an unwanted purpose. [1][2][3]

Many ways have been proposed to prevent e-mail systems and other resources from the treats. One of the most popular ways is to install a mail filtering software into an e-mail server. For instance, it is possible to filter all e-mails delivered to clients by running a mail filtering software between a MTA and a MDA. The main advantage is that installing mail filtering software does not require any significant changes of software and mail server. But the drawback is that the filtering system has to change depending on hardware platform, operating system and MTA at which it is installed. Another

drawback may come from the performance degradation of mail server. If a mail server does not have enough processing power, it is nearly impossible to efficiently process a heavy-loaded mail filtering software. Consequently, the server may not process all arriving e-mails in time and delay e-mail delivery.

Another popular way to protect e-mail system and to overcome the way of installing filtering software is to use an e-mail gateway. By the email gateway, we mean an application-oriented system that is responsible for connecting incompatible e-mail systems or for filtering e-mails. When an email gateway receives an e-mail, it filters the e-mail and forwards it to its actual destination mail server. Since the gateway is installed independent of operating system and hardware platform of mail server and MTA, it is not necessary to consider any details related to the existing mail server in implementing gateway. That is one of main merits of the gateway architecture. But, it is not possible to utilize the filtering capability of gateway for mails directly delivered between mail servers on an intra-network. The feature may bring a security hole of e-mail system. Another drawback of gateway architecture is that it takes a long time to install and remove an e-mail gateway from network. Since modifying the MX value of domain name service system is required for installing an e-mail gateway, installing or removing a gateway is finished after caching data in the domain name server is completely updated. Usually the caching update needs a severe amount of time.[4][5][6]

In this paper, we propose a bridge-type e-mail proxy architecture to release the bottlenecks of the above two popular mail security architectures. In the architecture, a bridge-type e-mail proxy is located in front of mail server just like a personal firewall and sniffs all flowing packets on the network. If a sniffed packet belongs to one of mail protocols, the filtering system receives and filters the packet just as an e-mail proxy system does. But if a sniffed packet does not belong to mail protocols, it is bridged just as a normal network bridge does. The proposed architecture has the following merits.

1) It is independent of hardware platform, operating system and MTA of mail server.
2) Installing/removing is easy since any modification of network configuration like the MX value of domain name server is not needed.
3) All mails flowing from or to mail servers on network can use the filtering capability provided by a bridge-type e-mail gateway.

In section two, the proposed architecture is described in detail. Some implementation issues are discussed in section three. And its performance is evaluated in section four. The last section wraps up this paper.

II. Bridge-type E-mail Proxy Architecture

Figure 1 shows a configuration of network where the proposed bridge-type e-mail proxy is located in front of an e-mail server (the proxy may be located at any places in network). The proxy sees all packets flowing on the network where the proxy is located (*intranet* in Fig 1). It pulls packets belonging to one of mail protocols (for example, SMTP or POP3) into

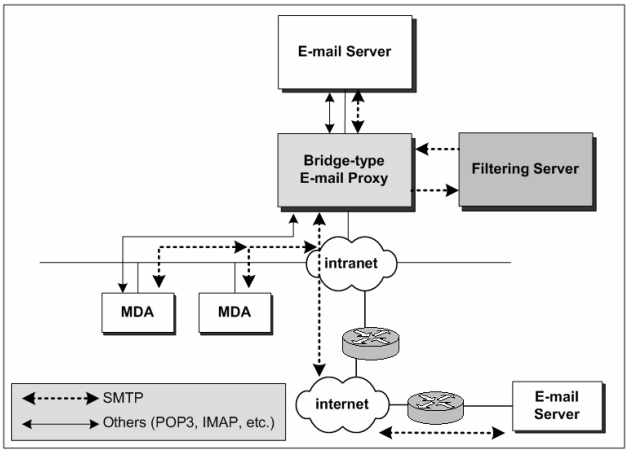

Fig 1. The architecture of bridge-type e-mail proxy system

its e-mail filtering module. Other protocol packets are just forwarded and nothing is done with the

proxy. Once mail packets are filtered by the e-mail proxy, the packets are delivered to either the e-mail server located after the proxy or their final destination external mail servers.

The e-mail filtering module (depicted as "Filtering Server" in the figure) could be either separated from the e-mail proxy or included in the proxy. The reason to separate the e-mail filtering module from the proxy is that usually many practical e-mail filtering functions are very time-consuming operations. In a small network operated with a relatively few clients, a system combining the proxy and the filtering function may be enough to deliver all flowing e-mails without severe delivery delay. But if a proxy including the filtering module is hired in a large network, the proxy may not properly deliver all incoming mails (even though the performance is dependent on the system it is implemented). In the case, it is better to separate the filtering module from the proxy. Once an e-mail is filtered by the Filtering Server, it is sent out to the destination utilizing the proxy.

The proposed architecture has several merits. Since it is implemented in a separate system, it does not depend on the hardware platform and operating system of e-mail server system. Since it intercepts all e-mail related packets from network, it can process all e-mails flowing on network, regardless of their sources and destinations. Another merits come from its transparency of configuration. Both the e-mail server attached to a proxy and the clients in the internet do not recognize the existence of proxy. Thus installing and removing a proposed proxy is very easy. A cumbersome task for modifying the MX field of DNS and e-mail server reconfiguration is absolutely not necessary.

III. Implementation

We have implemented the proposed architecture on two hardware platforms: a PC with Pentium III 1GHZ processor and 128M SDRAM, and an embedded system with a MIPS processor. Linux version 2.4.20 is patched for both hardware platforms to accommodate the e-mail proxy. And the mail filtering server (it does "mail filtering") is implemented separately.

Figure 2 shows the flow of control for the case where an external e-mail server sends an e-mail to the internal e-mail server through the implemented bridge-type e-mail proxy. The packet flow between the e-mail proxy and the filtering server is also shown.

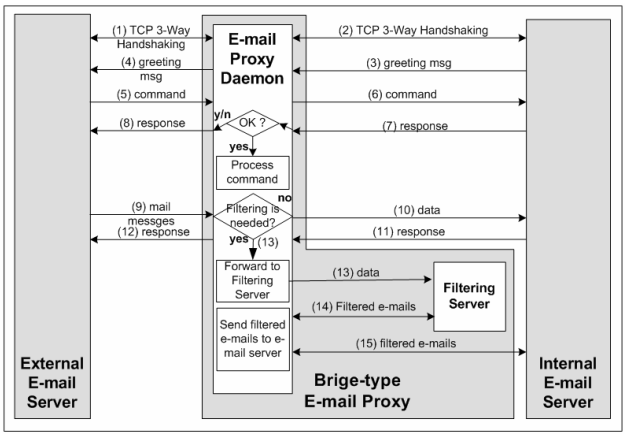

Fig 2. Packet flow in the e-mail proxy

The proposed e-mail proxy establishes a TCP connection with an external e-mail server on behalf of the internal e-mail server. Once the connection between the proxy and the external server is established, the proxy also establishes a TCP connection with the internal mail server, which is the server the external mail server wants to make a connection. The e-mail proxy transfers the SMTP commands (in this implementation the proxy only filters SMTP packets) from the external server to the internal server and then receives the reply from the internal server. When the reply is invalid, saying that the internal e-mail server doesn't allow the command. But if the reply is valid, the proxy either changes its state or stores the information and then replies it to the external mail server. For example, when the proxy transfers the mail sender contained in the 'MAIL' command received from the external server to the internal server and receives the reply from the internal server, saying that the sender is valid, the proxy has to keep the sender information as well as to transfer the information to the external

server. The SMTP protocol commands we utilize include HELO, EHLO, HELP, MAIL, RCPT, DATA, RSET, NOOP, VRFY, and AUTH. [7][8]

The information gathered during the two connections is utilized to decide whether the mail messages followed by a 'DATA' command should be filtered or not. For example, when the domain of sender and receiver is same or the sender is registered as a well-known safe one, the mail message needs not the filtering. If the proxy decides an incoming message is safe, then the message is directly transferred to the internal mail server. But if the proxy decides that the message has to be filtered, the connection between the proxy and the internal e-mail server is disconnected. And the message is transferred to the filtering server. The filtering server filters the message and then transfers the filtered message to the internal mail server through the proxy.

To keep the transparency of proxy, IP and source MAC have to be spoofed. For the spoofing, the destination IP of incoming packet is changed to that of proxy server IP and the server IP is properly modified. The MAC address is also modified using the ARP. Figure 3 shows the flow of spoofing

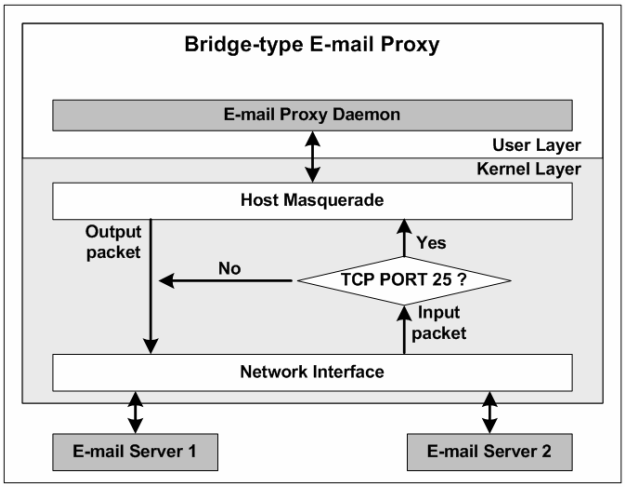

Fig 3 Packet handling for proxying e-mail

IV. Performance Evaluation

A main drawback of proxy architecture is that a proxy may degrade the performance of network. The performance we mean includes network throughput and network utilization. In the sense, we evaluate the performance of the proposed and implemented e-mail proxy in two categories. Firstly, the network bandwidth utilization is measured both when the e-mail proxy is installed and when not installed. Secondly, the amount of mails that the proxy can handle.

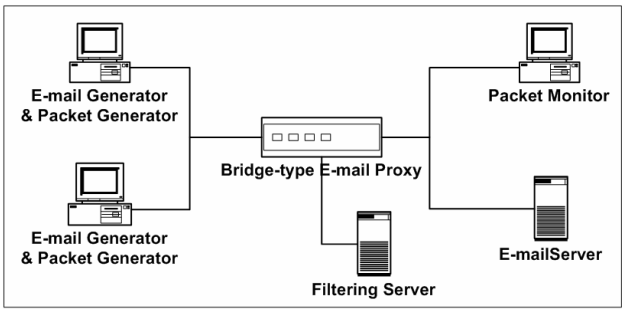

Fig 4 E-mail proxy performance test bed

The evaluation is performed in the test network shown in Fig 4. For the e-mail proxy server, a PC with PIII 650 MHz and 128 SDRAM is utilized. The filtering server was implemented in a PC with PIII 1GHz and 256 SDRAM. Two e-mail generators continuously generate e-mails for five minutes. The average size of e-mails is 41,366 bytes, which is the same as that of e-mails monitored for a week in Ajou University. The E-mail server in Fig 4 does nothing except receiving e-mails. The evaluation was repeated ten times and the figures in table 1 and 2 are their averages.

Table 1. Numbers of delivered e-mails for five minutes

Proxy installed	No. of delivered e-mails
No	30,700
Yes	13,200

Table 1 shows the number of e-mails treated for five minutes when the e-mail proxy is installed as shown in the test bed and removed from the configuration. As shown in the table, the number of e-mails that the e-mail proxy treats is much smaller than that by the mail server without the e-mail proxy. But considering

Table 2. Comparison of bandwidth utilization

Packet Size	Bridge		E-mail Proxy	
	Utilization(Mbit)	Packets/s	Utilization(Mbit)	Packets/s
64 Byte	52	78,350	51	77,000
128 Byte	81	68,400	80	68,000
256 Byte	99	45,600	99	45,600
512 Byte	99	236,00	99	236,00
1024 Byte	99	12,000	99	12,000
1518 Byte	99	8,148	99	8,148

the number of e-mail that the famous *Sendmail* [9] running on Linux can treat in the above test bed is approximately 1,000 for five minutes (we actually measured the throughput but did not present in the table), the throughput of the proposed e-mail proxy indicates that the proxy may achieve e-mail filtering ability, without reducing e-mail system throughput.

The bandwidth utilizations of a bridge and the proposed e-mail proxy are compared and presented in Table 2. The bridge was implemented on a PC that has the same hardware specification and operating system as the one used for the e-mail proxy. Two packet generators (previously used as e-mail generators) generate various different size packets and the bridge and e-mail proxy transfer the packets in their processing manners. As shown in the table, any significant reduction in throughput by the proposed e-proxy is not observed for all cases. [10]

V. Conclusion

We have proposed a new architecture for e-mail filtering system. The proposed bridge-type e-mail proxy screens all packets flowing on network. The packets belonging to e-mail protocols are processed with e-mail filtering process but other protocol packets are just forwarded. The e-mail proxy architecture has several merits over the existing mail filtering techniques. We have implemented the proposed e-mail proxy on an embedded system and a PC. The empirical study done on the implemented e-mail proxies showed that the throughput and bandwidth reduction by the e-mail proxy are not serious at all.

Acknowledgement

This work is supported by BK21 sponsored by MOE and NRL by KISTEP of Korea.

References

[1] Jay Chaudhry. “The e-mail battlefield: build a defense,” May, 2002 . http://www.ciphertrust.com/CTnews_download/

[2] Hal Beghel, Email-The Good, “The Bad, and the Ugly,” Communication of ACM, Vol.40, No.4, April 1997.

[3] Lorrie Faith Cranor and Brian A LaMacchia, “SPAM!,” Communication of ACM, Vol. 41. No.8, August 1998.

[4] W. Richard Stevens, TCP/IP Illustrated, Volume 1, Addison-Wesley, 1994.

[5] Bryan Costales with Eric Allman, Sendmail 2nd edition, O’ Reilly, November 1997.

[6] RedHat, “Email Program Classifications, RedHat Linux 7.3: The Official Red Hat Linux Reference Guide, “: http://www.redhat.com/docs/manuals/linux/RHL-7.3-Manual/ref-guide/s1-email-types.html

[7] Jonathan B. Postel, “Simple Mail Transfer Protocol,” RFC 821, August 1982.

[8] J. Klensin, N. Freed, M. Rose, E. Stefferud, D.

Crocker, “SMTP Service Extension,” RFC 1869, November 1995
[9] Sendmail: http://sendmail.org
[10] Cecui.com, “SecuiWall Performance Chart,” http://www.secui.com