Design and Implementation of A Content Filtering Firewall

Rongbo Du, Rei Safavi-Naini and Willy Susilo
School of Information Technology and Computer Science
University of Wollongong
Wollongong 2522
AUSTRALIA

`Email: {rd12, rei, wsusilo}@uow.edu.au`

Abstract

A firewall is a system for enforcing access control policy between two networks and is one of the most important measures to protect against network attacks. Firewalls traditionally protect the internal network from outside threats. But there has been increasing need for preventing the misuses of the network by the internal users which most previous firewalls overlook. In this paper, we propose a method of adding content filtering functionality to the firewall and describe its implementation. We also show a new attack that combines JAVA Applet and XML to get around the content filtering firewall, hence showing the need for clear usage policy for software and systems.

Keywords:

Network security, Information Technology, Legal and Ethical Aspects, Firewalls, Content Filtering

1 Introduction

1.1 The Need for Firewall

It is well known that the Internet is far from secure. There are hundreds of millions of people connected to the Internet. Some might maliciously try to penetrate into private networks to view or modify data, or even to damage the computer systems. For large organisations, these attacks might not only lead to financial losses, but could also harm the reputation of the organisations and hence reduce the clients' confidence. A firewall is an important tool to reduce these threats. A firewall sits between an external network and an internal network and serves as a guard for the internal network. A carefully designed and deployed firewall can deny unauthenticated requests or requests with potential threats, while permitting authenticated requests, thus protecting the internal network. Firewalls can also guard one part of a private network against its other parts. This is useful especially for organisations with large private networks and many divisions around the world.

1.2 Previous Works on Firewall and Limitations

There are three types of firewalls: Packet-Filtering, Application-Level and Circuit-Level firewall [1].

Packet-Filtering firewall works by dropping packets based on their source and/or destination address or ports [1]. To provide tougher security, Packet-Filtering firewall usually need to be used in conjunction with other firewall components.

Another commonly used firewall is Application-Level firewall in which special-purpose code is used for each desired application. Application-Level firewall makes it easy to control all incoming and outgoing network traffic [1].

The third type of firewall is the Circuit-Level firewall which relays TCP connections. The caller connects to a TCP port on the gateway, which connects to some destination on the other side of the gateway [1].

One problem with most traditional firewalls is that they concentrate on how to protect the private network from external threats without monitoring the behavior of the internal users. Without this monitoring, internal users may abuse the network. For example, during office hours, employees download MP3 or other types of music files, chat with friends, or browse web sites for leisure. Such misuses of network not only cost money and time but also impose unnecessary load on the traffic of the entire network, impeding other users' normal activities. In some cases such as downloading illegal software or music, these might be legal implications for the organisation too.

There are many web content filtering software, such as MIMEsweeper from Baltimore Technologies [2] and Smart Filter from Secure Computing Corporation [3]. However, few of these software are directly built into the firewall. The Check Point FireWall-1 from Check Point Software Technologies Ltd [4] provides content based functionality such as Computer Virus Screening, URL Screening and Java and ActiveX Scanning. However, these software are preserved and not freely available.

We extend the Firewall Package Toolkit FWTK [5] so that by analysing the content of the network traffic passing through the firewall, the firewall can enforce an access policy based on the contents. Most organisations give different access controls for different groups of internal users. For example, universities may require students to use the Internet for academic purposes only, however, academic staff should have less restriction on their accesses. The original Firewall Package Toolkit cannot accomplish this goal because it does not examine the content of the network traffic, thus have no means to detect the misuses of the network.

2 Extending the Firewall Package Toolkit FWTK for Content Filtering Functionality

2.1 Introduction to Firewall Package Toolkit FWTK

The firewall package toolkit FWTK [5] is from Trusted Information Systems, Inc [6]. It is a set of components, which can be used to create a secure firewall system. We chose the toolkit because its source code is publicly available. We also used Squid, a web proxy cache that is an open-source software to replace FWTK's HTTP proxy.

2.2 An Architecture for Content Filtering

We propose that content filtering to be added to FWTK as shown in Figure 1

- For incoming traffic, content filtering is after the firewall access control component, and before permitted traffic is sent to the internal users.
- For outgoing traffic, content filtering is after the firewall access control component, and before permitted traffic is sent to external network.

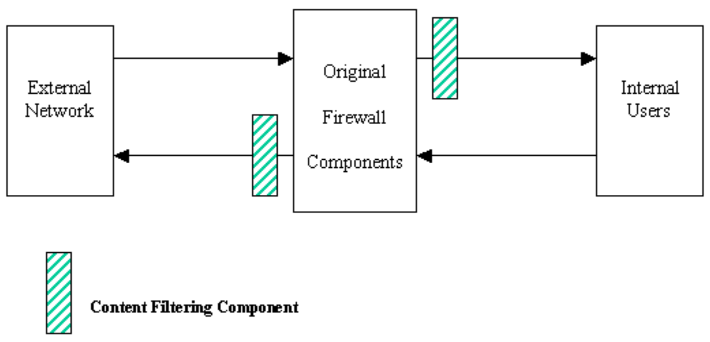


Figure 1: Content Filtering Architecture

The advantage of this architecture is that we do not need to modify the access control components of the original firewall.

2.3 Installation and Configuration of the Firewall

Steps of building the FWTK firewall:

1. Clean up the *bastion host*
A *bastion host* is a computer that is fully exposed to attack [7]. This step includes disabling IP forwarding and all unnecessary services on the host. The purpose of this step is to minimise potential threats to the firewall itself.

2. Build the firewall program
To build the firewall under Linux platform, some adjustments to the source codes are needed.

3. Integrate authentication into firewall
We chose S/KEY [8] as the authentication method. S/KEY is an one-time password system to provides secure authentication over network that are subject to eavesdropping and reply attacks.

4. Configure the firewall and set up network access control rules
Network access control rules are specified in the file `netperm-table` in the directory `/usr/local/etc`.

5. Testing
Network services including HTTP/GOPHER, TELNET, FTP, RLOGIN and FINGER have been tested.

2.4 Content Filtering for Plain Text File

Most files are plain text. Content filtering for these files is based on exact string matching. For example, the pornographic web pages containing a particular word may be detected by the firewall.

We have taken the following steps to implement content filtering for plain text file:

1. Locate the corresponding source files for various network services, such as HTTP, FTP and TELNET.

2. Modify these codes to add content filtering functionality.

3. Reconfigure and restart the firewall.

2.5 Content Filtering for Non-plaintext Files

Some files may be in non-text formats, such as Portable Document Format (PDF) or Microsoft Word (DOC), and some may be in compressed formats, such as zip and gzip. In this section, we discuss content filtering for non-text files.

We have taken the following steps for content filtering for non-plaintext file:

1. When a file passes through the firewall, the firewall will try to determine whether it is a non-plaintext file. This can be done by analysing the header of the file or by calling external programs such as `gunzip`. The firewall does not use the suffix of the file but look into the real content. In this way, the firewall can detect zip files that are renamed

2. If the file is a non-plaintext, then an appropriate program known to the firewall

extracts the text. Otherwise, the firewall will do content filtering directly for this file and transfer it to the destination.

3. The firewall does content filtering on the extended text files.

4. The firewall re-packs the checked files into their received format.

5. The firewall transfers the non-plaintext file to the destination.

3 A Possible Attack on Content Filtering Firewall

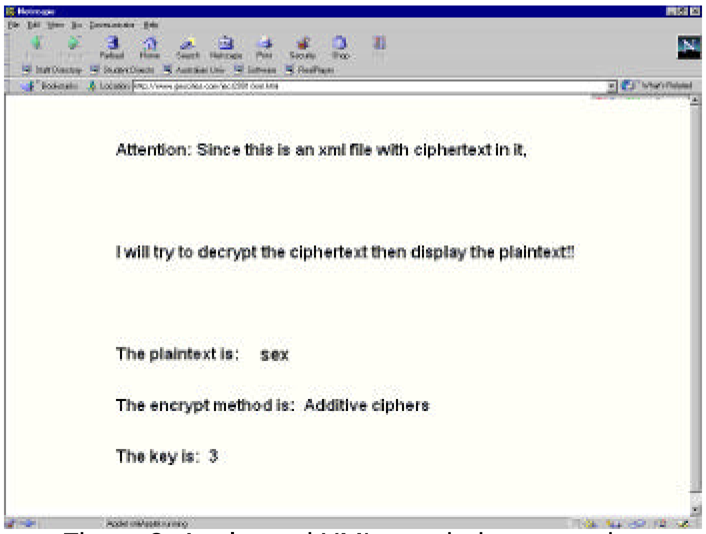
In this section, we describe an attack by combining JAVA Applet and XML to get around the content filtering firewall. This attack shows the difficulty of the content filtering firewall encounters when end-to-end encryption is used.

3.1 Introduction to XML (Extensible Markup Language)

XML is used to describe, store and exchange data. The main difference between XML and HTML is that XML is used to describe data, and to focus on what data is and HTML is used to display data, and to focus on how data would look. In XML, tags are not predefined and must be defined by the user.

3.2 Using Java Applet and XML to Get around the Content Filtering Firewall

Usually Internet browsers such as Internet Explorer or Netscape Communicator use Extensible Stylesheet Language (XSL) to display the content of XML files. If the XML files are encrypted, they can pass through the firewall. The encrypted content can be decrypted by Java Applet and displayed in the Internet browsers. We implement and test this kind of new attack using Java Applet and XML, as shown in Figure 2.


Attention: Since this is an xml file with ciphertext in it,

I will try to decrypt the ciphertext then display the plaintext!!

The plaintext is: sex

The encrypt method is: Additive ciphers

The key is: 3

Figure 2: Applet and XML attack demonstration

To process XML files, we use Java API for XML Processing (JAXP) from Sun Microsystems, Inc.

In Figure 2, the browser is displaying the file `xml.html`, which contains a Java Applet. Once started, this Java Applet first connects to the server and gets back the XML file `ciphertext.xml` then decrypts the text using the key in the same file. The XML file `ciphertext.xml` contains both the encrypted text and key and resides in the same directory as the HTML file `xml.html`.

The XML document ciphertext.xml:

```
<?xml version="1.0"?>
<secret>
<encryptmethod>Additive ciphers</encryptmethod>
<key>3</key>
<ciphertext>pbu</ciphertext>
</secret>
```

From Figure 2, we can see that this attack is successful. The reason that the firewall can not detect this is because the decryption happens in the client's PC, within the internal network. Furthermore, it is possible that the applet can accomplish much more complex tasks such as enabling hyper links inside the XML files pointing to other resources, for example, programs or images.

Although it is not necessary to use XML, combining JAVA and XML makes this attack more extensible.

3.3 Possible Solutions to This Attack

The simplest solution to this attack is to let the firewall block all JAVA applets. To block JAVA applets, we can rewrite the `<applet>` tags in the HTML files to empty tags, so that the Java Applets will not be downloaded or started. This solution totally eliminates this kind of attack, the obvious disadvantage is that users can not use JAVA applet.

A better solution is to block untrusted applets and only accept applets from trusted sites. It is possible that trusted sites contain malicious JAVA applets, however, the chance is much lower in this case.

4 Refining Content Filtering for Different Groups

4.1 The Need of Different Content Filtering for Different Groups

Access control policy is the set of rules that determinates who can use what service or resource, and what privileges users have.

4.2 A Proposal and Implementation

We propose to have different content filtering for different groups using the architecture shown in Figure 3.

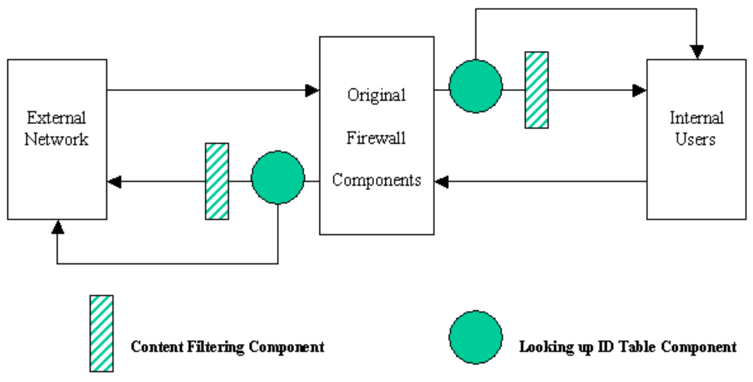

External Network
Original Firewall Components
Internal Users
Content Filtering Component
Looking up ID Table Component

Figure 3: Different Content Filtering for Different Groups

All the users must first login to the firewall proxy. We modify the firewall to record users' login ID in a table when the user logs in, then use the table to determine what content filtering rules to be applied.

For example, when an administrator or staff logs in, the firewall will not apply content filtering to all of his/her network access and downloads. But if a student logs in, content filtering will be applied to the traffic. The rules can be specified in the configuration file, which will be described in the following section.

5 Configuration File for Content Filtering

To allow the end users to specify the access control policy such as what types of format must be checked, we have modified the firewall so that when the firewall is started, it reads the configuration file and sets up the corresponding rules for content filtering.

Currently users can specify the following in the configuration file:

- The string to be checked or replaced.
- The kind of file to be checked.
- The external program to be used for extracting/re-packing of the non-plaintext files.
- Which user group should be checked.

The configuration file has syntax similar to the configuration files of Microsoft Windows, such as `win.ini` or `system32.ini`.

An example of the configuration file is given below:

```
[FILTERS]
sex                   // text to filter
*****                 // text to replace with
[TAR]
tar -xf OLDFILE       // command to extract tar files
tar -cf NEWFILE *     // command to repack tar files
[ZIP]
unzip OLDFILE         // command to extract zip files
zip NEWFILE *         // command to repack zip files
[GROUPS-NOT-TO-CHECK]
admin      // all users in admin group will not be checked
staff      // all users in staff group will not be checked
[GROUPS-TO-CHECK]
student    // all users in student group will be checked
```

This configuration file specifies that the firewall should check for the pornography string and replace it with "*****". It also tells the firewall that tar and zip files should be checked and specifies the external programs that are needed. The `OLDFILE` is the file to be checked and the `NEWFILE` is the file that is checked.

6 Legal and Ethical Issues

When using content filtering functionality, it is important that legal and ethical aspects to be carefully considered. Before the users use the organisation's network, they should be notified about the rules. It is also important not to use the results of content filtering for other purposes.

7 Conclusion

In this paper, we described a proposal and its implementation, for adding content filtering to FWTK. Also, we proposed and implemented a new attack, which combines JAVA Applet and XML to get around the content filtering firewall, and proposed protection methods against this kind of attack.

Future work will be on blocking virus and malicious codes using the firewall.

References

[1] **William R. Cheswick and Steven M. Bellovin**. 1994. *Firewalls and Internet security: repelling the wily hacker.* Addison-Wesley.

[2] **Baltimore Technologies plc.** *Available online at http://www.baltimore.com/*.

[3] **Secure Computing Corporation**. *Available online at http://www.securecomputing.com/*.

[4] **Check Point Software Technologies Ltd.** *Available online at http://www.checkpoint.com/*.

[5] **The FireWall Package ToolKit**. *Available online at http://www.fwtk.org/fwtk/*.

[6] **Trusted Information Systems**. *Available online at http://www.tis.com/*.

[7] **Kurt Dillard**. What is a bastion host?. *Available online at http://www.sans.org/newlook/resources/IDFAQ/bastion.htm*.

[8] **Neil M. Haller and Philip R. Karn and John S. Walden and Scott Chasin**. S/Key. *Available online at ftp://thumper.bellcore.com/pub/nmh*.

[9] **Sun Microsystems, Inc**. Java Technology and XML. *Available online at http://java.sun.com/xml/.*

[10] **Jonathan Hue**. Firewalls. *Available online at http://www.cru.fr/securite/GB/macfirew.txt.*

[11] **International Organisation for Standardisation**. Open System Interconnection. *Available online at http://www.iso.ch/iso/en/ISOOnline.frontpage*.

[12] **Red Hat, Inc**. *Available online at http://www.redhat.com/*.

[13] **Ian S. Graham and Liam Quin**. 1999. *XML Specification Guide.* Wiley Computer Publishing.

[14] **Jess Garms and Daniel Somerfield**. 2001. *Professional Java Security.* Wrox Press Inc.

[15] **Steve Holzner**. 1998. *XML Complete.* McGraw-Hill.

[16] **Dieter Gollmann**. 1999. *Computer security.* Wiley.

[17] **Chris Brenton**. 1998. *Mastering network security.* Network.

[18] **Dan Chang and Dan Harkey**. 1998. *Client/server data access with Java and XML.* Wiley.

[19] **Steven Brodsky**. 2000. *Mastering XMI : Java programming with the XMI toolkit, XML, and UML.* Wiley.

[20] **Michael Floyd**. 2000. *Building Web sites with XML.* Prentice Hall.

[21] **Ann Navarro and Chuck White and Linda Burman**. 1996. *Mastering XML.* Sybex.

[22] **Frank Boumphrey and Jon Duckett and Joe Graf and Paul Houle and Trevor Jenkins and Peter Jones and Adrian Kingsley-Hughes and Kathie Kingsley-Hughes and Craig McQueen and Stephen Mohr**. 1998. *XML Applications.* Wrox Press.

[23] **Karanjit Siyan and Chris Hare**. 1995. *Internet firewalls and network security.* New Riders Pub..

[24] **Tom Myers and Alexander Nakhimovsky** . 1999. *Professional Java XML programming with Servlets and JSP .* Wrox.

[25] **Rolf Oppliger**. 1998. *Internet and Intranet security.* Artech House.

[26] **Terry Bernstein**. 1996. *Internet security for business.* Wiley.

APPENDIX

The Experimental Network:
We implement our work in a private network with three computers running Linux (Red Hat 7.0) operating systems, as shown in Figure 4. Linux was chosen because it provides an excellent programming environment.

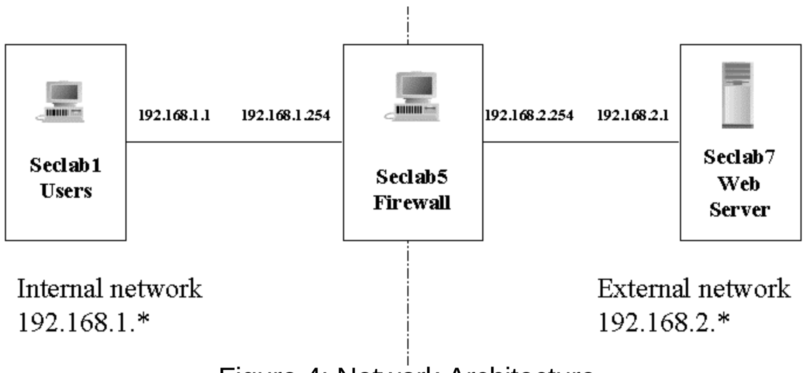
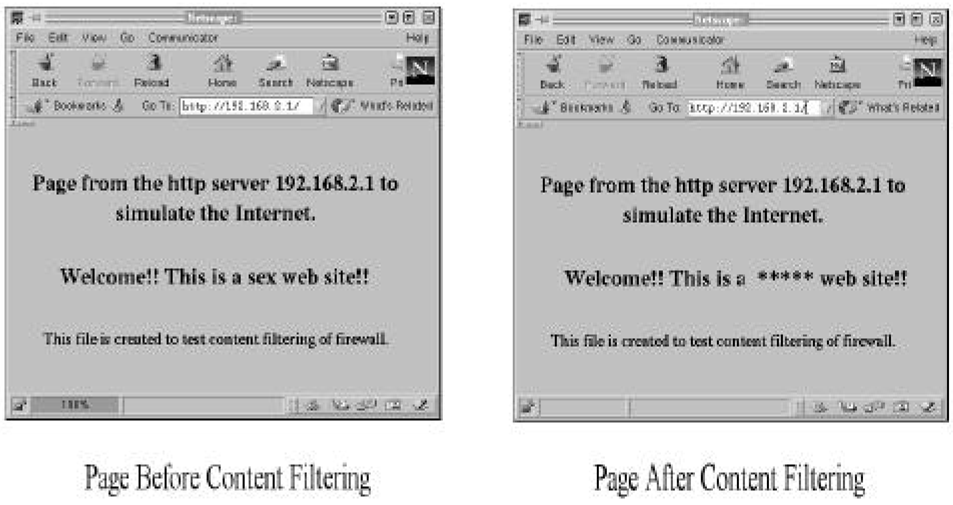


Figure 4: Network Architecture

The computer Seclab5 is the firewall. The computer Seclab1 is used to simulate internal machines and computer Seclab7 serves as the external web server. Seclab1 and Seclab7 are not directly connected but they are both connected to Seclab5. Therefore, all traffic between Seclab1 and Seclab7 must go through the firewall Seclab5 first.

Testing of the New Functionality:
The the new content filtering functionality has been tested for both plain text files and non- plaintext files, as shown in Figure 5 and Figure 6.


File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape
Bookmarks Go To: http://192.168.2.1/ What's Related

Page from the http server 192.168.2.1 to simulate the Internet.

Welcome!! This is a sex web site!!

This file is created to test content filtering of firewall.

Page from the http server 192.168.2.1 to simulate the Internet.

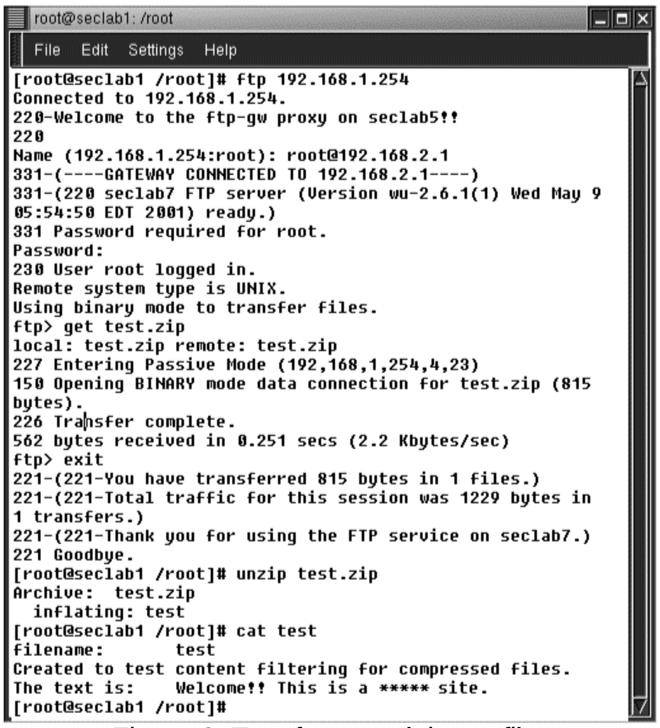
Welcome!! This is a ***** web site!!

This file is created to test content filtering of firewall.

Page Before Content Filtering

Page After Content Filtering

Figure 5: Test for plain text files



```
[root@seclab1 /root]# ftp 192.168.1.254
Connected to 192.168.1.254.
220-Welcome to the ftp-gw proxy on seclab5!!
220
Name (192.168.1.254:root): root@192.168.2.1
331-(----GATEWAY CONNECTED TO 192.168.2.1----)
331-(220 seclab7 FTP server (Version wu-2.6.1(1) Wed May 9
05:54:50 EDT 2001) ready.)
331 Password required for root.
Password:
230 User root logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> get test.zip
local: test.zip remote: test.zip
227 Entering Passive Mode (192,168,1,254,4,23)
150 Opening BINARY mode data connection for test.zip (815
bytes).
226 Transfer complete.
562 bytes received in 0.251 secs (2.2 Kbytes/sec)
ftp> exit
221-(221-You have transferred 815 bytes in 1 files.)
221-(221-Total traffic for this session was 1229 bytes in
1 transfers.)
221-(221-Thank you for using the FTP service on seclab7.)
221 Goodbye.
[root@seclab1 /root]# unzip test.zip
Archive:  test.zip
  inflating: test
[root@seclab1 /root]# cat test
filename:       test
Created to test content filtering for compressed files.
The text is:    Welcome!! This is a ***** site.
[root@seclab1 /root]#
```

Figure 6: Test for non-plaintext files

It can be observed from Figure 5 and Figure 6 that according to the content filtering rules specified in the previous configuration file in section 5, all the plain text files and non-plaintext files passing through the firewall have been checked and the text "sex" has been replaced by text "*****".