

Proxy Server FOR Intranet Security

Dr.Premchand.B.Ambhore¹, Ku.A.D.Wankhade²

¹(Assistant Professor in Information Technology, GCOE, Amravati, Higher and Tech. Education M.S., India

²(Assistant Professor in Information Technology, Govt. College Engg. Amravati (M.S.),

Corresponding Author: Dr.Premchand.B.Ambhore

Abstract : Taking into consideration, the situations and the security policies of organization, we have decided to implement the hybrid Application Layer Firewall (Proxy Server).we need for firewall and network security is very important with a rapid expansion of the Internet, and more and more use of computer networks throughout the world. Firewalls protect the computer systems connected to the private network or local network against attacks from the Internet. Firewalls are considered to be one of the best and most reliable means of network protection against intruders. Most of the today's firewalls are categorized as router-based, circuit-level, and application level. The goal of this project is to implement the application firewall as the application firewalls provides more security flexibility, scalability and maintainability, which is favorable for corporate security environment. We have decided to provide the following functionality in the Hybrid Application Layer firewall. When the application data from the external world is coming to the internal network through the firewall then it has to be scanned for virus at the proxy server itself. If no virus found then only, the data is allowed to pass through the firewall. If virus is found then the data is not allowed to pass inside and the appropriate alerting message is send to the sender of that data/administrator. Proxy server should maintain the log file for storing virus scan information. It contains the entries for the data for which virus is found. In this module, we have used F-Prot, a command line Antivirus for Linux operating system. Authentication for user is asked whenever a request comes from internal users to the proxy server. Proxy server should allow only the authorized users and should deny the access for unauthorized users. The information of the authorized users is maintained in the configuration file in encoded format(encrypted format). When the external client is requesting for the data that is stored as secure data by the real server then authentication is asked for the user. Only authorized users are allowed to access the secured data. The information of the requests, which are denied by the proxy server, is maintained in the log file. This log file contains the information about the sender and receiver of the request, the file name for which the request has come and the date and time to which the request arrived to the proxy server. Internal users are allowed to access the internal data after they authenticate themselves. Proxy server should provide the facility of cache. Whenever a request comes to proxy server, it first searches the requested file in its cache. If not found in cache, then proxy server gets it from the real application servers located inside. If found in cache, then proxy server checks whether that page is modified or not by communicating with actual real server. If not modified, proxy server gets that page from its cache and sends it to the requesting client. If modified, then proxy server should send modified copy of the requested file to the requesting client. The proxy server should maintain log file for caching information. It should contain the information about the most recently served requests, the sender of that request, the name of requested document, the date and time at which the request arrived to the proxy server and the path where the served request is stored i.e. path to cache memory and the last-modification-time of that document. The data send from internal users/clients to external world, has to be stored at the proxy server for system monitoring. This is helpful to monitor whether any secured information is send out of Internal network. The log file of data backup is maintained and it contains the information about the sender and receiver of application data, the time and date of the data transfer and the path where the backup copy of data is stored. The data is stored in compressed format at proxy server by using gzip utility. To speedup the data transfer over the net, the data is send in the compressed format. This module of compression is implemented by using the inbuilt Linux gzip facility. When the client request a data file then proxy server forward the request to internal web server. The data file from internal web server is forwarded to the proxy server in uncompressed format. This data is compressed in gzip format by gzip utility at the proxy server. The compressed data is then send to the requesting client. If the other client is requesting for the same data file, then it can be directly accessed from the proxy cache instead of accessing from the internal web server. If proxy server contains that document but in uncompressed format then it compress that document using gzip and send it to the requesting client, instead of getting that document in compressed format from actual real server and then sending it to requesting user. At client side, this document is decompressed. This saves the time in receiving the document at client sid

Keywords: *Http,Dns,Tcp/Ip,Www,Smtp*

Date of Submission: -24-02-2018

Date of acceptance: 12-03-2018

I. Introduction

Corporate internal networks today are often connected to the Internet to provide employees with the ability to do research using the World Wide Web (WWW). These networks are typically protected from external access or attack via network “firewalls”. Firewalls are special packet routers that allow or deny traffic (typically TCP/IP traffic) based on a variety of criteria. Many organizations utilize the network firewalls to control internal employee access to the external Internet resources and also to regulate external access to intranets. Nations without controlled borders cannot ensure the security and safety of their citizens, nor can they prevent piracy and theft. Networks without controlled access cannot ensure the security or privacy of stored data, nor can they keep network resources from being exploited by hackers. When you connect your private network to the Internet, you are actually connecting your network directly to every other network that’s attached to the Internet directly. There is no inherent central point of security control, in fact there is no security at all. Firewalls are barriers between a secure intranet and the open Internet. It is a system for enforcing access control policy between two networks and is one of the most important measures to protect against network attacks. A firewall may range from impermeable (allowing little or no traffic in or out) to porous (allowing most or all traffic in or out). A Firewall is a device or program that provides security to a network. A firewall provides a checkpoint where all data entering and leaving the network is monitored and controlled. Thus in a firewall, there is a single point for entry and exit for data, ie there is a centralized choke point. All the traffic between the network and the outside world must pass through the firewall. The firewall allows only the authorized traffic to pass through and filters off the rest. A firewall thus enforces a security policy in a network. All points of the network are thus provided with the same amount of security. If you are building a firewall, the first thing you need to worry about is what you are trying to protect. When you connect to the Internet, you are putting three things at risk:

There are three types of firewalls: packet filtering firewalls, Circuit level firewalls and Application layer firewalls

Packet Filtering: Rejects TCP/IP packets from unauthorized hosts and reject connection attempts to unauthorized services.

Circuit level Firewall: Makes high-level transport layer connections on behalf of internal hosts.

Proxy Servers: Makes high-level application connections on behalf of internal hosts in order to completely break the network connection between internal and external hosts..

Packet filtering firewalls works by accepting or dropping the packets based on their source and/or destination address or ports. To provide tougher security, packet-filtering firewall usually need to be used in conjunction with other firewall components. The circuit-level firewall 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. The third type of firewall is the application layer firewall in which special purpose code is used for each desired application. These firewalls make it easy to control all incoming and outgoing network traffic. Most firewalls perform encrypted authentication. This allows users on the public network to prove their identity to the firewall, in order to gain access to the private network from external locations. Some firewalls also provide additional subscription-based services that are not strictly related to security, but which many users will find useful for the features such as virus scanning and content filtering. Virus Scanning searches inbound data streams for the signatures of viruses. Keeping up with current virus signatures requires a subscription to the virus update service provided by the firewall vendor. Content Filtering: Allows you to block internal users from accessing certain types of content by category such as pornography. As hacking attacks and cyber crime incidents continue to increase, many companies are extremely interested in getting insights on the measures their enterprises should take to secure corporate networks. The firewall is the healthy and growing segment in IT market. The Internet is now a critical part of corporate networks, and Internet downtime can cause lost productivity and revenue. The explosion of e-commerce and the growth of the mobile workforce have significantly increased security challenges for the enterprises. Firewall vendors continue to add new features to their products as they compete to solve the increasingly complex problems of securing connections to the Internet, Intranets and Extranets. Network Security has become major concern all over the world with highly vulnerable Internet. With more and more people depending on Internet for their day to day activities, security breaches can be highly costly to the concerned party. Statistics show that there has been a surge in the number of security breaches in recent years. The secure Internet access for the organization can be provided through the stateful packet filtering firewall and

the Application (Proxy) firewall. The stateful packet filtering firewall cannot detect the application level attacks. High-end application firewalls are expensive and are not available because of the export restrictions. A need was felt to develop an indigenous Application level firewall based on the tiny secure proxy servers to meet the custom needs and features. Currently there are many application layer firewalls available in the market. Some of them are commercial firewalls. They are expensive and require much maintenance. While those firewalls that are free to all the users, are not safe to use for the organization because everybody knows their design and implementation. So it may be possible that one can break this firewall. The major commercial organizations can afford the commercial firewalls. But, small organizations, corporate offices, schools and colleges etc., sometimes cannot afford these highly expensive commercial firewalls and their maintenance. So the need come forward to develop the application firewall system for such organizations. Besides standard features of application firewall, the organization requires some extra features to be implemented for them. These features are, backup copy of data at the proxy server, check for virus, data compression over HTTP protocol and storing data in cache in compressed format etc. The organization planned to develop the proxy server with its own structure and implementation, which should not be open to others.

II. Application Layer Firewall

Application-level firewalls are so-called because they operate at the application layer of the protocol stack. An application-level firewall runs a proxy server application acting as an intermediary between two systems. Consequently, application-level firewalls are sometimes referred to as proxy server firewalls. An internal client sends a request to the server running on the application-level firewall to connect to an external service such as FTP, or HTTP. The proxy server evaluates the request and decides to permit or deny the request based on a set of rules that apply to the individual network service. Proxy servers understand the protocol of the service they are evaluating. Thus, they only allow packets through complying with the protocol for that service. They also enable additional benefits: detailed audit records or session information, user authentication, URL filtering, and caching. It runs a proxy server application acting as an intermediary between two systems. The proxy server determines whether to accept or deny the request sent by the client based on some set of rules. Application firewalls never allows the direct connections and forces all network packets to be screened for suitability. It can implement caching at server level, which is far better than that of the client level. Some of the advantages of application layer firewalls are,

User identity could be verified before the network connection is allowed to be established.

1. Descriptive logs could be generated. All traffic going through the firewall could be logged.
2. Simple and cost effective configuration process. Application-gateway firewalls are usually easier to be configured because Internet services could be supported by simply installing proxy servers at the firewall host.
3. Supports information and network hiding. As connections established between the internal and external networks are handled by the proxy servers, information of the internal network can be hidden from the external network.
4. Comparatively less-complex filtering rules.
5. Better controllability. A particular service will not be supported unless the proxy server for that service is explicitly installed. Some disadvantages of the application layer firewall are,
 1. Need a proxy server for each type of supported service.
 2. Network performance is degraded. Because application-gateway firewalls examine the contents of all application level messages across the firewall, network connection speed will be affected. It may not be fast enough to handle high-speed network traffic such as T3 or ATM network.
 3. The firewall is not transparent to users. Proxy server will intercept the communications across the firewall. So that different procedures are required for users to establish Internet services.
 4. Client applications may require modifications. As the client-server communication model is disturbed by the firewall, modification of the client applications may be required.
 5. "Delay" in new service support. Proxy servers will take some time to be developed for supporting new applications.
 6. More than one firewall hosts may be required. From the performance point of view, different servers may be required for different supported services.

An application level gateway has a special proxy code installed for each desired application. If the proxy for a particular application is not installed, then it cannot be forwarded by the application-level gateway. Also specific features of a protocol can be configured denying other features. For instance, consider the Telnet Service. If a telnet proxy is installed, then on establishing a telnet connection, only a subset of the Telnet commands is permitted. The outside client makes a connection with the telnet proxy and telnet proxy makes a connection to the required machine on the internal network. A proxy can be configured to maintain detailed

audit information using logs for the system administrator to check. The proxy should have no loopholes to exploit.

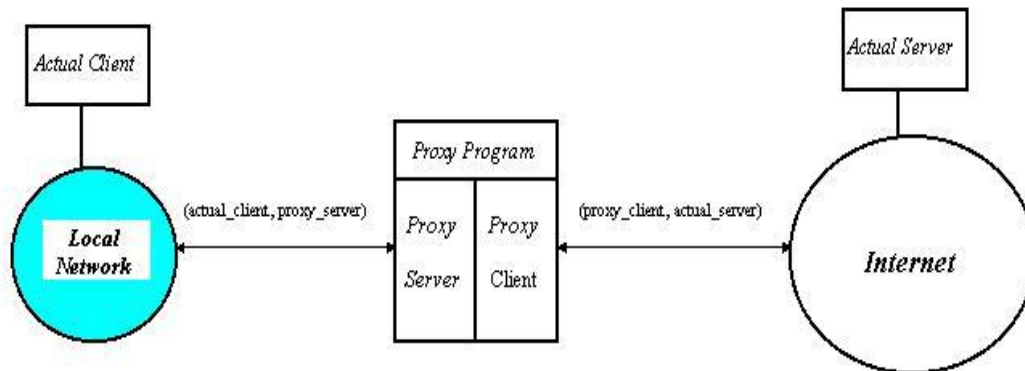


Figure 1: Service Flow of Telnet Proxy

As illustrated above, a proxy contains two parts- A proxy client and a proxy server. The outside client establishes a connection with the proxy server. The proxy server passes the message to the proxy client, which then establishes a connection with the internal server. There are two different types of proxies: Classical proxy
Transparent proxy

Classical Proxy: A classical proxy is commonly used. In this, all packets are addressed to the proxy, and the proxy directs the packets. As above, there are 2 sessions established, a client-proxy and a proxy-server. The client uses the address of the proxy, and the proxy redirects it. Thus, a classical proxy hides the IP addresses of all machines on the network. When a packet arrives at a proxy, then if it is permitted, the proxy prompts the source for the target machine name and ip address. Once this is provided, the proxy forwards it to the destination machine. The proxy can carry out sometimes user authentication. **Transparent Proxy :** The disadvantages of using classical proxies are that they are not transparent, and all addressing is done to the proxy. To avoid this, and make the operation transparent, transparent proxies were devised. In a transparent proxy, the outside machine isn't aware of the existence of a proxy, and the internal machines are addressed directly. A transparent application proxy is often described as a system that appears like a packet filter to clients, and like a classical proxy to servers. Apart from this important concept, transparent and classical proxies can do similar access control checks and can offer an equivalent level of security, at least as far as the proxy itself is concerned. The client machine directly addresses a machine on the internal network. The client has a valid route to the internal machine via the proxy. All the intermediate routers and gateways should lead to the proxy. Once this packet reaches the proxy, the proxy sees that the destination address is not its own and it accepts the packet as if it was its own. The proxy modifies the standard TCP/IP stack and sets the Local IP field to the destination address that the client wants to reach. Hence there is no need for the proxy to ask the client for the address of the destination machine. The advantage of using transparent proxies is that normal client software may access remote servers without changing user procedures.

The following table shows a comparison between classical and transparent proxies.

Issue	Classical proxy	Transparent Proxy
IP addressing	All systems address the proxy.	Client network systems need to address remote networks.
IP routing	All systems need a valid routing entry for proxy.	Client network systems also need routing entries for remote entries.
IP address hiding	Systems on each side of the proxy are hidden from each other.	Systems from client sides are hidden from other sides.
Proxy software requirements	Runs on std. TCP/IP stack. Portable.	Requires special TCP/IP stack. Not 100%portable.
Client software	Requires proxy capable software or user	Nothing more than for a direct connection.

	education.	
DNS	Full isolation possible.	Resolution of outside names by inside systems is required.

Table 2: comparison between classical and transparent proxies

Elements Of Application Layer Firewall

Application layer firewall is composed of many elements some of them are,

1. HTTP Protocol
2. URL Filtering
3. Cache at Proxy Server

Http 1.1 Protocols

The Hypertext Transfer Protocol (HTTP) is an application-level protocol for distributed, collaborative, hypermedia information systems. It is a generic, stateless, object-oriented protocol, which can be used for many tasks, such as name servers and distributed object management systems, through extension of its request methods. A feature of HTTP is the typing and negotiation of data representation, allowing systems to be built independently of the data being transferred. HTTP has been in use by the World-Wide Web global information initiative since 1990. HTTP is also used as a generic protocol for communication between user agents and proxies/gateways to other Internet systems. The specification of HTTP protocol includes the following parts: Request , Methods ,A list of headers in the request message, Response ,Status Codes

Overall Operation: The HTTP protocol is a request/response protocol. A client sends a request to the server in the form of a request method, URI, and protocol version, followed by a MIME-like message containing request modifiers, client information, and possible body content over a connection with a server. The server responds with a status line, including the message's protocol version and a success or error code, followed by a MIME-like message containing server information, entity meta-information, and possible entity-body content. HTTP protocol uses port 80 of server on which it is running. The clients trying to connect the server through HTTP protocol, establish a connection at port 80 of server. To start the communication, the requesting client creates a socket connection with server at port 80 (default port). The client sends the request to the server in HTTP protocol format. After validating the request, server sends the requested file to the client. If file is not present or not accessible then appropriate message is send to the client. The first header line in the request from client to server contains the type of Method, URL and HTTP protocol version. The Method token indicates the method to be performed on the resource identified by the Request-URI. The method is case-sensitive. Some commonly used methods of HTTP protocol are, GET, HEAD, POST, PUT, DELETE etc. URL starts with syntax, " http:// ". The next field in URL is IP Address or Sitename, which is followed by path to the requested file. HTTP version number in request header tells the server which version of HTTP protocol should be used for further communication. Next header lines of request contains the information about the sender of request i.e. hostname, date, client browser information. The response of server contains response header and the requested file. The response header contains status code for request. Commonly used response status codes are,

- "200" : OK
- "304" : Not Modified
- "305" : Use Proxy
- "400" : Bad Request
- "401" : Unauthorized
- "402" : Payment Required
- "403" : Forbidden
- "404" : Not Found
- "405" : Method Not Allowed
- "407" : Proxy Authentication Required
- "501" : Not Implemented
- "505" : HTTP Version not supported

HTTP communication usually takes place over TCP/IP connections. All HTTP date/time stamps MUST be represented in Greenwich Mean Time (GMT), without exception. HTTP messages consist of requests from client to server and responses from server to client.

Example of HTTP request :

```
GET http://gcoea.com/ecomputer/test1.html HTTP/1.1
Host: gcoea.com
```

User-Agent: Mozilla/5.0
Accept: text/html,image/jpeg,image/gif;q=0.2,text/css,*/*;
Accept-Language: en-us, en;q=0.50
Accept-Encoding: gzip, deflate, compress;q=0.9
Accept-Charset: ISO-8859-1, utf-8;q=0.66, *;q=0.66
Keep-Alive: 300
Proxy-Connection: keep-alive
Proxy-Authorization: Basic bWV2anRpOnZqdGl2anRp

Example of HTTP response Header :

HTTP/1.1 200 OK
Date: Sun, 25 Jan 2004 17:23:29 GMT
Server: Apache/2.0.40 (Red Hat Linux)
Last-Modified: Sat, 29 Nov 2003 14:23:43 GMT
Accept-Ranges: bytes
Content-Length: 300
Connection: close Content-Type: text/html; charset=ISO-8859-1

HTTP Authentication

HTTP provides a simple challenge-response authentication mechanism that MAY be used by a server to challenge a client request and by a client to provide the authentication information. It uses an extensible, case-insensitive token to identify the authentication scheme, followed by a comma-separated list of attribute-value pairs which carry the parameters necessary for achieving authentication via that scheme. The 401 (Unauthorized) response message is used by an origin server to challenge the authorization of a user agent. This response MUST include a WWW-Authenticate header field containing at least one challenge applicable to the requested resource. The 407 (Proxy Authentication Required) response message is used by a proxy to challenge the authorization of a client and MUST include a Proxy-Authenticate header field containing at least one challenge applicable to the proxy for the requested resource. The authentication parameter realm is defined for all authentication schemes:

realm = "realm" "=" realm-value
realm-value = quoted-string

The realm directive (case-insensitive) is required for all authentication schemes that issue a challenge. These realms allow the protected resources on a server to be partitioned into a set of protection spaces, each with its own authentication scheme and/or authorization database. The realm value is a string, generally assigned by the origin server, which may have additional semantics specific to the authentication scheme. Note that there may be multiple challenges with the same auth-scheme but different realms. A user agent that wishes to authenticate itself with an origin server--usually, but not necessarily, after receiving a 401 (Unauthorized)--MAY do so by including an Authorization header field with the request. A client that wishes to authenticate itself with a proxy--usually, but not necessarily, after receiving a 407 (Proxy Authentication Required)--MAY do so by including a Proxy-Authorization header field with the request. Both the Authorization field value and the Proxy-Authorization field value consist of credentials containing the authentication information of the client for the realm of the resource being requested. The user agent MUST choose to use one of the challenges with the strongest auth-scheme it understands and request credentials from the user based upon that challenge.

The “If-Modified-Since” Service

The If-Modified-Since request-header field is used with the GET method to make it conditional i.e. if the requested variant has not been modified since the time specified in this field, an entity will not be returned from the server; instead, a 304 (not modified) response will be returned without any message-body.

If-Modified-Since = "If-Modified-Since" ":" HTTP-date

An example of the field is:

If-Modified-Since: Sat, 4 Jan 2003 19:43:31 GMT

A GET method with an If-Modified-Since header and no Range header requests that the identified entity be transferred only if it has been modified since the date given by the If-Modified-Since header. The algorithm for determining this includes the following cases:

1. If the request would normally result in anything other than a 200 (OK) status, or if the passed If-Modified-Since date is invalid, the response is exactly the same as for a normal GET. A date, which is later than the server's current time, is invalid.
2. If the variant has been modified since the If-Modified-Since date, the response is exactly the same as for a normal GET.

3. If the variant has not been modified since a valid If-Modified-Since date, the server must return a 304 (Not Modified) response.

The purpose of this feature is to allow efficient updates of cached information with a minimum amount of transaction overhead. If-Modified-Since times are interpreted by the server, whose clock may not be synchronized with the client.

URL Filtering: URL is an acronym that means Universal Resource Locator. Every resource on the Internet has a unique URL. URLs are used by the browser for locating resources on the Internet. The URL for the page you are visiting is generally displayed on the browser's location bar. Other URLs for page objects are not displayed and are located in page source code.

The following is the example of URL:

<http://www.gcoea.ac.in/PHDcomputer/info.htm>

Now the following is the URL for the image displayed on some page and is located in the page's source code. You won't see this URL displayed in your browser or in your browser's history file.

<http://www.gcoea.ac.in/PhDcomputer/annualday.gif>

URL examined by its component parts:

Component part	Description
http	Protocol type
://	Syntax used following protocol type
Www.gcoea.ac.in	site name
/	Separates sitenames, directories and files
Mecomputer	Directory
Info.htm	file name

IP Addresses and Site Names

For every site name, there is an IP (Internet Protocol) address. One site name can have more than one IP address. An IP address or site name can be used interchangeably in the URL as shown below.

<http://www.webwasher.com> and <http://194.231.14.106> will take you to the same site.

The names are much more commonly used than IP addresses in URLs for a few reasons. One reason is that names are easier to remember and guess than IP addresses. Another reason is that companies often register a domain name the same as the company's name, i.e. microsoft.com. Another reason is that IP addresses can change while the name remains the same, e.g., moving a domain to another server. In order for your computer to connect with a site by name, the IP address for the sitename must be known. Your ISP (Internet Service Provider) generally has a service called DNS, which means Domain Name System that is used for the purpose of translating names to addresses. The sequence of events in making a named connection is, your browser sends a query to your ISP's DNS service. The DNS service returns the IP address for the name to your browser. After receiving the IP address, your browser then proceeds to establish a connection. you wish to filter or block a site you must block it according to the way it defined in the URL. For example if a site is accessed by IP address, filtering it by sitename will be of no avail and visa versa.

Filtering by URL

The URL filter allows us to restrict access according to object type and URL. The filter list is entirely user defined and its content is up to the user. a person want to block the little Visa advertisement at AltaVista's main page. If he is using Netscape or Internet Explorer I could right click on the image and select Add to filter list (some URL filter). The URL filter would then enter the complete URL the filter list as follows: [://www.altavista.com/static/i/visa.gif](http://www.altavista.com/static/i/visa.gif) As long as this URL doesn't change this Visa advertisement, will always be blocked. The URL Filter list doesn't require a full URL. We can block portions a URL through the use of **wildcards**.the above URL we could modify the URL in the filterlist using the wildcard * which means 'anything' and get a variety of results as shown below.

***/visa.gif** :would block AltaVista's visa.gif file and every file called VISA.GIF in the world.

altavista.com/static/ would filter objects in the static directory at AltaVista

/static/ would filter the static directory at AltaVista and all directories of the same name on all sites.

www.altavista.com - would filter one site.

altavista. - would filter content from altavista.com, altavista.org, www.altavista.com as well as other AltaVista sites such as jump.altavista.com.

.com would filter content from every site using the sitename suffix **.com**

IPFilter

Ipfilter is a powerful packet filter. It includes advanced Packet filtering and NAT (Network Address Translation) Functions. With Ipfilter, it's simple to connect a whole private Network to the Internet with just one global IP address. It works the same way as Linux Masquerading. Ipfilter has some Proxy's for common used Application Protocol's like FTP or Realaudio.

Need Of ipfilter

Here are some good reasons for using Ipfilter:

Ipfilter is portable (Solaris, *BSD, True64, Unix, Linux). Changing the Platform were Ipfilter runs is possible and simple. Runs on Risc and CISC CPU's.

Ipfilter is stateful (tcp, udp and icmp). Sessions (TCP 3 Way Handshaking) and request/ response (UDP/ICMP) are recognized in the sense of related packets.

Ipfilter has a powerful well thought out filtering Language similar to the Cisco Pix and way better than netfilter (cmd line).

Ipfilter has some NAT features not available in Linux (1x1 or blocks of IPs).

Ipfilter can cope with thousands of Rules Ipfilter has many Options for Filtering traffic based on IP, Proto, Port and Header Fields Installing Ipfilter is simple (Solaris) if needed at all (*BSD). Just a Kernel Modul plus some User Programms. Only two simple configuration Files.

Packet Filtering Router: A packet filtering router is basically a simple router, which scans every packet that is forwarded to it. The router then decides whether to allow the packet to enter the network or to discard it. The router has routing table entries regarding which packets to accept and which ones to discard. When a packet arrives at a router, the router compares its table entries with the packet header information. If the table entry permits the packet, then it is allowed to pass through, otherwise it is discarded. The rules are mostly based on the source or destination address and the type of service, which is given by the port number. For instance, the router can be configured to discard packets arriving from a particular source, or to discard all FTP packets i.e packets having destination port 21. Thus there are two ways to filter:

- Filtering by Address
- Filtering by Service

Filtering by Address : Filtering by address is the simplest method, but not commonly used. Filtering is done on the basis of source and/or destination addresses. This type of filtering allows only certain external hosts to talk to certain internal hosts. The following are a typical set of packet filtering rules.

Rule	Direction	Source address	Destination address	Action
A	Inbound	173.54.10.221	Any	Permit
B	Outbound	Any	173.54.10.221	Permit
C	Either	Any	Any	Deny

- Rule A permits any packet from a trusted host 173.54.10.221 to be forwarded to any machine in the internal network
- Rule B permits all packets from the internal network to be forwarded to 173.54.10.221
- Rule C denies packet transfer in either direction.

Thus the outcome of above 3 rules is that only packets to and from the trusted host, i.e 173.54.10.221 is permitted. Everything else is denied.

The drawback of using address filtering is that any hacker can spoof the IP of the host, i.e. he can forge the host IP address and send packets with the forged address to the packet filtering router. The router has to permit the packet as it is from a known source IP. Thus filtering based on Address is insecure.

Filtering by Service

In this, filtering is carried out on the basis of a particular protocol or service. The packets thus have fields containing source as well as destination port numbers. Here the use of any protocol can be restricted. For example, if in an organization, the system administrator feels that FTP file transfer could lead to security

breaches, then he can configure the packet filtering router to deny all packets having destination port 21(FTP port). The following is a set of filtering rules based on service.

Rule	Direction	Source address	Destination address	Packet Type	Source Port	Dest. Port	Action
A	Incoming	173.54.26.9	Any	TCP	>1023	25	Permit
B	Outgoing	any	Any	TCP	25	>1023	Permit
C	Either	any	Any	TCP	any	any	Deny

- Rule A permits an SMTP transfer(port 25) from 173.54.26.9 to any machine in the network
- Rule B permits an SMTP transfer from any machine in the network to outside
- Rule C denies any other packets other than ones in Rules A & B

Even though service filtering is secure, it is not necessarily foolproof, as port numbers for a particular service can be changed. The main drawback of using packet-filtering routers is that a large number of table entries have to be maintained by the router, i.e. for each address or service, a separate entry has to be made. This causes an increase in overhead.

Caching At Proxy Server : Before serving a document a proxy server should check if the original document still exists in its cache and if it is unchanged.

- if it no longer exists the no document should be served and the cache copy should be purged.
- If it has changed then the cache should be refreshed and the new version served.
- If the original site cannot be contacted then the document should be served with a warning that the validity of the document could not be verified.
- If the document still exists unchanged then the cache copy can be served.

Proposal for Caching

The cache coherency problem is a separate issue because it requires a special request to the original server to determine the status of the actual document. Although it is suggested that the HEAD request is sufficient for this purpose, but it is found that it entirely too inefficient for a caching mechanism (because of the server overhead from connecting twice and finding the file twice). The solution is to implement a conditional GET request -- one that includes a date to be checked against the last-modified date of the information object. The normal GET request was followed by some sort of last modification date header similar to the current Content-type, Authorization, etc. Let's call this an If-Modified-Since header. Formally, if the server receives the request in the following form: GET /ICShome.html HTTP/1.0 If-Modified-Since: Thu, 06 Jun 2003 15:57:41 GMT

Then the server would respond in one of the following ways:

- If the object /ICShome.html is inaccessible (for whatever reason), then the server should return a 4XX message just like it does now.
- If /ICShome.html no longer exists, the server should return a 404 Not Found response (i.e. same as now).
- If /ICShome.html is accessible but its last modification date is earlier (less than) or equal to the date passed (Thu, 06 Jun 2003 15:57:41 GMT), the server should return a 304 Not Modified message (with no body).
- If /ICShome.html is accessible and its last modification date is later than the date passed (Thu, 06 Jun 2003 15:57:41 GMT), the server should return a 200 OK message (i.e. same as now) with body.

In this way, cache managers would just send a GET request with the If-Modified-Since date equal to the date it originally requested the local copy of the object it has in its cache.

Implementing this protocol would have no effect whatsoever on existing servers and clients. Old clients (and any without caches) would just continue making requests without If-Modified-Since headers. Old servers (at least the NCSA httpd 1.0 and 1.1) will already accept a message of the above format and just ignore the If-Modified-Since header.

Security Advantages Of Proxies: The process of request regeneration and the fact of a proxy's location between the external and internal networks provide a number of security advantages:

Client Hiding: The major security feature of proxy servers is client hiding. Like Network Address Translation, proxy servers can make an entire internal network appear to be a single machine from the Internet because only a single machine passes requests onto the Internet. Like Network Address Translators, proxy servers prevent external hosts from connecting to services on internal machines. In the case of proxy servers, no

route to the clients exists because the address domains of the internal and external networks may be incompatible and because transport layer routing does not exist between the two networks.

Proxies perform this feature by completely regenerating service-level requests rather than simply changing and recalculating address headers. For example, when a web client makes a request through a proxy server, the proxy server receives the request as if it were the destination web server on the internal network. It then regenerates the request on the external network as if it were atypical web browser. When the proxy receives the response from the ultimate web server, it serves that response to its internal client. Only HTTP passes through the proxy, not TCP or IP. TCP/IP (and other low-level protocols) are regenerated by the proxy; they do not route through it unless the proxy is misconfigured. Another aspect of client hiding is that of connection multiplexing; a proxy server can be used to share a single Internet connection and IP address among an entire network.

URL Blocking: URL blocking allows administrators to disallow the provision of certain websites based on their URLs. In theory, this will keep your employees from viewing websites you don't want them to have access to. This function is easy to implement. The proxy simply checks every request for a web page (or other service URL) against a list of denied pages before it regenerates the request. If the URL is blocked, the proxy will not request or return the page. URL blocking is easy to circumvent, however, because a website can be just as easily addressed by its IP address or even by the whole number address. For example, a user could type in any of the following in their web browser to access exactly the same home page:

<http://www.linuxexposed.com/index.php>

<http://212.190.116.128/index.php>

<http://59672698250359936/index.php>

But your URL blocker will (probably) only be checking for the full text URL. URLs can contain DNS names or IP addresses. Most people are familiar with the first two examples of site references, but have never heard of the third: an IP address specified as a whole number rather than as a "dotted quad notation." The concept is simple: An IP address is just a 32-bit number, and though we refer to them in dotted quad (10.0.0.0) notation for convenience sake, there's no reason why they can't be referred to as whole numbers. To convert a dotted quad number to a whole number, use the following formula ("a" is the most significant quad, "d" the least):

$a \times 224 + b \times 216 + c \times 28 + d.$

Converting everything to easily calculable numbers, the formula becomes :

$a \times 16777216 + b \times 65536 + c \times 256 + d.$

So, for example, turning the IP address for <http://www.linuxexposed.com/>, 209.68.11.152, into a whole number makes it $209 \times 16777216 + 68 \times 65536 + 11 \times 256 + 152 = 59672698250359936$. Put 59672698250359936 into your web browser's address bar and you'll see the Linux Exposed web page come up. Note that websites behind proxy servers (like Microsoft.com) don't come up because the whole number IP address must be programmed in to the proxy for the proxy to recognize it.

The other major problem with URL blocking for security administrators is simply keeping up with sites to block. Problem sites like hacking depositories, pornographic sites, and game sites have the ephemeral life of a mayfly they pop up and disappear just as quickly. Most people who engage in the activities ascribed by these sites just use search engines or Usenet news lists to keep up with where their favorite sites have moved. You will not be able to stay ahead of that activity with your URL-blocked database.

Content Filtering: Because proxies retransmit all protocol payloads and are protocol specific, the proxy service can be used to search the payload for suspicious content. This means that you can configure your HTTP proxy service to strip out ActiveX controls, Java applets, or even large images if you feel they could present a security problem. You could also use an SMTP proxy to strip out executable file attachments and archived zip files if you felt they were a security problem. Content filters can also be used to check web pages for the presence of certain words or phrases, such as the trademarks of your competition or some current news item. You should filter ActiveX controls in websites, Java applets, and executable files in e-mail because they can be used to install Trojan horses inside your network. If someone needs to transfer an executable file, have him or her transmit it as a zip file or use Bin Hex or some other encoder to transfer it in a text format. This will require effort to decode, thus preventing the accidental transfer of a virus or Trojan horse into your network.

Consistency Checking: Consistency checking refers to checking the content of a protocol to be sure it makes sense for that protocol. Consistency checking ensures that specifically malformed types of content can't be used to exploit a security weakness in your internal network. For example, a buffer overflow occurs in URLs that are

longer than 256 characters. Early web browsers were flawed because the end of the URL beyond 256 characters could contain executable code that would be executed by the browser software.

Consistency checking with your proxy software can ensure that these sorts of problems are eliminated at the proxy so they won't affect internal machines. Unfortunately, the problems to check for usually are not known until some hacker exploits them, so most consistency checks are only available after an exploit has been found. And with automated worms, a large portion of the web servers on the net can be exploited within a few hours, so the "countermeasure" aspect of hot fixing servers is rather ineffective.

Route Blocking: Transport layer packets need not be routed because the request is completely regenerated. This eliminates Transport layer exploits like source routing, fragmentation, and various denial-of-service attacks. By eliminating routing, you can also ensure that any protocol for which you have not established a proxy service cannot be passed to the public network. Route blocking is perhaps the most important advantage of proxy servers. Because no TCP/IP packets actually pass between the internal and external networks, a vast number of denial-of-service and exploitation attacks are prevented. Unfortunately, route blocking is not used often enough. Because many protocols exist for which there are no good proxy services, administrators often must enable routing on the proxy server, which completely eliminates the security gain achieved by route disconnection. If you can, avoid allowing low-level network packets to pass through your proxy server. Most proxy server software will allow you to create generic TCP proxy services for any port using a generic SOCKS proxy or the Unix `redir` utility. These generic proxies, although they cannot perform content filtering, still allow you to keep TCP/IP packets from flowing between your networks.

Logging and Alerting: The final security advantage of proxies is the logging and alerting facilities they provide. Proxies ensure that all content flows through a single point, which gives you a checkpoint for network data. Most proxy software will log the usage characteristics of the proxy by user and can be configured to retain a log of sites they visit. This will allow you to reconstruct the user's web browsing sessions if you suspect some illegal or unethical activity has occurred. The alerting facility provided by some proxies can alert you to attacks in progress, even though the proxy facility of a server is not generally subject to attack. But the facility can alert you to attempted proxy connections from the external interface, which hackers frequently try to exploit to launder their connections.

Filtering decision:

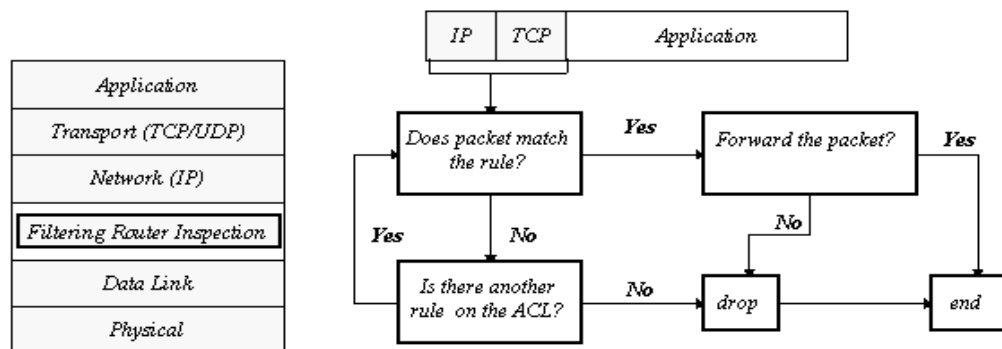
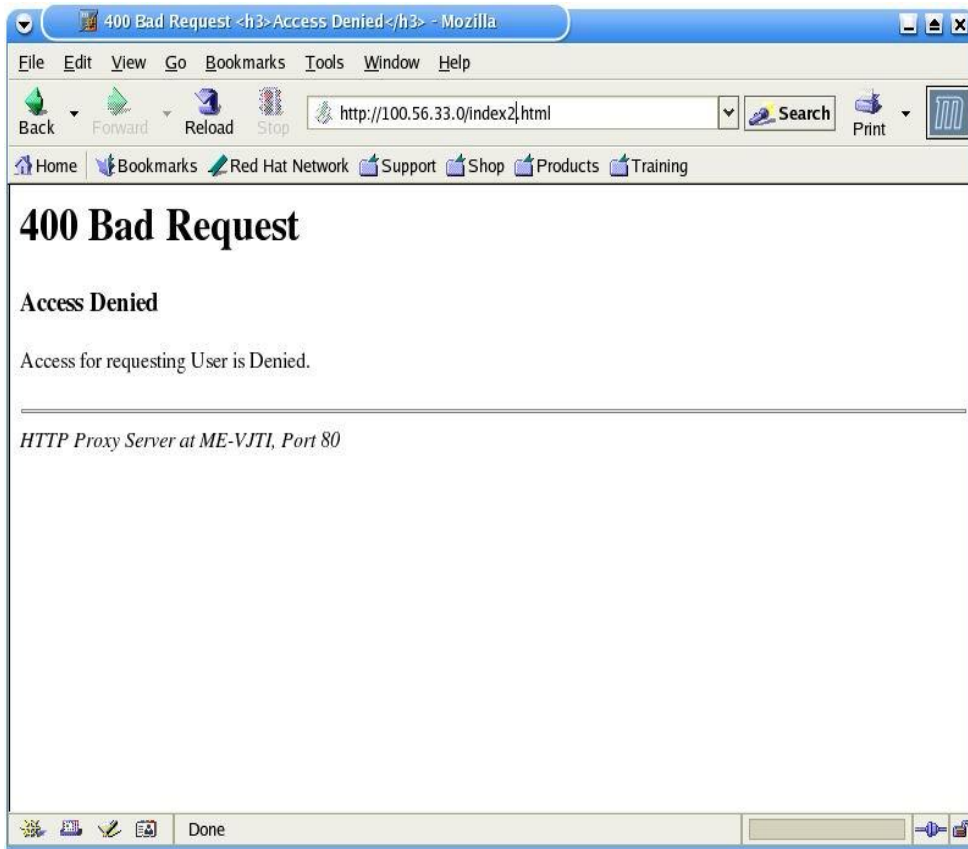


Figure 1 Step of filtering decision

The above figure shows the process of Filtering decision and the steps involved in it. The process works on the approach that, all, which is not expressively permitted, is prohibited. Consider the situation when the packets that does not match any entry in the ACL? In this situation two different approaches may be adopted:

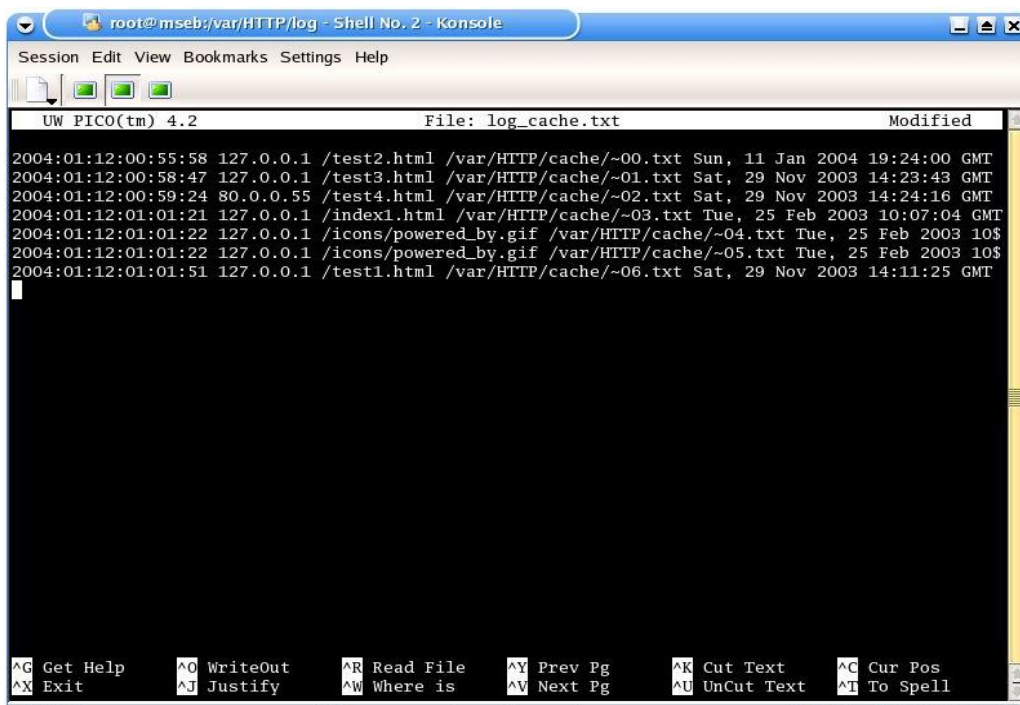
- That which is not expressively permitted is prohibited, that is the Filtering Router will drop these packets
- That which is not expressively prohibited is permitted, that is these packets will forwarded by the Filtering Router
- To ensure a high-quality product, diagrams and lettering MUST be either computer-drafted or drawn using India ink.

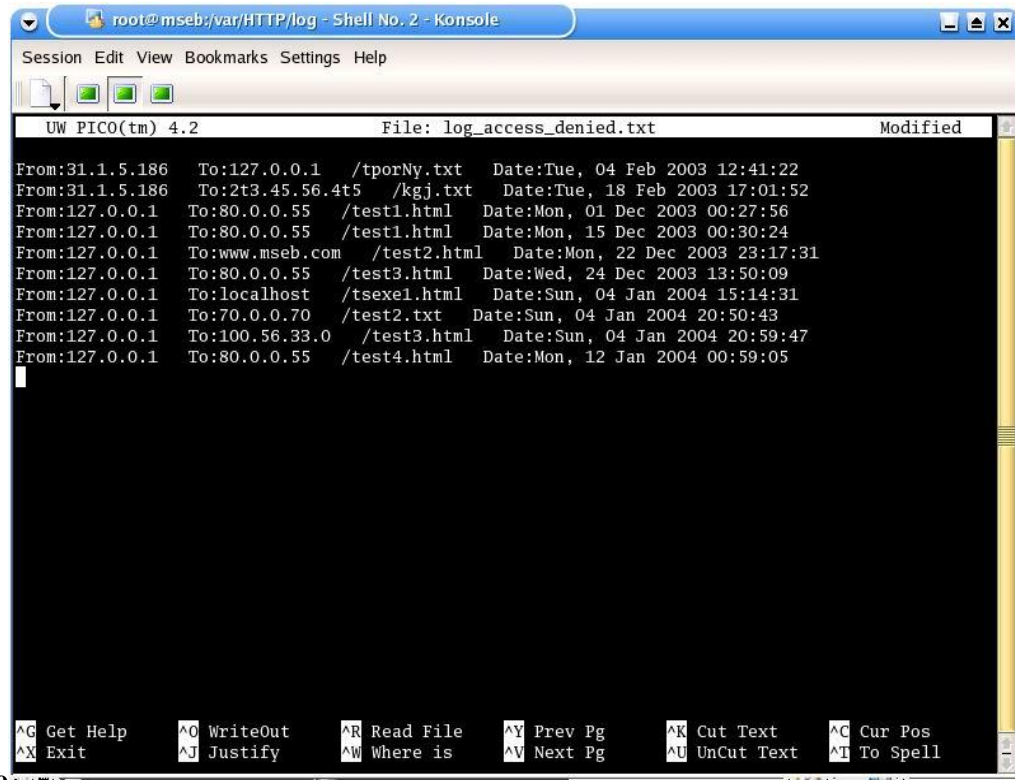
III. Screens



Screen1: This screen shows the response of proxy server to the requesting user when the request from such user contains virus. If virus found then the user is informed by sending above message from proxy server.

Screen2 : This screen shows the Log-file for cache at Proxy Server. This File contains entries for last access date and time, IP address of requesting client, file name, storage path of file in cache at proxy server and last-modification-Date for the file



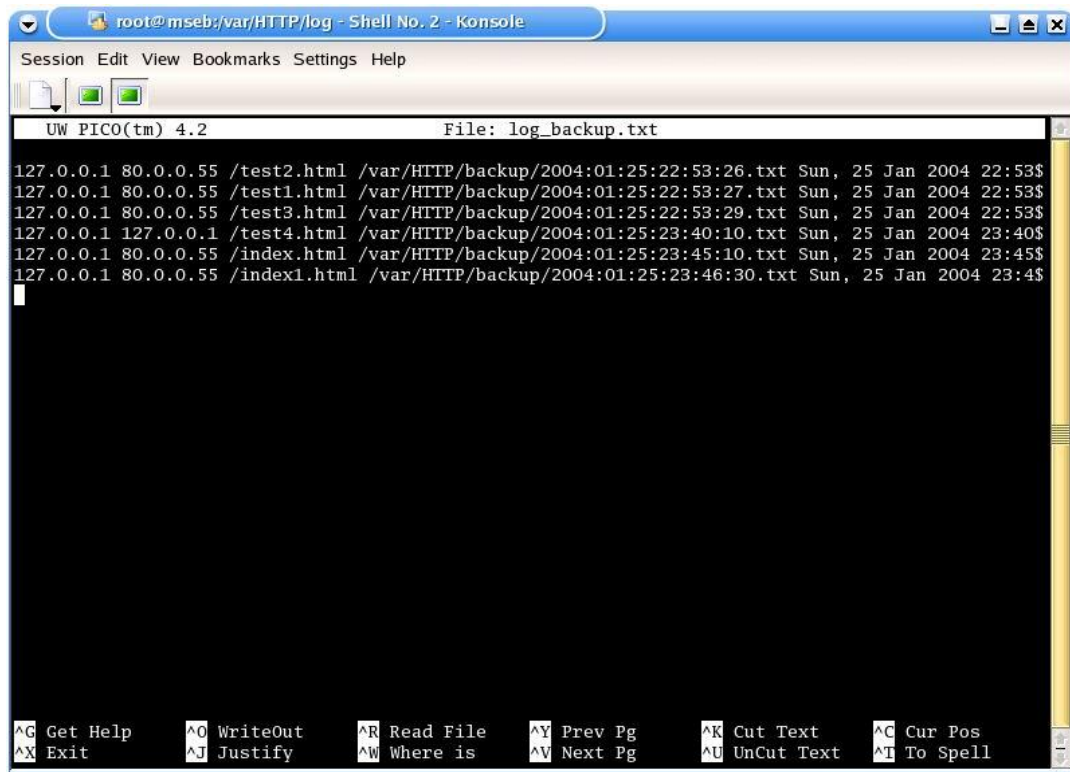


```

root@mseb:/var/HTTP/log - Shell No. 2 - Konsole
Session Edit View Bookmarks Settings Help
UW PICO(tm) 4.2 File: log_access_denied.txt Modified
From:31.1.5.186 To:127.0.0.1 /tporNy.txt Date:Tue, 04 Feb 2003 12:41:22
From:31.1.5.186 To:2t3.45.56.4t5 /kgj.txt Date:Tue, 18 Feb 2003 17:01:52
From:127.0.0.1 To:80.0.0.55 /test1.html Date:Mon, 01 Dec 2003 00:27:56
From:127.0.0.1 To:80.0.0.55 /test1.html Date:Mon, 15 Dec 2003 00:30:24
From:127.0.0.1 To:www.mseb.com /test2.html Date:Mon, 22 Dec 2003 23:17:31
From:127.0.0.1 To:80.0.0.55 /test3.html Date:Wed, 24 Dec 2003 13:50:09
From:127.0.0.1 To:localhost /tsexel.html Date:Sun, 04 Jan 2004 15:14:31
From:127.0.0.1 To:70.0.0.70 /test2.txt Date:Sun, 04 Jan 2004 20:50:43
From:127.0.0.1 To:100.56.33.0 /test3.html Date:Sun, 04 Jan 2004 20:59:47
From:127.0.0.1 To:80.0.0.55 /test4.html Date:Mon, 12 Jan 2004 00:59:05
^G Get Help ^O WriteOut ^R Read File ^Y Prev Pg ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where is ^V Next Pg ^U UnCut Text ^T To Spell

```

Screen 3: This screen shows Log file for Access-Deny at Proxy Server. This file contains the entries for IP address/SiteName of Source and Destination of the request, the file name and date and time at which request arrives at proxy server



```

root@mseb:/var/HTTP/log - Shell No. 2 - Konsole
Session Edit View Bookmarks Settings Help
UW PICO(tm) 4.2 File: log_backup.txt
127.0.0.1 80.0.0.55 /test2.html /var/HTTP/backup/2004:01:25:22:53:26.txt Sun, 25 Jan 2004 22:53$
127.0.0.1 80.0.0.55 /test1.html /var/HTTP/backup/2004:01:25:22:53:27.txt Sun, 25 Jan 2004 22:53$
127.0.0.1 80.0.0.55 /test3.html /var/HTTP/backup/2004:01:25:22:53:29.txt Sun, 25 Jan 2004 22:53$
127.0.0.1 127.0.0.1 /test4.html /var/HTTP/backup/2004:01:25:23:40:10.txt Sun, 25 Jan 2004 23:40$
127.0.0.1 80.0.0.55 /index.html /var/HTTP/backup/2004:01:25:23:45:10.txt Sun, 25 Jan 2004 23:45$
127.0.0.1 80.0.0.55 /index1.html /var/HTTP/backup/2004:01:25:23:46:30.txt Sun, 25 Jan 2004 23:4$
^G Get Help ^O WriteOut ^R Read File ^Y Prev Pg ^K Cut Text ^C Cur Pos
^X Exit ^J Justify ^W Where is ^V Next Pg ^U UnCut Text ^T To Spell

```

Screen 4: This screen shows the Log-file for backup at Proxy Server. This File contains the entries for latest data passed through the proxy server. This log file contains source and destination of request, the requested file, and the path where the file is stored in compressed format at proxy server and time and date when the request is served by proxy server

IV. Conclusion

The journey through this paper work has been tremendous learning experience. This paper work is developed for small organizations such as schools, colleges, small industries etc. The this paper was to provide the basic functionality of application layer firewall (Proxy Server) along with some user specific requirements. However, this project can be extended to provide some advanced functionalists such as "POST" and "DELETE" methods for the HTTP protocol. Also GUI based options can be provided to the user. This project can be extended to provide application firewall for other application protocols such as SMTP, DNS. The application firewall is implemented for the organization. The facilities provided are Internal user authentication, external user authorization, virus scan facility, allow/deny access to secure data files, log files for cache, backup and access denied. These facilities are shown in the screen shots of proxy server. This implementation provides various benefits such as user specific system design, less maintenance cost, easy to operate and modify as its design is available with user, backup facility for system monitoring. The proxy server is implemented in C language on LINUX O.S. The user friendly GUI is provided for the Proxy server. Thus the implementation of the application layer firewall at organization level provides more benefits than the commercial, purchased firewalls to make the organization secure from the external attacks. The facilities provided are Internal user authentication, external user authorization, virus scan facility, allow/deny access to secure data files, log files for cache, backup and access denied. These facilities are shown in the screen shots of proxy server. This implementation provides various benefits such as user specific system design, less maintenance cost, easy to operate and modify as its design is available with user, backup facility for system monitoring. The proxy server is implemented in C language on LINUX O.S. The user friendly GUI is provided for the Proxy server. Thus the implementation of the application layer firewall at organization level provides more benefits than the commercial, purchased firewalls to make the organization secure from the external attacks.

Firewall can be handled for following implementations. The important extensions can be:

- [1] Implementation of proxy server to FTP, DHCP, SMTP and other complex protocols
- [2] Authentication of the user accessing the system
- [3] Authorization techniques can be embedded to greater extent with various available techniques

Also, for future research, current simple model of single firewall should be to be extended to accommodate the following

- I. Complex distributed networks with multiple firewalls and perimeter layers
- II. Various and distinct firewall technologies available in various hardware and software platforms
- III. An effective mechanism to handle a feedback from application layers such as mail servers detecting incoming massive spam mails
- IV. Efficient and real-time algorithms to handle massive volume of log files and data mining
- V. further investigation and analysis on time-dependent statistical behavior of network traffic and policy rules for the faulty and leaky network inside of firewall perimeter
- VI. the aggregation of non-contiguous subnet masking and 128-bit IP address of IPv6

References

- [1] DOUGLAS E. COMER, DAVID L STEVENS "INTERNETWORKING WITH TCP/IP" VOL. III, PEARSON EDUCATION, 2001
- [2] W. Richard Stevens "Unix Network Programming", Prentice-Hall Inc., 2000
- [3] Chapman D. Brent and Zwicky Elizabeth D., "Building Internet Firewalls", O'Reilly & Associates, Inc., 1995
- [4] Matthew Strebe, Charles Perkins "Firewalls, 24 Seven" 2nd Edition, BPB Publications, 2003
- [5] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, "Hypertext Transfer Protocol -- HTTP/1.1" RFC 2616, June 1999.
- [6] Bill Ball, David Horvath, Abhijit Menon, "Redhat Linux 7.2 Unleashed", SAMS Publications, 2002
- [7] William R. Cheswick and Steven M. Bellovin, "Firewalls and Internet Security: Repelling the Wily Hacker" Addison-Wesley, 1994.
- [8] Squid Internet Object Cache User guide, Available on-line at <http://squid-cache.org>
- [9] Robert L. Ziegler, "Linux Firewalls", New Riders.
- [10] <http://www.cache.ja.net/events/workshop/32/manchester.html>

Theses:

- [11] Dr P.B.Ambhore "Carapace for Intranet Security in the year 2015 ,Government College of Engineering, Amravati, India
- [12] Note that thesis title is set in italics and the university that granted the degree is listed along with location information

Proceedings Papers:

- [13] Velasco J.R, Velasco L.A., "Benefits of Compression in HTTP Applied to Caching Architectures" 3rd International WWW Caching Workshop, 1998.

Dr.Premchand.B.Ambhore "Proxy Server FOR Intranet Security." IOSR Journal of Computer Engineering (IOSR-JCE) 20.2 (2018): 01-14.