Unit4 - Firewalls and Perimeter Defence

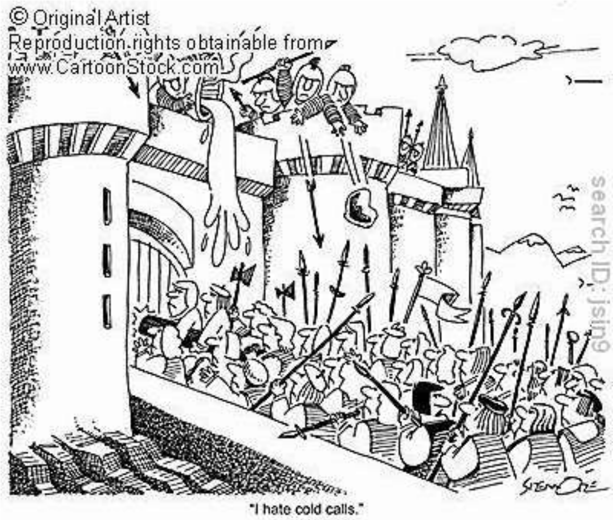
Rich Macfarlane

This unit gives an overview of firewall concepts. It also includes an introduction to the various different types of firewalls currently used for network security, as well as some of the most common topologies implemented. The remainder of the unit contains a taxonomy of the most common current firewall architectures.

4.1 Introduction

Firewalls are applications or devices which controls the data flowing between two different networks. They are typically deployed on the perimeter of an organisations network, where it connects to external networks such as the Internet. Firewalls can also be placed between internal networks to enforce different security postures, and give a layered approach to network security. For example, internal firewalls might be used to protect critical network segments like server farms. Firewalls can also be deployed on mobile users systems, outside the organisations network, such as staff working from home.

In buildings a firewall is a concrete wall which prevents fires from spreading from one part of the building to another. In vehicles, a firewall is barrier between the engine and the passenger compartment. In network security, a firewall is a barrier between two networks, used to separate them at a trust boundary. The firewall should only let specified “good traffic” between the networks.



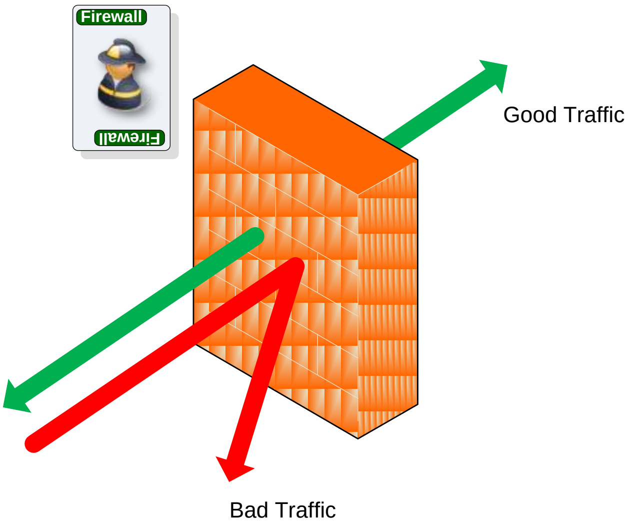
"I hate cold calls."

Firewall is like a medieval castle gate

"Is that the firewood we ordered or are we under siege?"

A good analogy is a **Medieval Castle and the Guard on its Gate**. The walls keep certain people in, and certain people out. The Gate is the only place to move between the outside and the inside of the castle, and the Guard vets the people passing through the gate and decides if they can enter or leave the castle.

A firewall should be the only transit point between the two networks. It's a choke point which all traffic must pass through. A Network firewall allows good traffic to pass between one network and the other, and blocks bad traffic. The good and bad traffic, in each direction, should be defined in the Network Security Policy.



Firewall Types

There are many types of firewalls, ranging from desktop software firewalls to enterprise appliances, which use various different technologies to allow or block network traffic. To be able to implement firewalls, the strengths and weaknesses of the different types of firewalls have to be appreciated, as well as where to deploy firewalls within a network. The following sections describe firewall concepts, the main types of firewalls and their architectures, and the network topologies which can be used to deploy perimeter firewalls.

Firewalls can be split into three types, Host-Based Personal Firewalls, Network Server-based Firewalls, and Network Device-based firewalls.

Host-based Personal Firewalls

Firewall software running on a host machine. The software runs on the machines general purpose Operating System (OS), giving the user protection when connecting to the Internet. Used by home users, and mobile business users. In an organisation they can provide another layer of security, when used in conjunction with other firewalls. Examples include **Windows Firewall** (integrated into XP and Vista), **ZoneAlarm** which is owned by Checkpoint - who also make enterprise firewalls, and **Comodo Personal Firewall**.

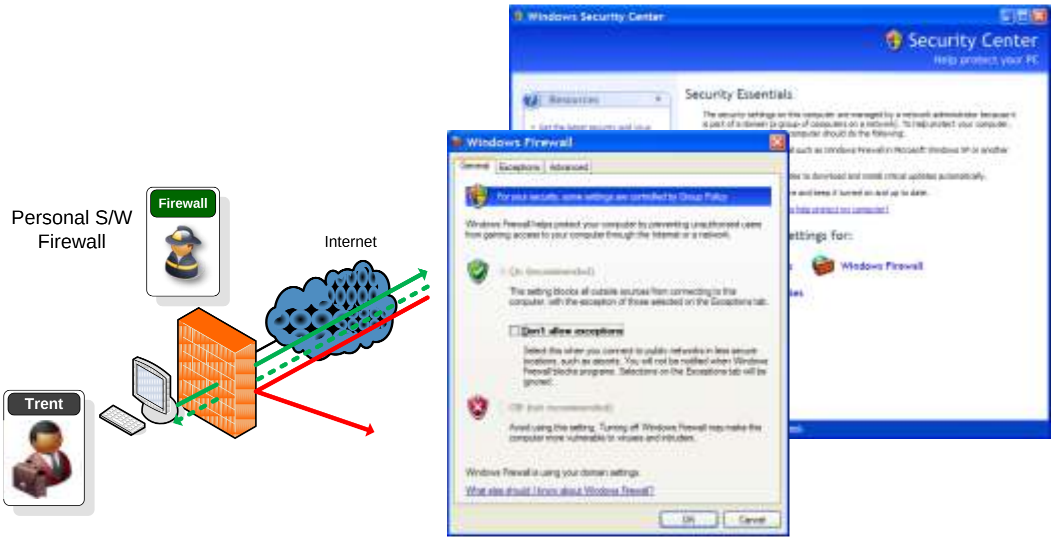


Figure 1 Personal Firewall - Windows Firewall (Windows XP)

Server-based Network Firewalls

Firewall software running on a general purpose server. The software runs on a server, on top of a network Operating System (OS), such as Unix, Linux, Windows, or Novel. It would have its OS hardened, removing any services that were not essential. Examples of this type of firewall would be Checkpoint FW-1, Microsoft Internet Security and Acceleration Server (ISA), Linux IPtables, and Novell BorderManager. Network Software firewalls can provide powerful solutions for small to medium sized businesses. They are sometimes incorporated into in an all-in-one security product, along with Virtual Private Network (VPN) functionality, and Quality of Service (QoS) provision, such as those provided by Checkpoint. The Checkpoint firewall/VPN management GUI is shown in

Figure 2.

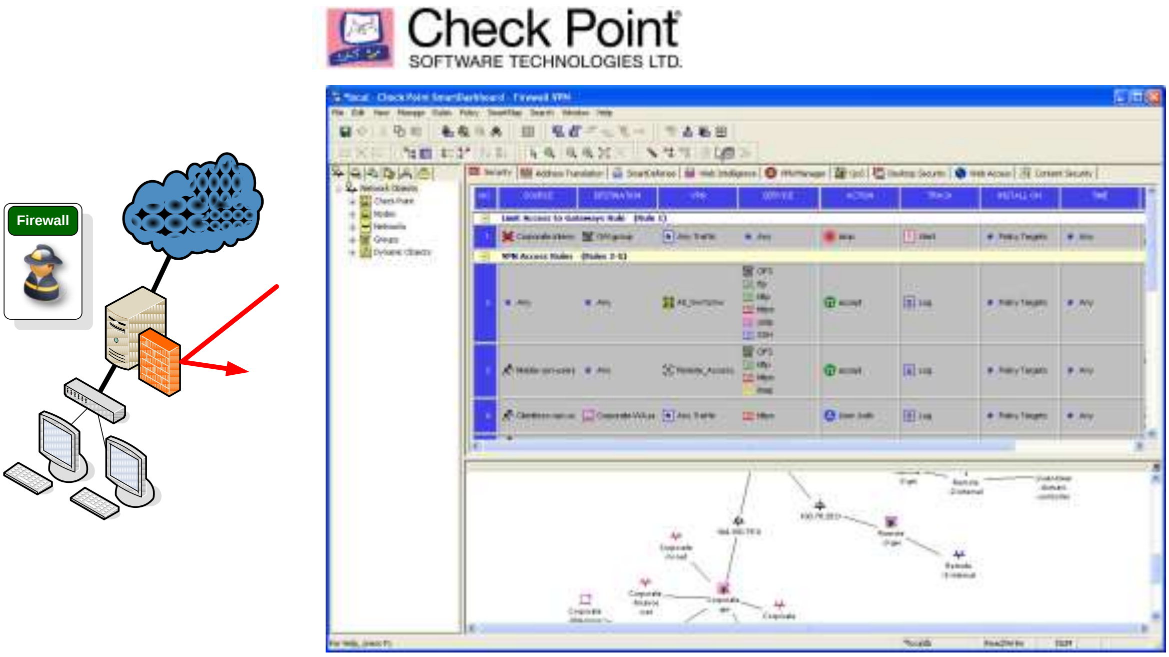
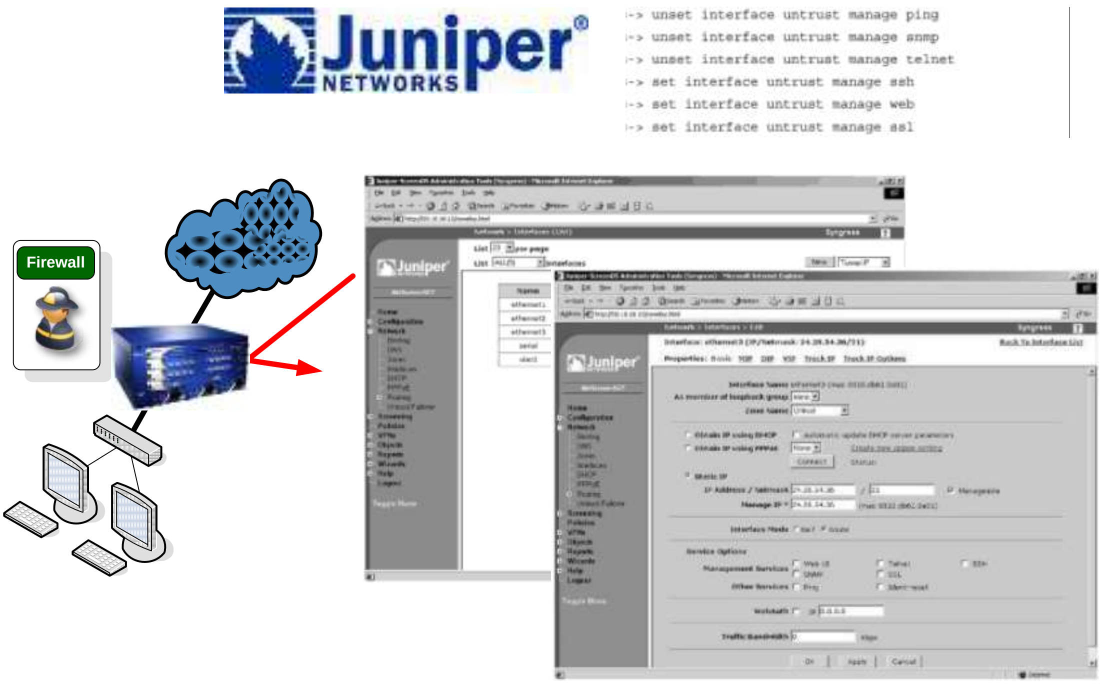


Figure 2 Server-Based Network Firewall – Checkpoint Firewall GUI

Device-based Network Firewalls

These are network firewalls, running on dedicated network devices. They range from firewall software running on a router, up to enterprise grade firewall or security appliances – with custom software running on custom hardware, optimised for purpose. They would typically be deployed as perimeter firewalls for all sizes of organisations, from an employee working from home using a small router with built in firewall, to a large organisation using an enterprise perimeter security appliance including firewall functionality.

For example, a home user or Small Office/Home Office (SOHO) network, could use a router from **DLink, Netgear**, or **Linksys** (Linksys is owned by Cisco) A Linksys WRT series wireless router, at around forty pounds, which runs Linux with IPTables, provides stateful firewalling and Network Address Translation (NAT). A large company might deploy one or more **Cisco ASA 5500 series** appliances, costing several thousand pounds, as a perimeter firewall for the whole organisation. An ASA provides Stateful and Application Protocol Inspection Firewalling, VPN termination, and an Intrusion Prevention System (IPS), and very high throughput. Some of the top hardware firewall vendors are **Cisco**, **Juniper**, **Nokia** – running Checkpoint software, and **SonicWall**.



```
-> unset interface untrust manage ping
-> unset interface untrust manage snmp
-> unset interface untrust manage telnet
-> set interface untrust manage ssh
-> set interface untrust manage web
-> set interface untrust manage ssl
```

Figure 3 Device-Based Network Firewall – Juniper Appliance, & management via the CLI and GUI

Configuration of hardware firewalls is typically more complex than software firewalls, and historically was done via a Command Line Interface (CLI), although both Cisco and Juniper now offer fully functional GUI configuration tools for all functionality. Hardware firewalls generally have better performance than software firewalls, as the hardware and software they use is optimised for packet inspection and network throughput, rather than a general OS such as Windows, which would have to be configured as best it could. Research shows Cisco hardware copes more gracefully with high traffic loads, and software firewalls tend to have a big hit on performance as the firewall policy increases in size. Figure 4 shows the different types, and how they are deployed.

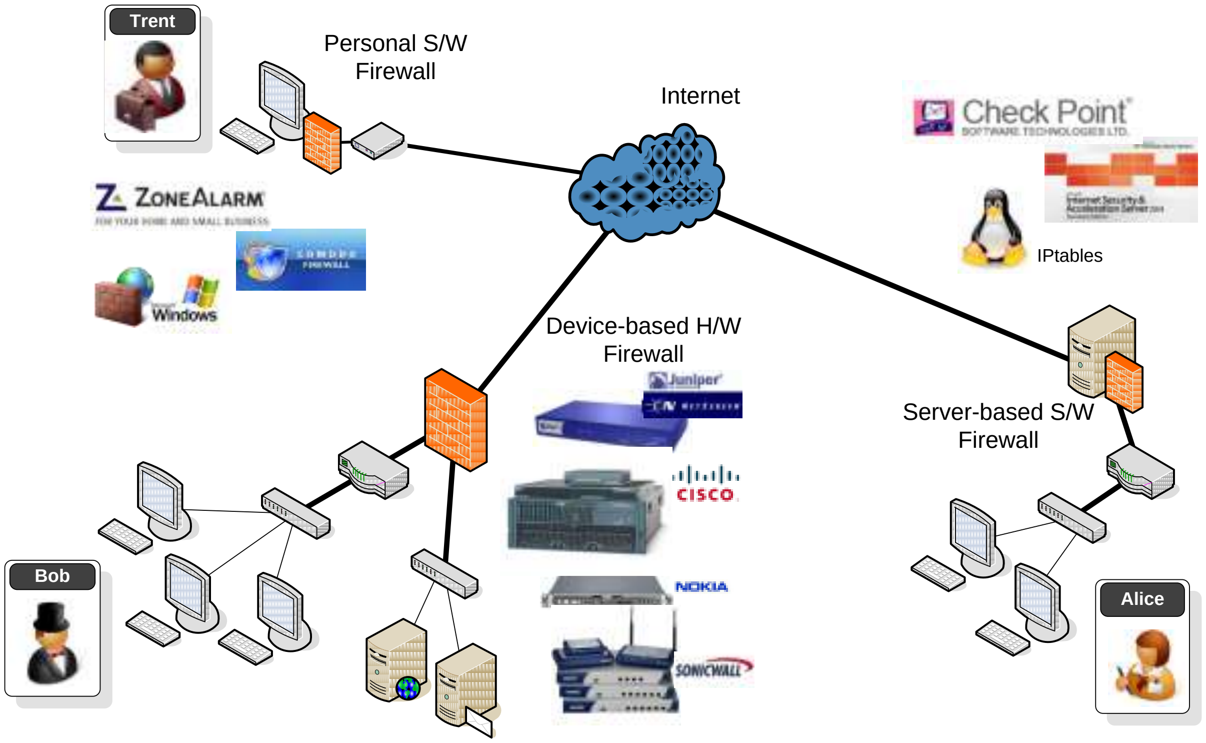


Figure 4 Perimeter Firewall Types and Vendors

Software firewalls tend to be implemented on OS's which can be extensively configured at a low level such as Linux, so they can be hardened against attacks. The OS is typically hardened against direct attacks, as the firewall is the last thing an administrator would want to be compromised. On a software-based firewall, the OS would have to be hardened manually, by an administrator before the firewall software is installed.

Typically, hardware firewalls are much higher in cost, but this depends on the functionality and throughput provided, and most hardware firewall vendors have a range of products, at different price points. A software firewall can be run on general purpose hardware, and so only involves the cost of the firewall software and the time to harden the server. Hardware firewalls also typically have better support for failover mechanisms than firewall software. A summary of the main features of hardware and software firewalls is given in Table 1.

Table 1 Hardware vs Software Firewalls

	Hardware Firewall	Software Firewall
Configuration	Complex	Easier
Performance	Faster – Optimised Hardware	Slower
Device Security	Good – Hardened OS	General OS – Needs hardened
Cost	Higher	Lower

4.2 Firewall Concepts

Security Policy

Before any decisions can be made on firewalls, and the individual policies they should have, an overall network security policy should be created. Risks to the network - from the risk analysis, and management - high level policies can be used to create the overall **network security policy**. This is then used to specify firewall policy guidelines and the firewall policy procedures. Procedures will define the traffic firewalls allow to flow through them, and the guidelines define how the firewall should be implemented in general. This is shown in Figure 5.

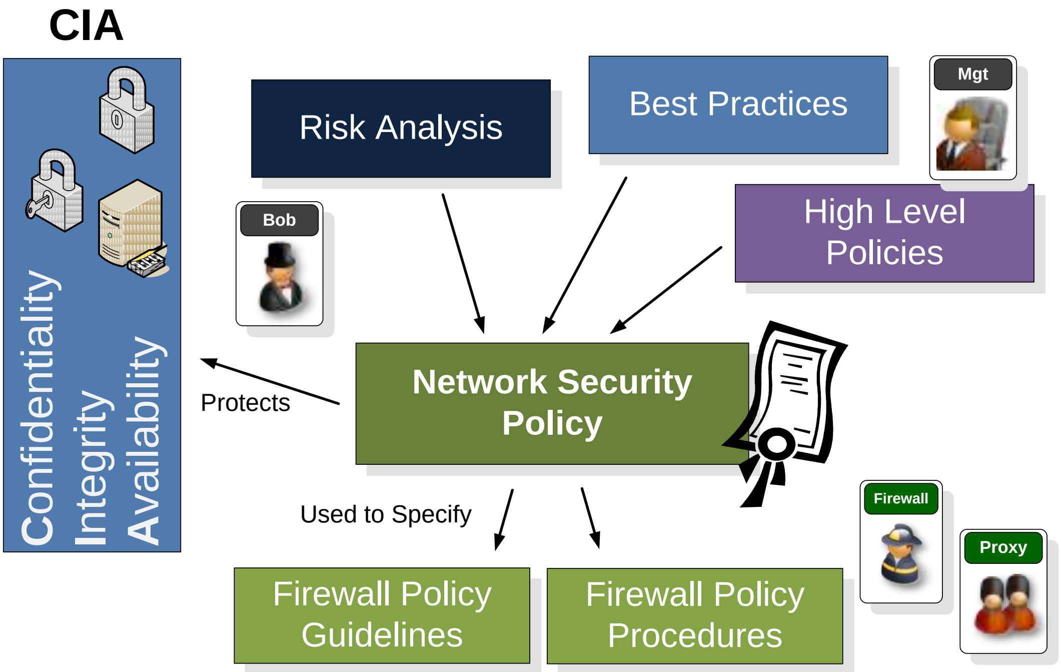


Figure 5 Firewall Policy

Guidelines for firewalls could include:

- The perimeter firewall should be the only point of entry from the untrusted Internet.
- The firewall should provide logging of traffic.
- The firewall should be a closed security posture.
- The firewall should provide stateful traffic filtering based on the rule sets defined in the Policy Procedures documents.

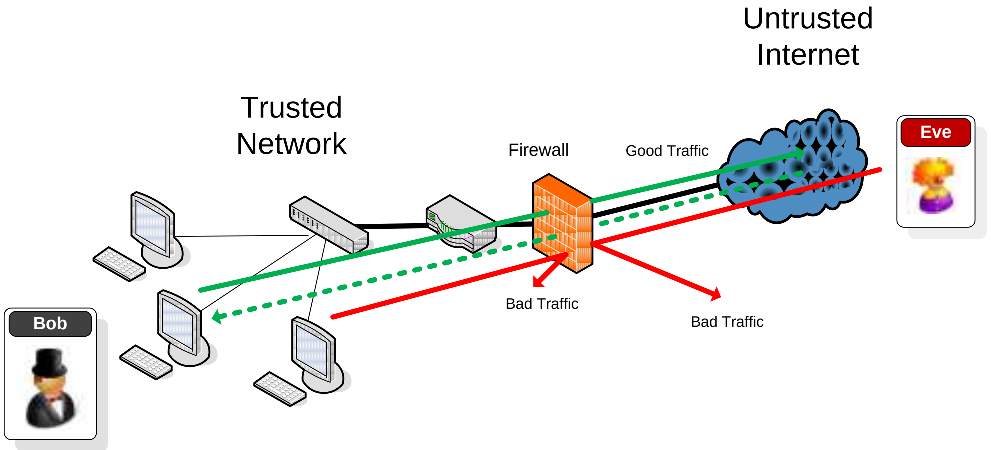
Procedures for Bobs network could include the following:

- Traffic allowed though the firewall, from the trusted private network to the untrusted public Internet:
  - HTTP – web traffic
  - HTTPS – encrypted web traffic

- Traffic allowed though the firewall, from the untrusted public Internet to the trusted inside network:
  - No traffic

This would allow web access to the Internet if the traffic is initiated from the trusted network – as a stateful firewall is being used, but no other traffic would be permitted in either direction.

In this simple example, the **trusted network** consists of the private network which is connected to the public Internet through the perimeter firewall. The only traffic which is allowed to pass through the firewall is the good traffic from Bobs subnet to the Internet, and any responding traffic. All other traffic, whether from Bob, or from the Internet is filtered out (blocked).



Trusted and Untrusted Networks

Trusted networks are within an organisation's control. It is not always this easy to define what is trusted and what is not. In a large organisation certain network segments may be more or less trusted, based on the assets they contain and their risk levels, such as a finance department, or a server farm which may contain more sensitive information than a user access segment. A classic example of this is an organisation's public facing services, such as web servers, which typically are moved forward onto a **demilitarized zone** (DMZ). This is a neutral zone, which is less trusted than the private internal network, but more trusted then the public Internet. This leads to internal network perimeters, where firewalls may be employed to segregate these, more and less, trusted networks. Internal firewalls, to separate network segments inside the trusted network, are common these days, as a large number of attacks – as much as 70% - may now come from insiders. This is an example of adding layers of security, known as **defence-in-depth**. Internal networks can be segregated by user groups, or by access to certain services. An example of defence in depth, using firewalls, is shown in Figure 6.

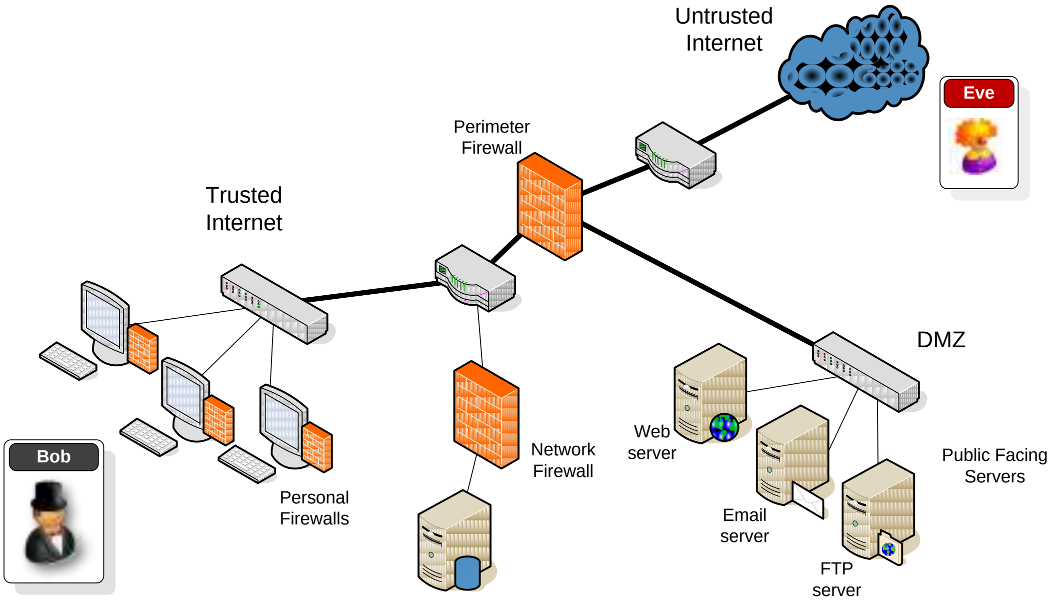


Figure 6 Defence in Depth using Firewalls

The **untrusted network** is typically the public Internet, as in the case of an organisations perimeter firewall. It is out of your control, as it is external to your security perimeter, even traffic coming from other parts of it

– like other branches of a business – are not trusted. The internet was originally used by a small number of Academics in the 1980's. It was a closed research community and was difficult to get access to, and so it didn't really require any security. The protocols and operating systems which were created at the time, such as the Unix OS and TCP/IP Protocol, did not need any built in security. In the 1990's, the use of the Internet had spread, and it was realised that commercial data had value, and needed to be secured. As more companies connected to the internet, the security vulnerabilities in the TCP/IP protocols and Unix OS were exploited. By the end of the 1990's millions of hosts were being connected and the threats to those connected grew. Over the years these external attackers gained expertise, and became a significant problem. In 1988 Robert Morris released the most famous Worm, which crashed around 6,000 computer. This was 10% of the machines on in the Internet. The Internet community of the time, was neither expecting, or ready to deal with the attack. In 2001 the Code Red worm infected 350,000 systems in less than 24 hours. This showed the potential for external attacks coming from the Internet, and with connected systems having gone from a thousand to more than a billion, there are many more potential intruders.

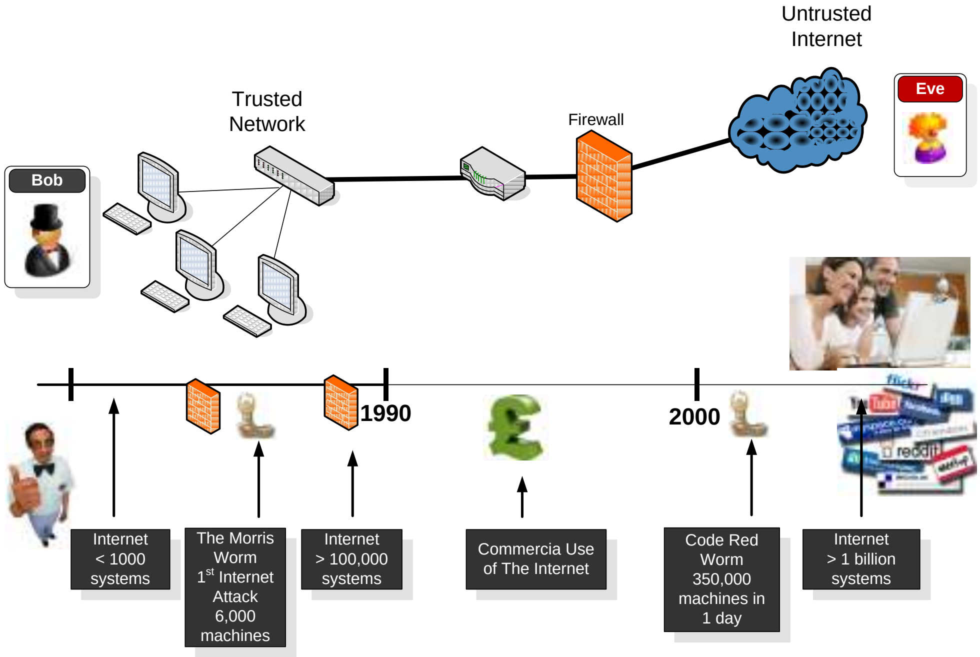
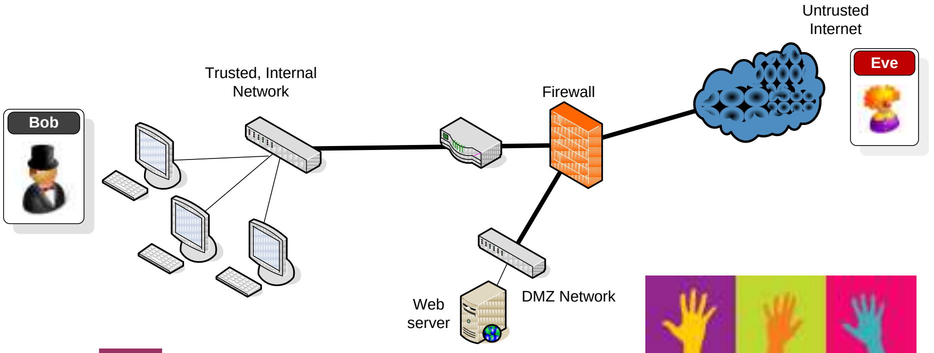

Untrusted Internet
Eve
Trusted Network
Firewall
Bob
1990
2000
Internet < 1000 systems
The Morris Worm 1st Internet Attack 6,000 machines
Internet > 100,000 systems
Commercia Use of The Internet
Code Red Worm 350,000 machines in 1 day
Internet > 1 billion systems

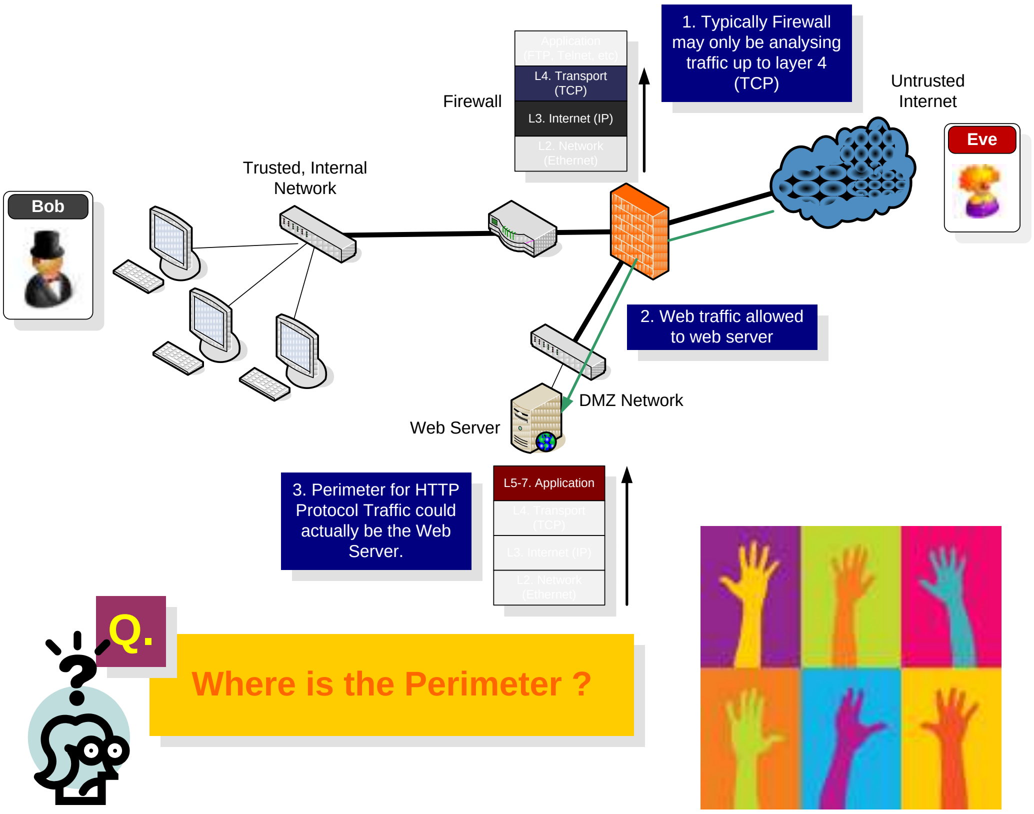
Figure 7 Internet History

Network Perimeters

When a firewall is deployed it is done so to enforce a security policy at a defensible boundary, called a network perimeter. **Question:** In the figure below, where is the network perimeter, at the firewall?

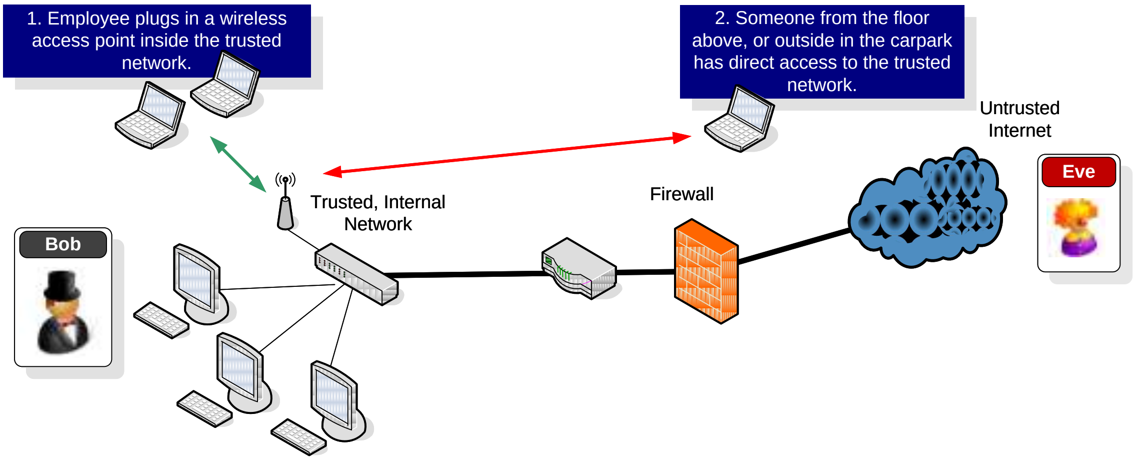


Answer: Actually it depends on the traffic! There are possible threats in the HTTP traffic between the Internet and the web server. Bobs policy allowed HTTP traffic from the Internet to the public web server. Some application protocols, such as Instant Messaging (IM) and peer to peer traffic can be encapsulated in HTTP packets. The firewall may only filter up to OSI Layers 3 & 4 – the Transport layers, and allows the HTTP traffic to the web server, not knowing what's inside. So in this case the perimeter for the HTTP traffic could actually be the web server.



Application (FTP, Telnet, etc)
L4. Transport (TCP)
L3. Internet (IP)
L2. Network (Ethernet)
Firewall
1. Typically Firewall may only be analysing traffic up to layer 4 (TCP)
Untrusted Internet
Eve
Trusted, Internal Network
Bob
2. Web traffic allowed to web server
DMZ Network
Web Server
3. Perimeter for HTTP Protocol Traffic could actually be the Web Server.
L5-7. Application
L4. Transport (TCP)
L3. Internet (IP)
L2. Network (Ethernet)
Q.
Where is the Perimeter ?

In the figure below, a wireless access-point has been plugged into a user access network segment. The access-point doesn't have any security, but this seems ok to the employee as it is inside the trusted network. If someone on the floor above, or worse still someone wardriving outside in the carpark outside, logs onto the access point, they have unauthorised access to the trusted network. The perimeter for wireless traffic is the access-point, and not the main firewall, as all the traffic is not being forced through the main firewall. The access-point would need to be installed with firewall software to address this problem (authentication should also be used). It is extremely important to know where the perimeters are in our networks, and traffic should be forced to cross them at **choke points,** where security – such as firewalls - can be focused.

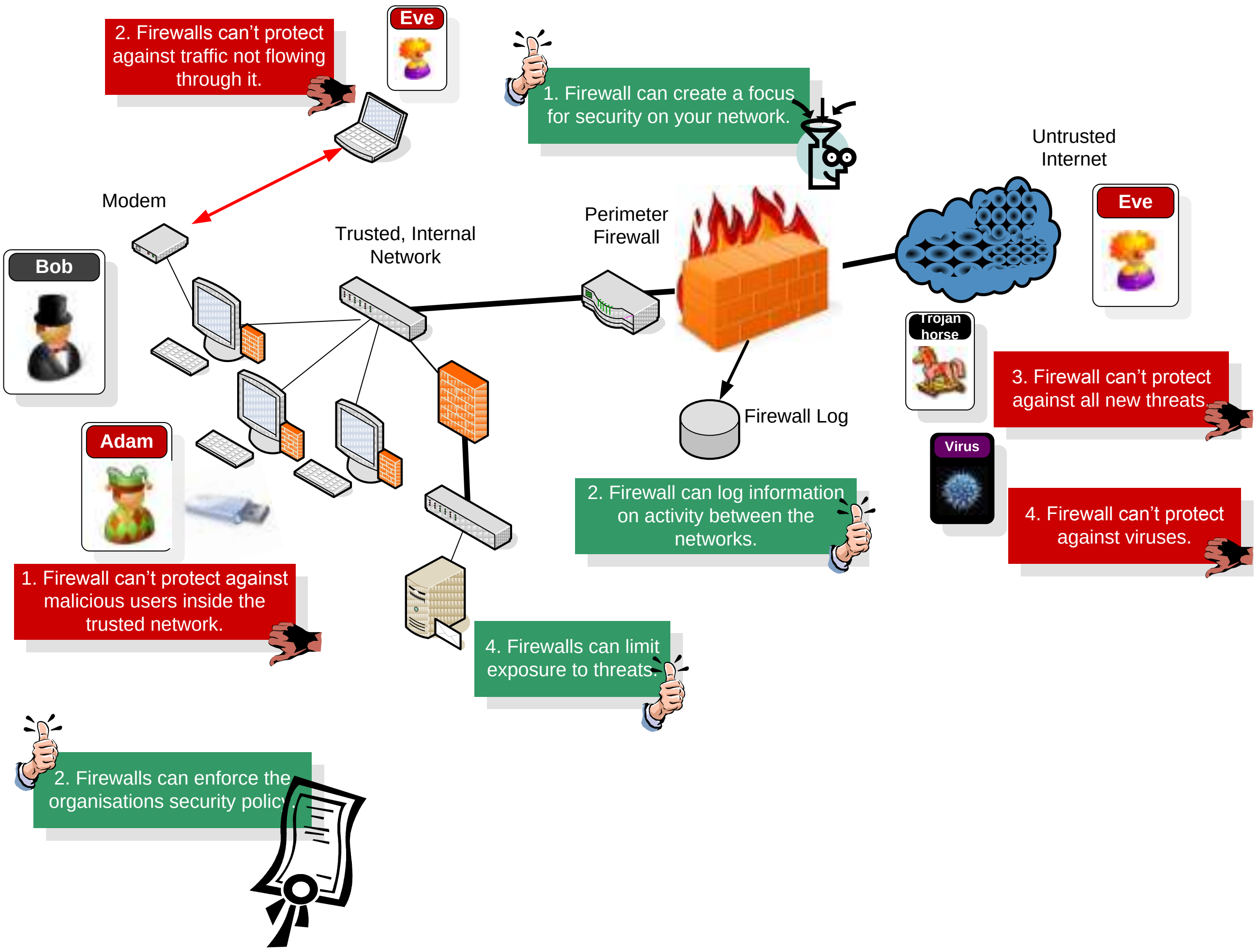


1. Employee plugs in a wireless access point inside the trusted network.
2. Someone from the floor above, or outside in the carpark has direct access to the trusted network.
Untrusted Internet
Eve
Trusted, Internal Network
Firewall
Bob

What firewalls Can & Can't Do

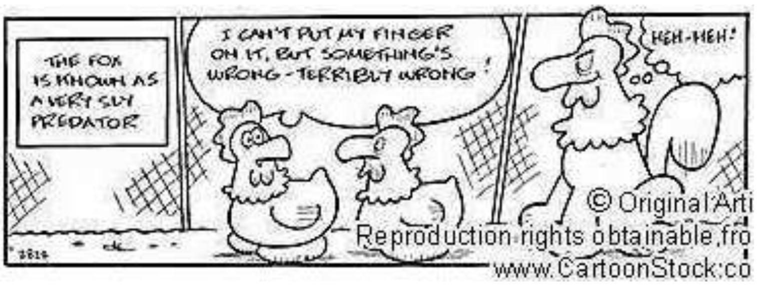
Firewalls Can

1. **Firewalls can be a focus for security in the network.** Where all traffic flows through, sometimes called a choke point. Leverage network security here, even though it may cost a lot for enterprise firewall, it is more appropriate and less expensive than trying to implement on all the separate devices and end user hosts and servers.
2. **Firewalls can enforce the network security policy.** A Firewall is a sentry, between two networks, only allowing approved traffic between the networks. These are configured based on the organisations security policies. For example, one organisation may not allow any services inbound from the internet at all, but another may let a certain set of services into the DMZ, and none into the internal network.
3. **Firewalls can log traffic activity between the networks.** This is a good place to log the use, as well as the misuse of the network, as all traffic should pass through the firewall. Note: Logging can affect performance.
4. **Firewalls can limit exposure to threats.** If an attack, such as a worm propagating, is on one network, it should be contained there by the firewall at that networks perimeter. For example internal firewalls, separating critical servers from the rest of internal network, would mitigate malicious software spreading from users machines to these important servers.


2. Firewalls can't protect against traffic not flowing through it.
Eve
1. Firewall can create a focus for security on your network.
Untrusted Internet
Modem
Trusted, Internal Network
Perimeter Firewall
Eve
Bob
Trojan horse
3. Firewall can't protect against all new threats.
Firewall Log
Adam
Virus
2. Firewall can log information on activity between the networks.
4. Firewall can't protect against viruses.
1. Firewall can't protect against malicious users inside the trusted network.
4. Firewalls can limit exposure to threats.
2. Firewalls can enforce the organisations security polic

Firewalls Can't

1. **Firewalls can't protect against malicious insiders.** Once the Fox is in the hen house there is very little the firewall can do. It can stop the user transferring stolen data out of the system, if it goes through the firewall, but the user can simply write it to a memory stick and walk out the front door.


THE FOX IS KNOWN AS A VERY SLY PREDATOR
I CAN'T PUT MY FINGER ON IT, BUT SOMETHING'S WRONG - TERRIBLY WRONG!
HEH-HEH!
© Original Arti
Reproduction rights obtainable fro
www.CartoonStock.co

A firewall is like a castle wall and the guards at its gate. It is there to stop intruders getting in, but it is useless against an attack from within. Security guru Bill Cheswick described a network system protected by a perimeter firewall as "a sort of crunchy shell around a soft, chewy center" (1).

2. **Firewalls can't protect against traffic not flowing through them.** For example dialling into the trusted network, via a modem behind firewall. Sometimes, a system administrator, or even developers will set up their own way into a system, a **backdoor**, bypassing the perimeter security, as they regard it as too restrictive. This was made famous by the 1983 film **WarGames** (1), were a teenager hacks into the the US Department of Defence (DoD) war simulation computer - via a backdoor - and very nearly starts world war III. The film was very influential in the hacking community inspiring **War Dialing** and **War Driving**. War Dialing is a reconnaissance technique were a modem automatically dials phone numbers looking for computer systems. This might be followed by an escalation attack, like password cracking (in the film the backdoors owner, *Dr Falcon*, uses a poor password). The War Dialling technique is still popular, and effective today (2) For more see (3). War Driving is a similar search for systems to break into, but by driving around in vehicle looking for wireless access points to connect to and attack.

3. **Firewalls can't always protect against new threats.** A firewall should only let specified services through the firewall, and block all others (a closed system). New threats in permitted services are being developed all the time, such as protocols being tunnelled within other protocols.
4. **Firewalls are not good at protecting against viruses.** Some virus protection can be provided by some firewalls, but in general firewalls won't be able to recognise programs within traffic.
5. **Firewalls can't configure themselves.** Purchasing an enterprise firewall is not the end of the story. Every organisation and network is different, and the firewall will not provide good protection straight from the box. That's were security experts come in.

Firewalls can not provide the complete solution. Certain threats are not within the control of the firewall. Physical security, end host security and user guidance are also needed to create an overall security implementation.

4.3 DMZs and Firewall Topologies

Network firewalls can be deployed in various different network topologies depending on the requirements of the organisation and its users. In Figure 8, Bob's organisation has a typical topology. There is the untrusted public Internet - with a trust level of 0% – no trust, the DMZ network - with a trust level of 50% – not very trusted, the Internal Networks – trust 90%-100% – the most trusted. The Servers network segment is at a higher trust level as it has been found to contain more critical data than the user access segment, during the risk analysis.

Demilitarised Zone (DMZ)

In the real world the Demilitarised Zone (DMZ) is a no-mans land between North and South Korea which remains neutral. In network terms, the DMZ is a neutral zone between the inside private network, and the outside public network. It is the place where public facing servers can be deployed. It provides another layer of security. For example, if the public facing web server is compromised the internal network is likely to be still secure.

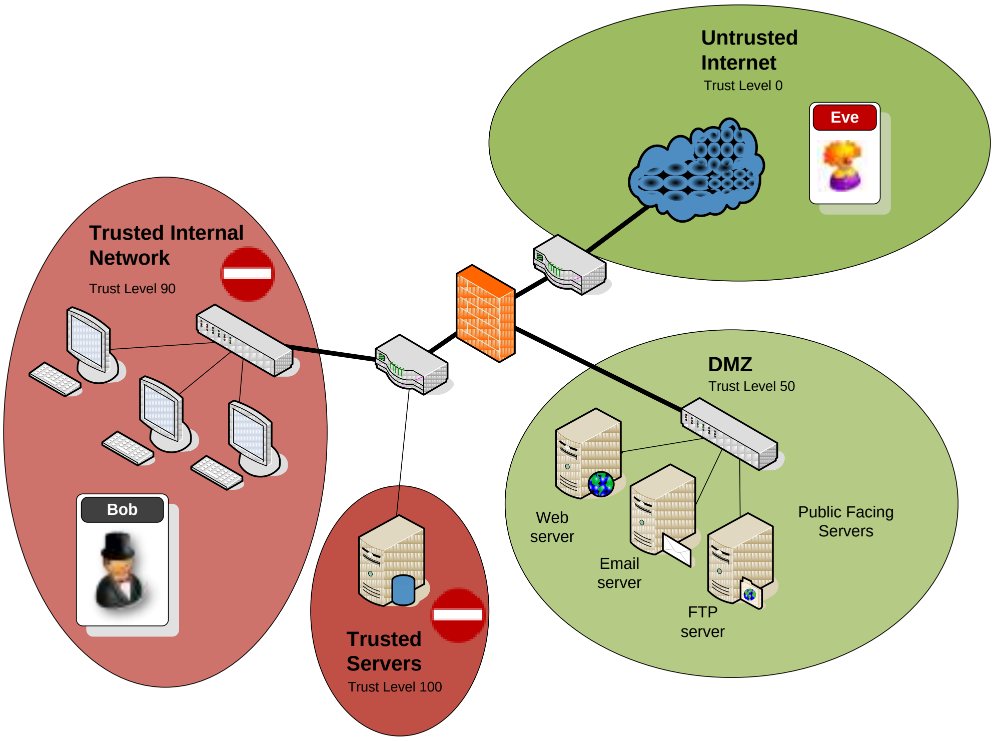


Figure 8 Bobs Network Firewall Topology

This has another castle analogy, as castle were often built with inner and outer walls, with the important buildings and people within the inner walls and in the central building known as the keep. If the outer walls were overrun, the defending soldiers would fall back to the inner walls, and then to the keep. A DMZ needs two logical firewalls, one between the outer network and the DMZ, and one between the DMZ and the inner network. This is often now done within one physical firewall with three interfaces, as shown in Figure 8.

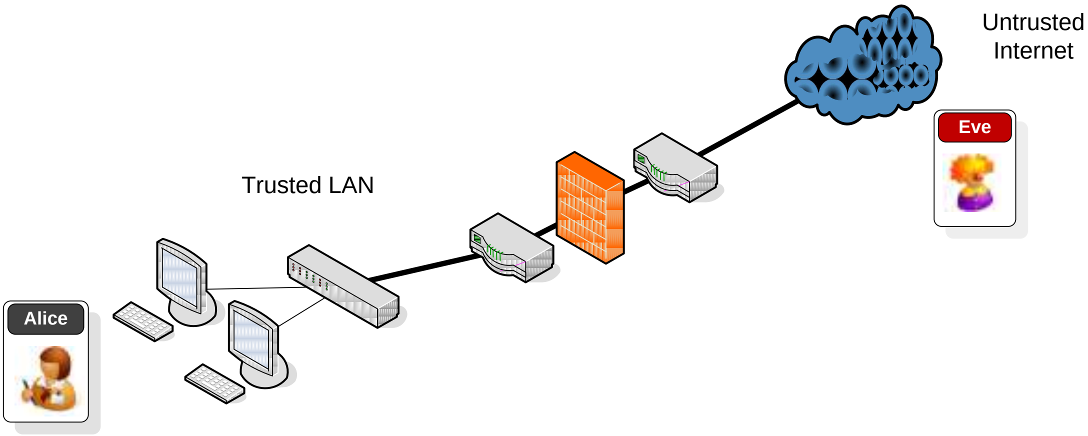
Steps to decide the design of Firewall Topology and DMZ(s) to Use:

1. Analyse the existing network infrastructure, and the current security – network mapping and analysis of the operating systems and applications being used.
2. Identify risk levels to our various assets and services through a risk analysis and then from that, the different zones of security which are needed, and their trust levels.
3. Identify the perimeters and the choke points between zones.
4. Design perimeter topologies based on the findings.

Firewall Topologies

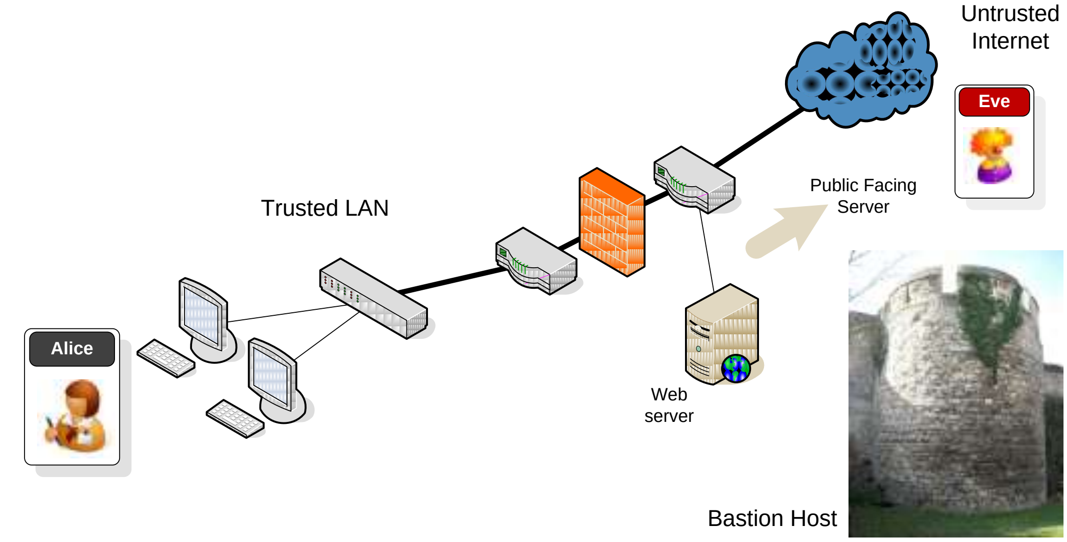
Single Firewall

This would be used to protect a small business or home network, in this case Alice's. Alice has no need to provide external services, so a DMZ is not necessary. This could also be used for segregating internal networks, such as protecting a server farm, or a development subnet which has more sensitive data to protect.



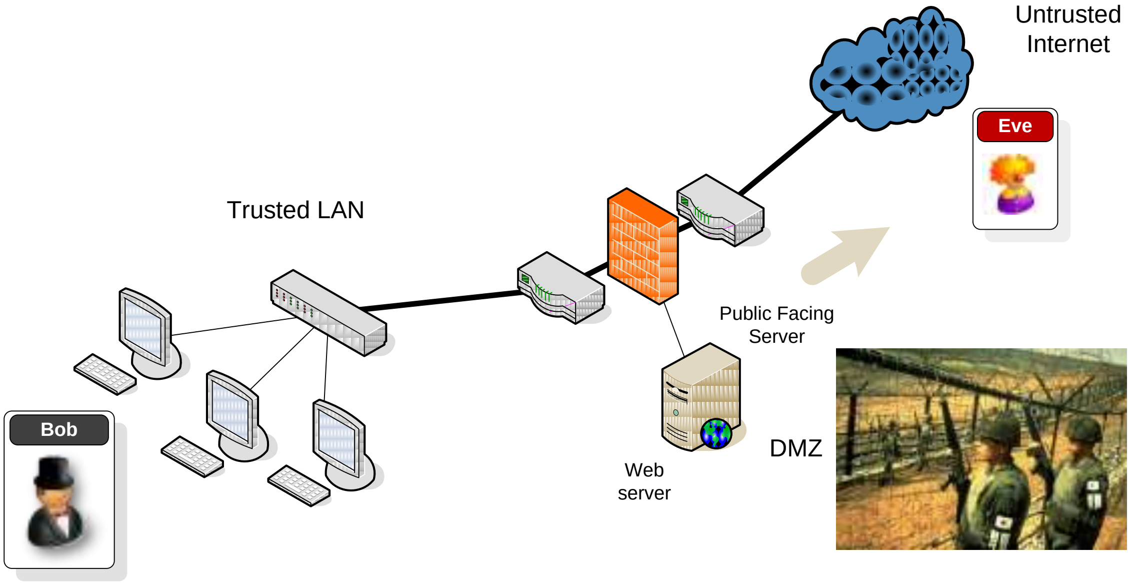
Single Firewall with a Public Facing Server (Bastion Host)

This could be used to provide public services to the untrusted public network, with the web server located outside the trusted network, in front of the firewall. The public facing server is sometimes called a **bastion host** as it is exposed to attack like a bastion in a castle wall. A bastion host will be configured differently from other servers. It would be hardened against attacks, removing all unnecessary services, protocols, and applications. The security has to be done on the host itself as it has nothing to protect it. This design might be used to provide static services to the public network such as a web presence, but not e-Commerce, or anything which would need updated from the trusted network. It is important that it does not share authentication and authorisation with the internal hosts, user accounts, which could be used to access the internal network. This is to prevent, if compromised, the attacker performing a trust exploitation attack and get access to the internal network. i.e. the attacker doesn't get the keys to the castle!



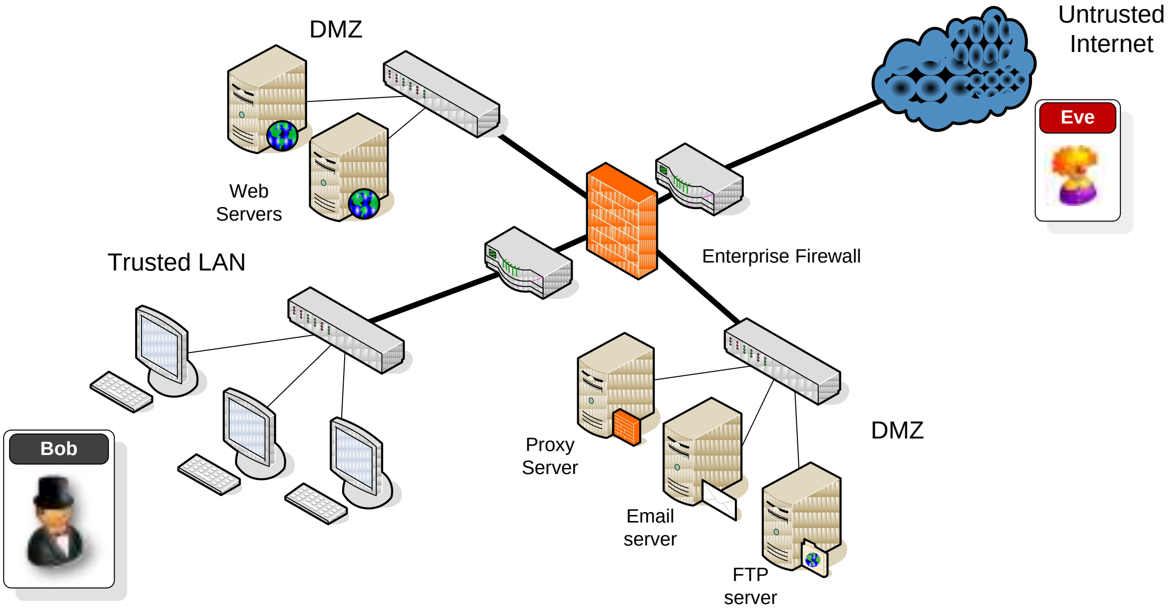
Single Firewall with Public Server in a DMZ

This would be used to provide services to the public network, with the public servers in a DMZ network. The firewall protects the public server from the Internet, and the internal network from the public server. In this case only web traffic would be allowed from the Internet into the web server on the DMZ. This would be used if dynamic public services are to be provided. For example, the servers might need access and updates from the internal network. If multiple services are to be provided, it is good practice for them to be on different servers in the DMZ. For example, if the FTP server is compromised, the web server isn't compromised along with it.



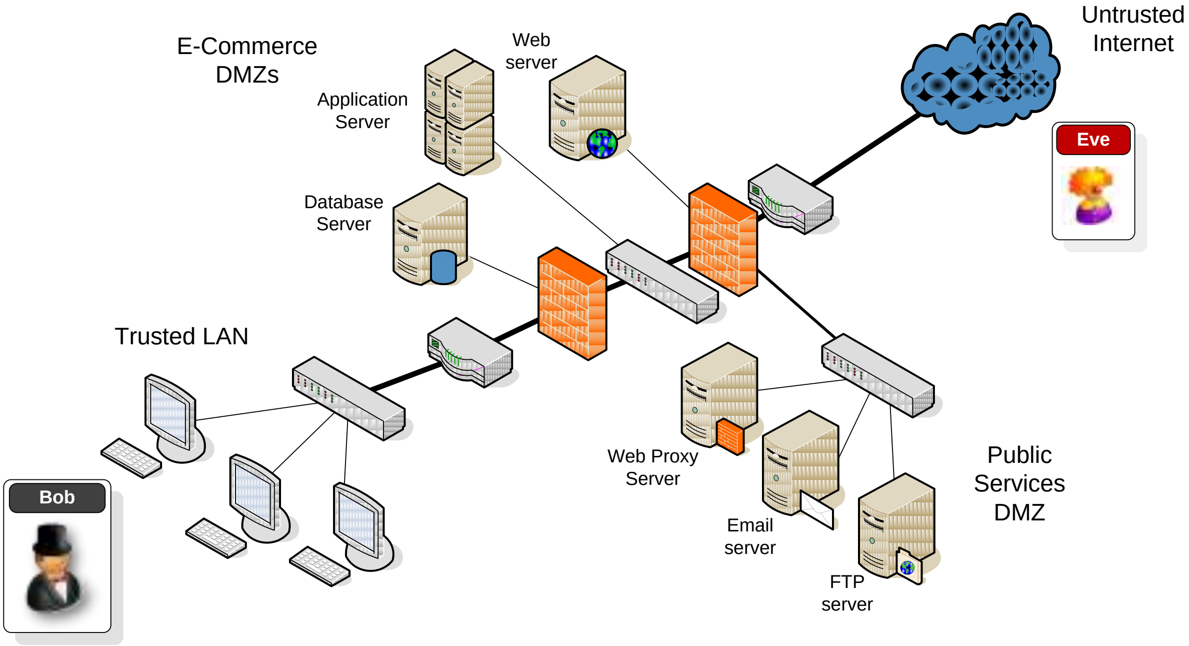
Enterprise Firewall with Multiple DMZs

This might be used to provide services to the public network, with the public servers in several DMZ networks, using an enterprise level firewall with multiple interfaces. Again this is to prevent trust exploitation tactics from one server to another, or from a server internal network. An example of a firewall product which could provide this type of multiple interface firewalling would be a Cisco or Juniper firewall appliance.



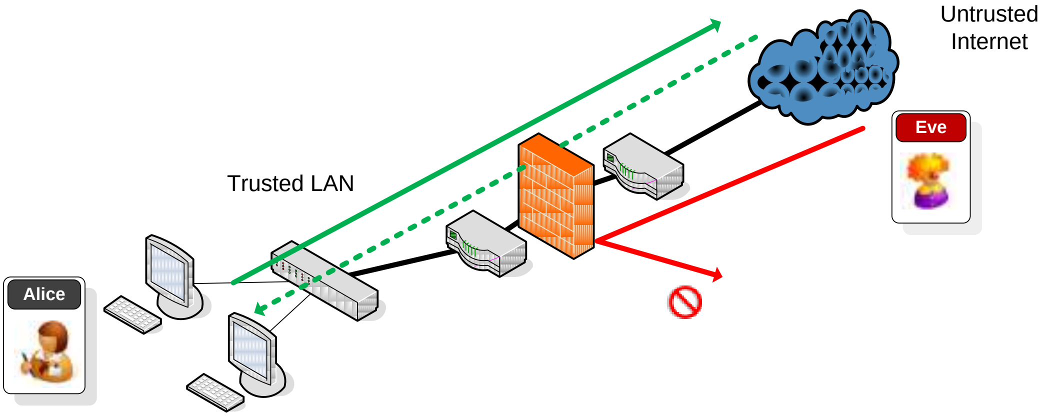
Dual Firewalls with multiple DMZs

This could be used to provide services to the public network, with the public servers, in various different DMZ networks. In the example below, three different security zones are created by the two firewalls. This topology might be used if access was needed to the public servers from inside the trusted network, as well as Internet access, and access between the different types of servers. For example an e-commerce setup, with web servers, application servers, and database servers, which need to communicate with each other, but need different levels of protection. The public network would access the web servers, the web servers would then access the application servers, which could have access to the database servers in a three tier setup. Each access would be through a firewall with extra layered defence by having two different physical firewalls. Two firewalls also gives an extra level of protection, as even if one of the firewalls was to be compromised, the other firewall would still protect the more important data. This is a typical approach used by organisations providing e-Commerce services.



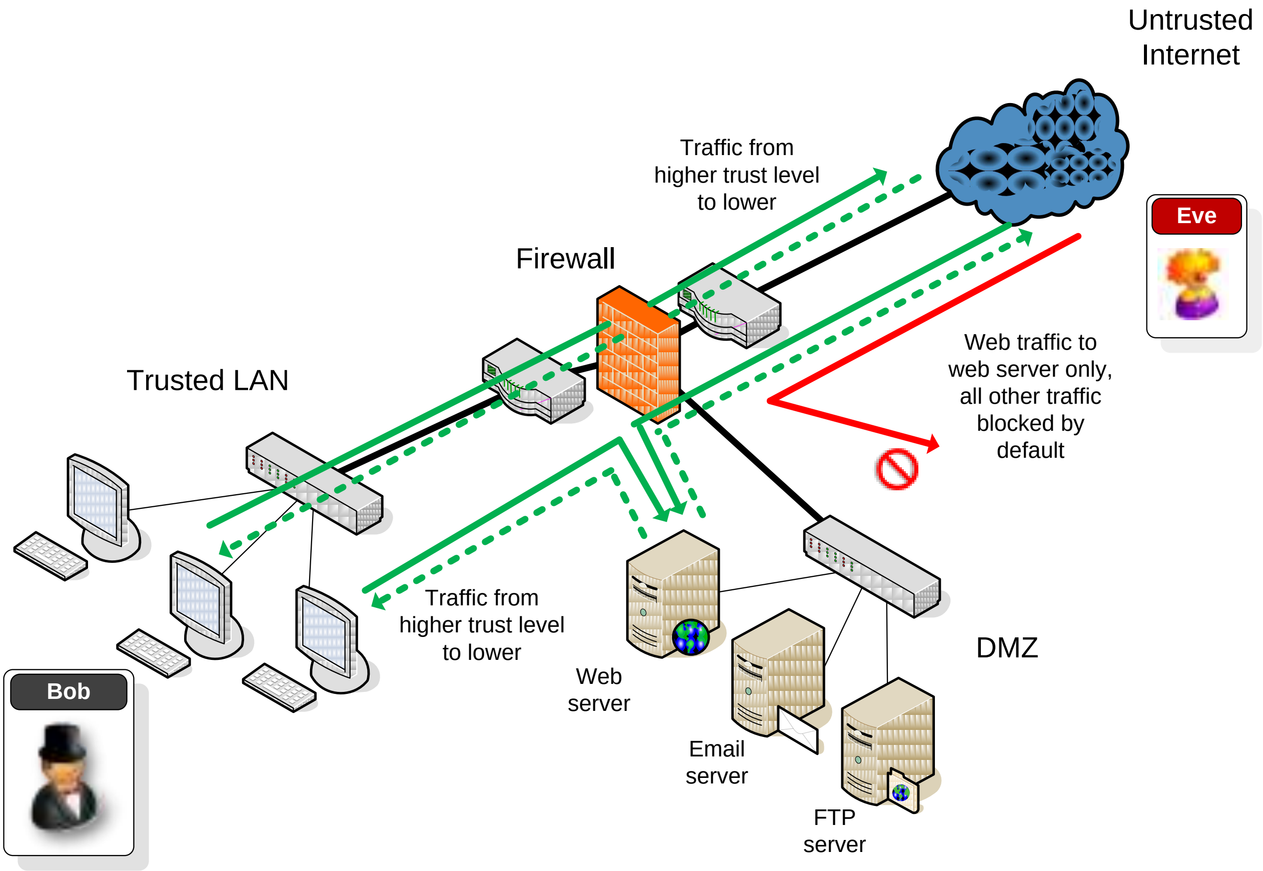
Single Firewall Traffic Flow

Going back to the first of the topologies, the basic single firewall, separating two networks. The firewall allows all traffic outbound. This is because traffic is flowing from a higher trust zone (trusted internal network) down to a lower trust zone (the Internet). Inbound connections which did not originate inside the trusted network will be dropped. Inbound traffic which is part of connections initiated on the trusted network is allowed (shown be dotted line). This is typical of the default behaviour of DSL/Cable router firewalls, or Personal software firewalls.



Single Firewall with a Public Server in a DMZ - Traffic Flow

This would typically have the same unrestricted outbound traffic flow, from the trusted network to the Internet by default, as well as associated return traffic would be allowed. Again this is because the traffic is flowing from a higher trust level to a lower one. Specific traffic to our forward facing servers would be allowed from the Internet through to the DMZ. This traffic would have to be the correct type of traffic for the specific server. For example, traffic to the web server would have to be HTTP, or HTTPS traffic, or it would be dropped at the firewall. As this traffic flows from a lower level security zone to a higher level zone, firewall rules to allow this would need to be configured on the firewall. Traffic from the trusted internal network to the DMZ is also allowed as it is from a higher security level to a lower one.

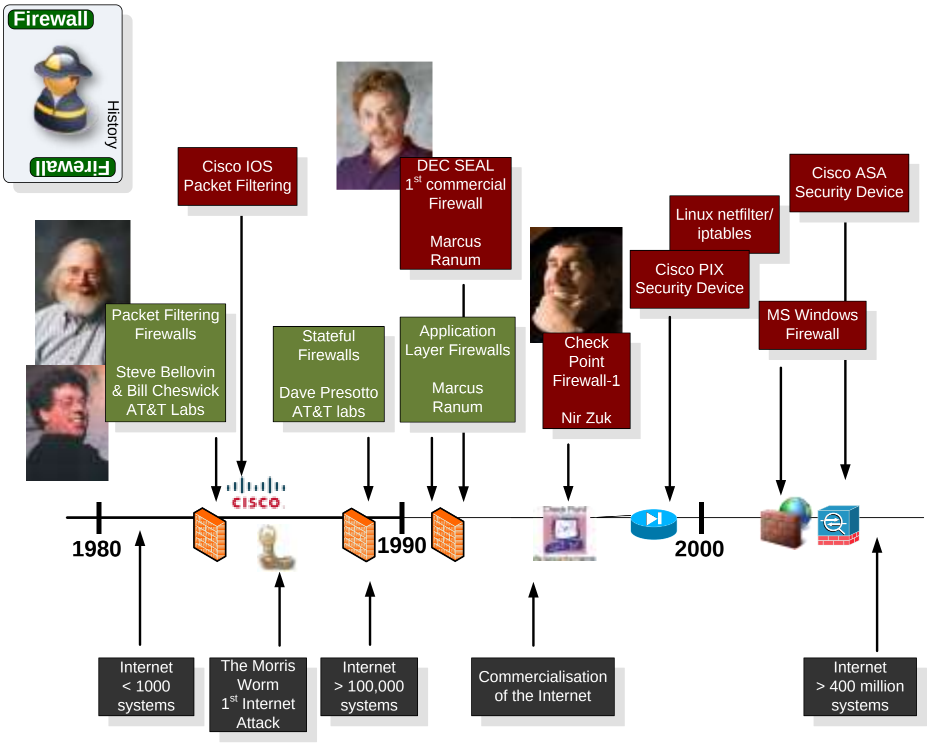

Untrusted Internet
Eve
Traffic from higher trust level to lower
Firewall
Trusted LAN
Web traffic to web server only, all other traffic blocked by default
Traffic from higher trust level to lower
Web server
Email server
FTP server
DMZ
Bob

4.4 Firewall Architectures

History

Firewall technology has been around since the mid 1980's, almost as long as routers, but techniques have developed rapidly. As the Internet grew, and changed from a closed group of researchers to millions of systems, its commercial use also grew. Firewalls have had to evolve, to protect commercial data, and in response to new attack techniques and performance demands. In the Mid to late 1980s, there was work on traffic filtering at AT&T labs by **Bill Cheswick** and **Steve Bellovin**, as well as at DEC by **Jeff Mogul**. These were the pioneers of firewalling, and developed the first generation of firewall technology. This was called **packet filtering**, and Cisco incorporated this filtering into some of its routers not long after. In the late 1980s work, again by Bellovin and Cheswick at AT&T labs, was done to produce the first stateful firewall technology - the circuit level firewall. Packet filtering individual packets, but also checking the context, in terms of being part of a valid TCP connection.

In the early 1990s the first commercial firewall DEC SEAL was created by **Marcus Ranum**. Also, various stateful firewall research continued, and dynamic packet filtering was developed. At the same time, the first **Application layer firewall** was created by Marcus Ranum and Bill Cheswick at AT&T labs. Their firewall acted as an **application proxy**, to make the server invisible to the remote user, and to perform application layer filtering. **Checkpoint**, a software firewall (3), was the first commercial firewall, based on the stateful packet filtering research. As stated on their website: "Check Point pioneered and patented Stateful Inspection. U.S. Patent # 5,606,668, issued on February 25, 1997, covers, among other things, Check Point Software's implementation of "Stateful Inspection" technology for controlling network traffic". In the late 90s Cisco released its **Cisco PIX firewall**, a market leading firewall and all round security appliance, which has stateful packet filtering, application layer filtering, as well as NAT and VPN termination functionality. **Microsoft** first shipped **Windows Firewall**, a basic stateful personal firewall, with its XP OS - SP2, but it is a basic stateful firewall and only filters incoming traffic, providing no protecting from malicious processes sending traffic outbound from a system. The latest **Windows Firewall with Advanced Security** which is part of the Vista OS, is a much improved, and highly configurable, stateful firewall.


Firewall
History
Firewall
Cisco IOS Packet Filtering
DEC SEAL 1st commercial Firewall
Marcus Ranum
Cisco ASA Security Device
Linux netfilter/ iptables
Cisco PIX Security Device
MS Windows Firewall
Packet Filtering Firewalls
Steve Bellovin & Bill Cheswick AT&T Labs
Stateful Firewalls
Dave Presotto AT&T labs
Application Layer Firewalls
Marcus Ranum
Check Point Firewall-1
Nir Zuk
CISCO
1980
1990
2000
Internet < 1000 systems
The Morris Worm 1st Internet Attack
Internet > 100,000 systems
Commercialisation of the Internet
Internet > 400 million systems

Firewall Architectures Overview

Several types of firewall architectures are mentioned in the history section. We can compare the technologies by looking at the layers of the OSI and TCP/IP model that each firewall technology can filter on. The addresses contained in the **Physical Layers** are concerned with the physical addresses of devices on the local network, the **Network Layer** use virtual network addresses called IP Addresses to route traffic across networks to the traffics destination, the **Transport Layer** deals with application communication sessions between two connecting hosts, and finally the **Application Layer** defines an application process – an instance of an application running on a host machine - and the data for that specific application.

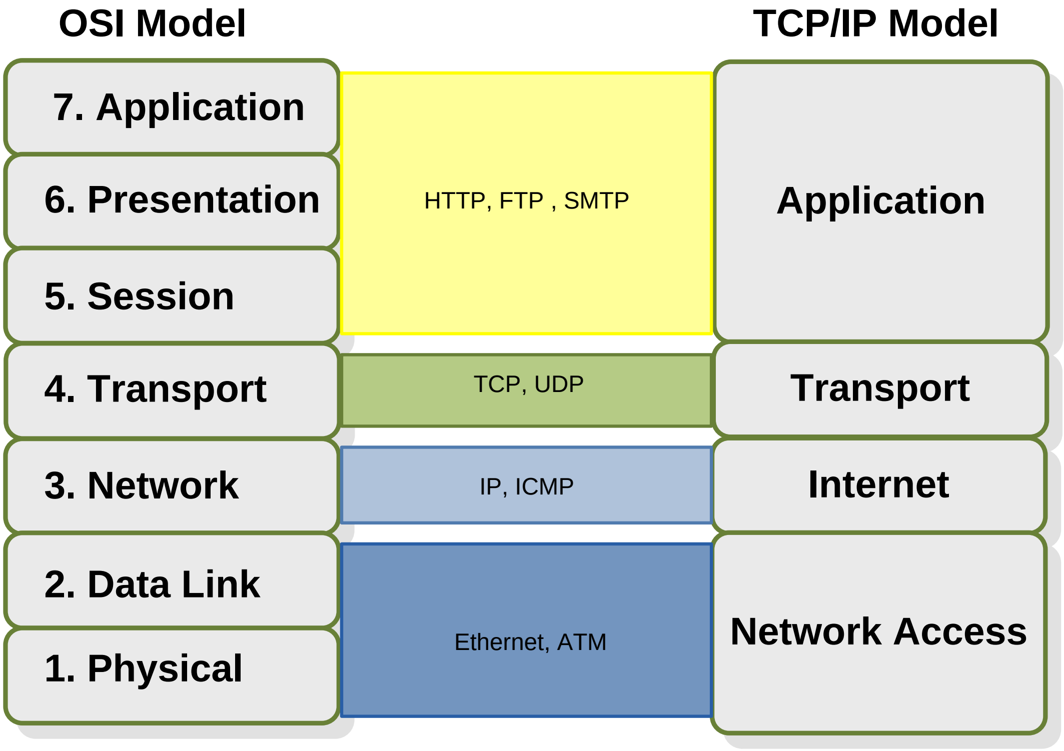


Figure 9 OSI and TCP/IP Internet Models

Basic firewalls work at the lower levels, typically Layers 3 and 4, while more advanced firewalls can filter at all of the layers, and the protocols in each. Firewalls which can only work at the lower layers can only filter based on Physical or Network addresses, and not specific applications, or users. A firewall which can analyse traffic at the Application layer, and filter on the application data, can enforce user authentication, as well as filtering on application specific data.

Packet Filtering

A basic type of firewall is a Packet Filtering firewall, which could be described as a stateless filtering and routing device. Firewalls are always routers, which just happen to also perform filtering of the traffic which they route. Stateless packet filters perform their filtering based on a set of filtering rules, known as a ruleset. They compare the rules with the information in the packet to make a decision on whether to route the packet on to its destination or filter out the packet (drop the packet).

Data Encapsulation/De-capsulation

If Alice is browsing the web, she sends and receives traffic to and from a web browser. At Alice’s end of the communication channel the Application Data is encapsulated, with a header added for each layer of the TCP/IP protocol model. It is then sent over the physical media to its destination network. When it reaches the firewall the message is de-capsulated up to the Layer 4, the Layer headers are popped off. All routers – including firewalls - need to do this to identify the packets destination network IP Address, so the packet can be routed to the correct network. The firewall can then perform packet filtering based on the IP and TCP layer packet header information. If the packet is passed by the firewall filtering rules, it is then routed to the appropriate connected network. The firewall compares the header information against its firewall rules to check whether this type of traffic is to be passed or dropped.

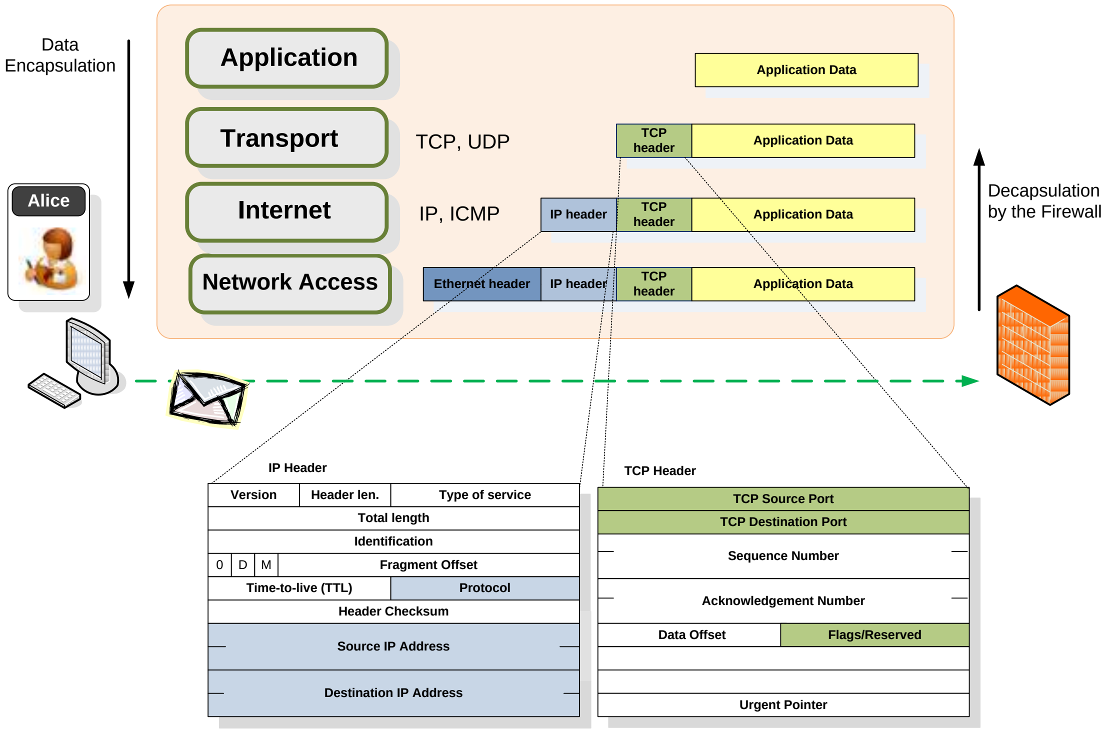


Figure 10 Packet Filtering at OSI Layers 3 and 4

IP & TCP Headers

Layer 3, the Network Layer

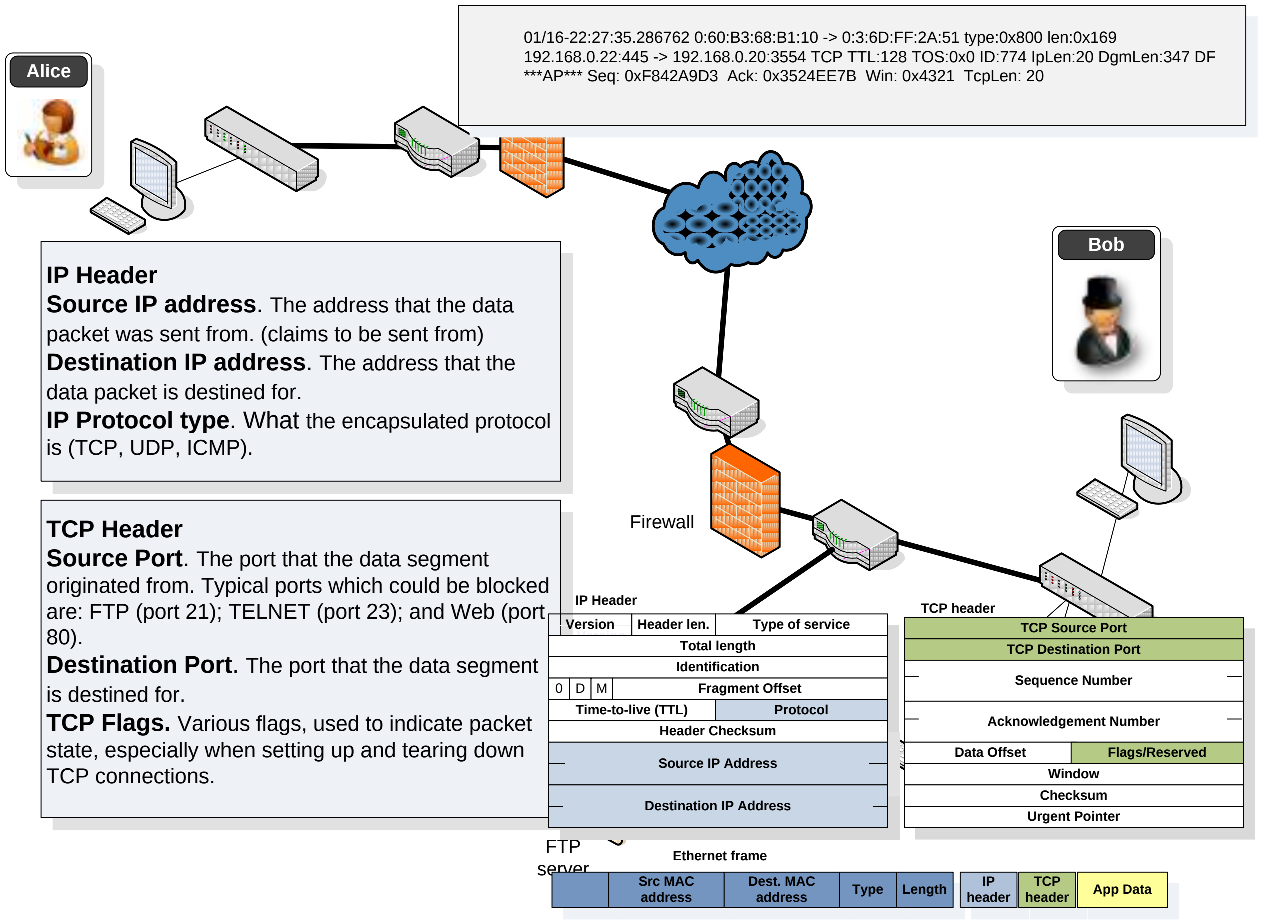
The most basic packet filtering would be done at Layer 3, the Network Layer (or Internet Layer in the TCP/IP Model). The IP Header is used in Packet Filtering as it contains the source and destination network addresses – the **IP Addresses** - for the hosts which are communicating. Typically we would filter on the:

- **Source IP Address** Network address the packet originates from. In this case Alices network address, such as 192.168.2.1
- **Destination IP Address** Final destination network address. In this case, Bobs web server, such as 146.182.1.10
- **Protocol field** The transport protocol, encapsulated within the IP packet, being used for communication, such as TCP or UDP.

Layer 4, the Transport Layer

If the Protocol is TCP or UDP, filtering can be done on the Layer 4, the Transport Layer header data. The Source and Destination Ports from the TCP header can be used to filter the communication session details. The Layer 4 header does not have any information on the source or destination of the traffic, but identifies the process or service the data is for. The Protocol, from the IP Header, and the Port together make up the Service. For example TCP and Port 80 identifies the HTTP service used for web traffic.

- **Source/Destination Port** In a client server model, the server always has a well known port number, such as 80 for web server running HTTP. So when the client connects to the server the destination port of the first packet is always the well known port number, and the client port is usually a random port above 1024. So in this case the packet might have a dynamic source port of 1025 (not well known), and a destination port of 80 (Bob’s web server).



```
01/16-22:27:35.286762 0:60:B3:68:B1:10 -> 0:3:6D:FF:2A:51 type:0x800 len:0x169
192.168.0.22:445 -> 192.168.0.20:3554 TCP TTL:128 TOS:0x0 ID:774 IpLen:20 DgmLen:347 DF
***AP*** Seq: 0xF842A9D3  Ack: 0x3524EE7B  Win: 0x4321  TcpLen: 20
```

IP Header
Source IP address. The address that the data packet was sent from. (claims to be sent from)
Destination IP address. The address that the data packet is destined for.
IP Protocol type. What the encapsulated protocol is (TCP, UDP, ICMP).

TCP Header
Source Port. The port that the data segment originated from. Typical ports which could be blocked are: FTP (port 21); TELNET (port 23); and Web (port 80).
Destination Port. The port that the data segment is destined for.
TCP Flags. Various flags, used to indicate packet state, especially when setting up and tearing down TCP connections.

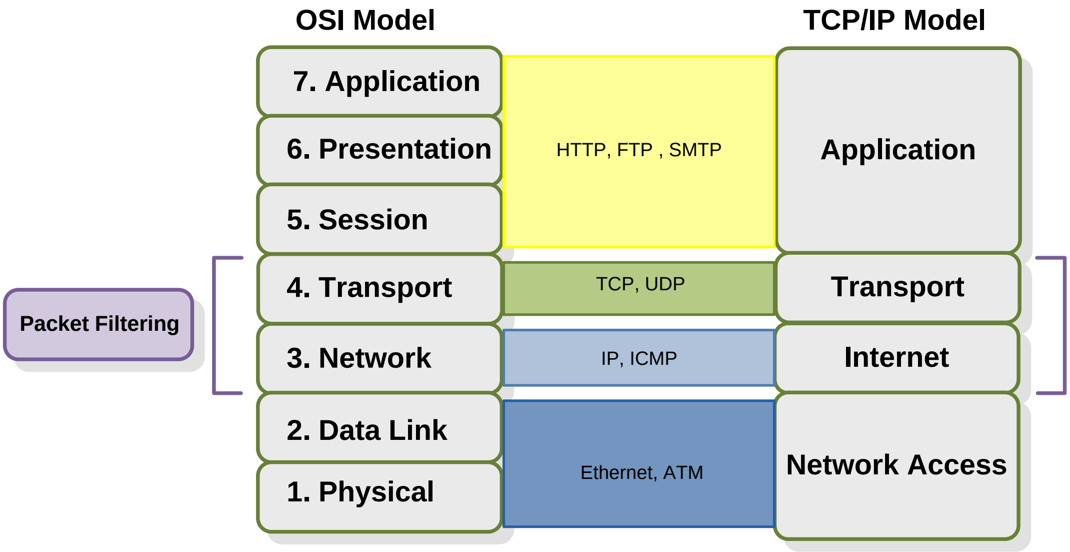
So in this way port numbers can be used to block, and allow services. Port numbers have a range of 0 to 65,536. Well known protocols use the same port numbers every time, ports 0-1024, and dynamic ports are above 1024. A range of well known Protocols and their ports are shown in Table 2 below. The Internet Assigned Numbers Authority (IANA) are responsible for assigning port numbers, and for a full list of port numbers and protocols refer to their web site (4).

Table 2 Common Protocols and Port Numbers

Port Number	Protocol
20	File Transfer Protocol (FTP) – Data
21	File Transfer Protocol (FTP) – Control
23	Telnet
25	Simple Mail Transfer Protocol (SMTP)
53	Domain Name Services (DNS)
80	Hypertext Transfer Protocol (HTTP)
110	Post Office Protocol (POP)
161	Simple Network Management Protocol (SNMP)

Packet Filtering Firewalls

The firewall inspects packets as they enter an interface (inbound or outbound). Each packets header information at layers 3 & 4 (IP & TCP headers) is decapsulated, and compared with the filtering ruleset to decide if the packet can be routed or is to be dropped.



Each packet is dealt with individually, there is no context – or state- taken into account. This is known as **static packet filtering**, and the process is done exactly the same way for each packet, there is no concept of a communication session, a TCP connection, how it is built, the data transfer, or the teardown.

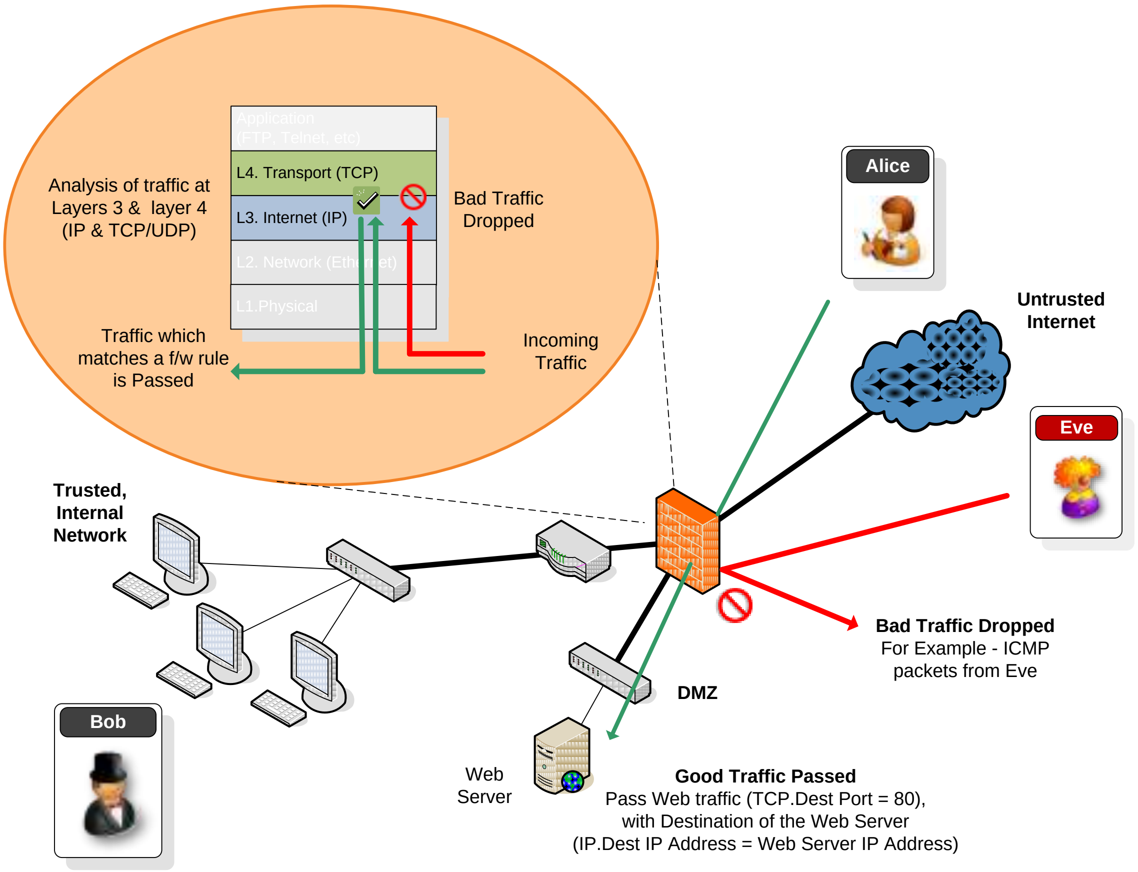

Application
(FTP, Telnet, etc)
L4. Transport (TCP)
L3. Internet (IP)
L2. Network (Ethernet)
L1.Physical
Analysis of traffic at Layers 3 & layer 4 (IP & TCP/UDP)
Bad Traffic Dropped
Traffic which matches a f/w rule is Passed
Incoming Traffic
Alice
Untrusted Internet
Eve
Trusted, Internal Network
Bob
Bad Traffic Dropped
For Example - ICMP packets from Eve
DMZ
Web Server
Good Traffic Passed
Pass Web traffic (TCP.Dest Port = 80), with Destination of the Web Server
(IP.Dest IP Address = Web Server IP Address)

Figure 11 Packet Filtering Firewall

Firewalls which only use packet filtering, and ignore the higher layers have one main strength - they are **very fast**. Because of this, they are typically used as a first line of defence, deployed on a boundary router. They are often used to perform bulk filtering in front of a firewall, such as IP Spoofing - RFC 2827 and 1918 – filtering, or mitigation of reconnaissance, for example filtering ICMP traffic. The higher level filtering can be done by another firewall behind the boundary router - another example of defence in depth. Of course a very fast high end firewall may be able to cope with all traffic, and a boundary router may not be needed in this case.

Packet filtering firewalls are also very **scalable** as they work below and are independent of high level protocols such as user application protocols. They are **flexible**, because they work on the lower levels of the layered models, they can provide filtering for almost any network based protocol.

The main drawback to packet filtering firewalls is that they **cannot provide the full range of security** filtering that is required in networks today. Because they do not look at the upper layers data, they are unable to prevent attacks on application based vulnerabilities. They also cannot provide any sort of user authentication as they are not able to check that higher level of data.

Most operating systems and routing devices have packet filtering built in. Unix and Linux both have advanced packet filtering firewalls. Windows has a more basic packet filtering firewall, although the new Vista version has much improved features.

Packet Filtering Process

The firewall searches through a list of **firewall rules** (a ruleset) to determine what to do with each packet. The typical actions are **Pass** the packet onto its destination (i.e. route it) or **Drop** the packet at the firewall. **Reject**ing the packet, by sending an informational packet back to the client, informing them that the packer was dropped, is another common option. Also, with most firewalls, it is usually possible to **Log** the action taken along with the packets information. This can then be analysed to identify attacks or problematic traffic.

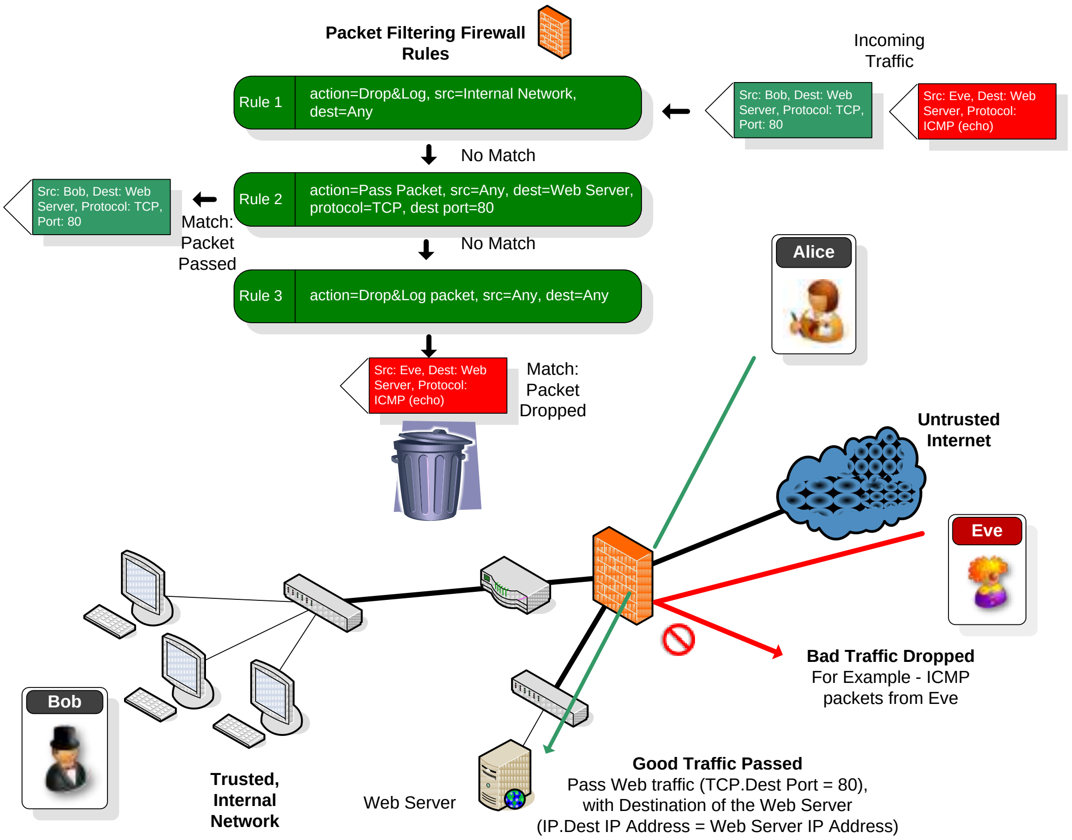

Packet Filtering Firewall Rules
Incoming Traffic
Rule 1
action=Drop&Log, src=Internal Network, dest=Any
Src: Bob, Dest: Web Server, Protocol: TCP, Port: 80
Src: Eve, Dest: Web Server, Protocol: ICMP (echo)
No Match
Rule 2
action=Pass Packet, src=Any, dest=Web Server, protocol=TCP, dest port=80
Src: Bob, Dest: Web Server, Protocol: TCP, Port: 80
Match: Packet Passed
No Match
Rule 3
action=Drop&Log packet, src=Any, dest=Any
Src: Eve, Dest: Web Server, Protocol: ICMP (echo)
Match: Packet Dropped
Alice
Untrusted Internet
Eve
Bad Traffic Dropped
For Example - ICMP packets from Eve
Bob
Trusted, Internal Network
Web Server
Good Traffic Passed
Pass Web traffic (TCP.Dest Port = 80), with Destination of the Web Server (IP.Dest IP Address = Web Server IP Address)

Figure 12 Packet Filtering Process

If a packet does not match any specific rule in the ruleset, a default action is applied. The packet is typically dropped on most firewalls. So we **Pass what we know** and **Drop what we don't**. This is what's known as a **Closed Firewall**, and is an example of a closed security stance, which is generally regarded as best practice. An example filtering ruleset, for **incoming traffic,** is shown in Figure 12.

Packet Filtering Problem

Most Internet services are bi-directional - data is sent in both directions. All TCP connections are bi-directional, having a two way client-server handshake to set up the connection, before any data is communicated. The TCP connection is characterised by the Source and Destination IP Addresses and the Source and Destination ports. But only the Destination Server port identifies the service, for example Web Servers always run on port 80, but the Source port is a random port, and usually dynamically chosen at run time, and so is unpredictable. So the traffic going to the Web Server is fairly easy to create firewall rule for, but there is a problem with the firewall catering for the return traffic.

To illustrate the problem, let's take an example of Bob connecting to an external Web Server somewhere out on the Internet.

Outward traffic:

- Source=Bob, SourcePort=random(1024-65,536), Destination=Web Server, DestinationPort=80.

Return Traffic:

- Source=Web Server, SourcePort=80, Destination=Bob, DestinationPort=random(1024-65,536).

The packet filtering firewall rules to allow this traffic to pass are shown in Figure 13. Bobs port number is not known until run time, so the wildcard 'Any Ports' must be used for the return traffic rule – Rule 2. Unfortunately, Rule 2, is not very secure. An attacker like Eve can very easily spoof the source port of 80 and use any destination port they like. These packets will be allowed to pass through the firewall because they match Rule 2, but they can access any port causing a security hole in the firewall. Static filtering rules have big limitations.

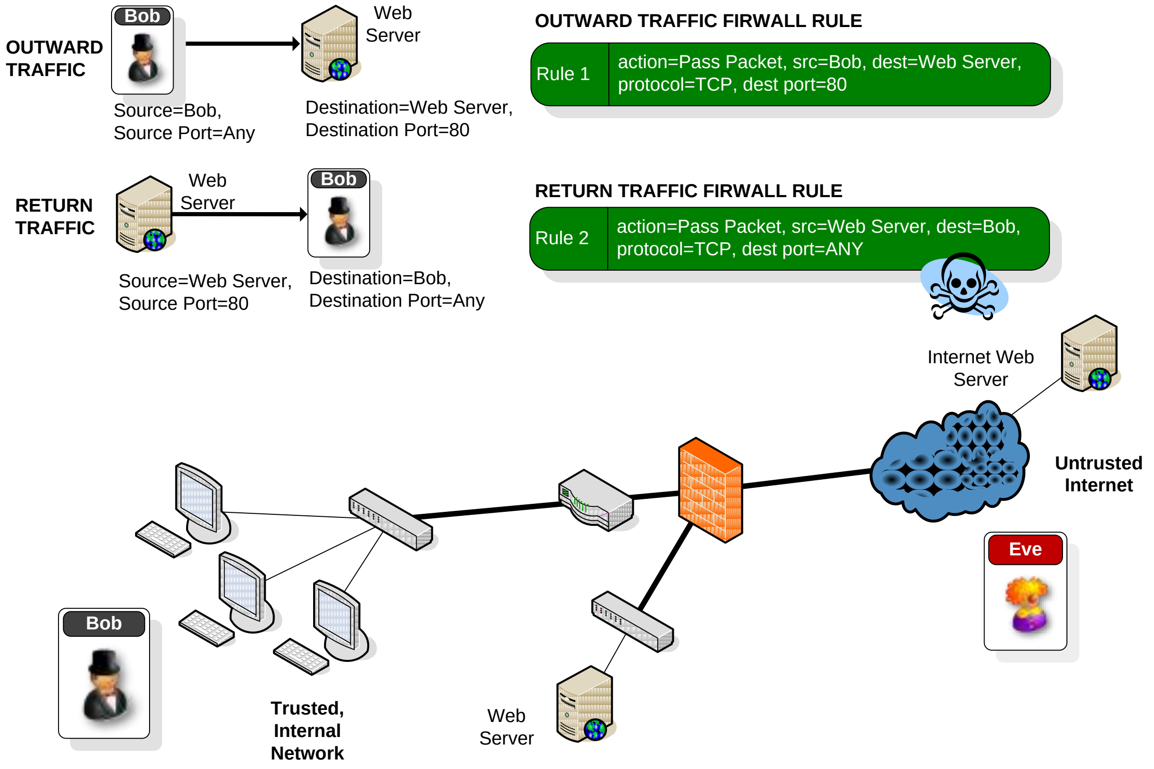
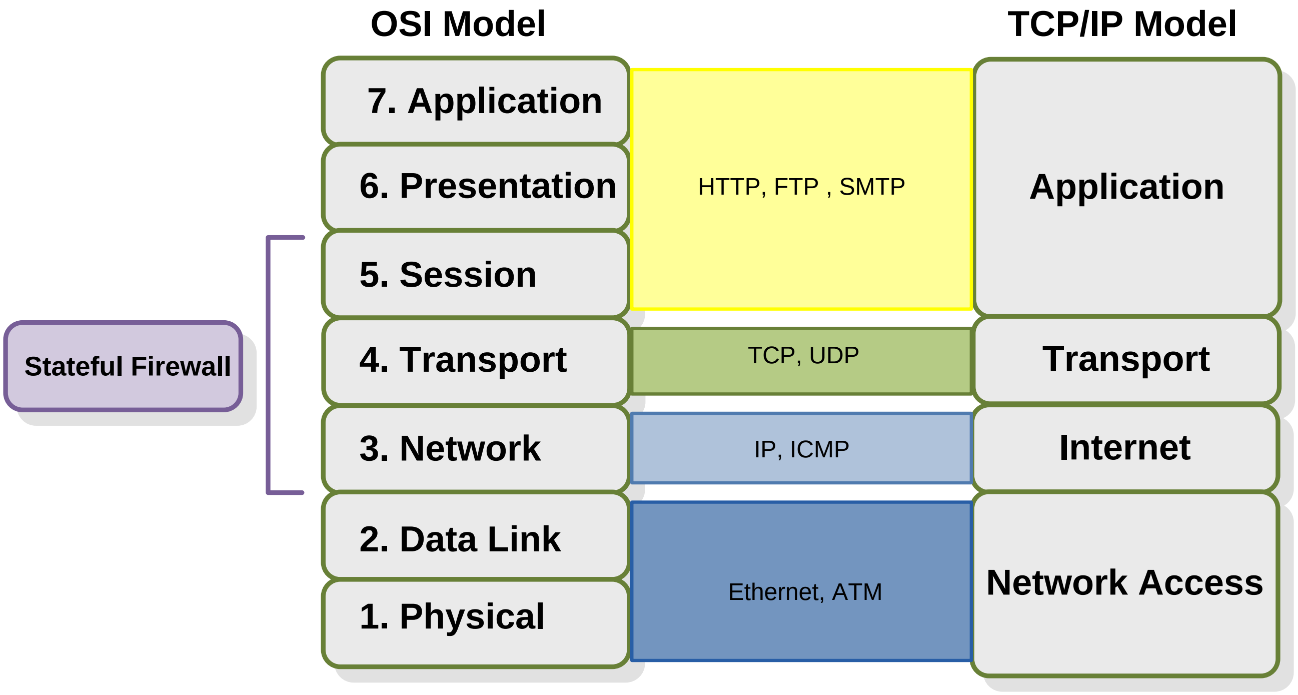

OUTWARD TRAFFIC
Bob
Web Server
Source=Bob, Source Port=Any
Destination=Web Server, Destination Port=80
OUTWARD TRAFFIC FIRWALL RULE
Rule 1
action=Pass Packet, src=Bob, dest=Web Server, protocol=TCP, dest port=80
RETURN TRAFFIC
Web Server
Bob
Source=Web Server, Source Port=80
Destination=Bob, Destination Port=Any
RETURN TRAFFIC FIRWALL RULE
Rule 2
action=Pass Packet, src=Web Server, dest=Bob, protocol=TCP, dest port=ANY
Internet Web Server
Untrusted Internet
Eve
Bob
Trusted, Internal Network
Web Server

Figure 13 Packet Filtering Problem

Therefore almost all modern firewalls currently go further than static packet filtering, and are **stateful**. Static Packet filtering firewalls should only be used for basic tasks, such as segmenting an internal network, or simple bulk filtering on a boundary router, such as filtering for IP Spoofing.

Stateful Firewalls

Most firewalls can now be stateful, to address the limitations in static packet filtering. It is the most commonly used firewall technology, probably the most flexible. Stateful firewalls keep track of established connections in both directions, flowing through the firewall. Basic Stateful Firewalls operating at Layers 3, 4 of the OSI model, the network and transport layers, and to an extent layer 5 the session layer. They can do everything a packet filtering firewall can, filtering on each packet - using filtering rules, but they also take into account the **context of the connection**, that the packet is part of. They can keep track of the **state** of the connection, remembering established TCP flows between a client and a server.



The state is tracked by storing information about each connection in a **state cache**, kept within the firewall. Depending on the traffic passing through the firewall, the state of the connection is updated in the state cache, and this is then used to check if packets are valid packets in the context of the connection, and whether they should be passed through the firewall or dropped.

Stateful packet filtering is sometimes called **dynamic filtering**, as 'virtual' return firewall rules are generated for a connection, based on the initial outbound packets.

The main benefit of stateful firewalling is that it provides a better range of security than static packet filtering. Broad insecure firewall rules for return traffic are no longer necessary. Another benefit of this is that the firewall rulesets are a lot shorter, and therefore easier to create and manage, and the system administrator is less likely to make errors when configuring the firewall. Firewall Rulesets can be very large and extremely complex to understand and manage, especially as multiple firewalls around the network may have different policies and so different rulesets. There has been extensive research into these areas, but no simple solutions, which solve all the problems, have been found. See (5) and (6) for further reading on this topic.

Another major benefit of a stateful firewall is performance, as the cache can be implemented in a very efficient way, such as a hash table or search tree. This is significantly faster than comparing every packet against every rule in a ruleset.

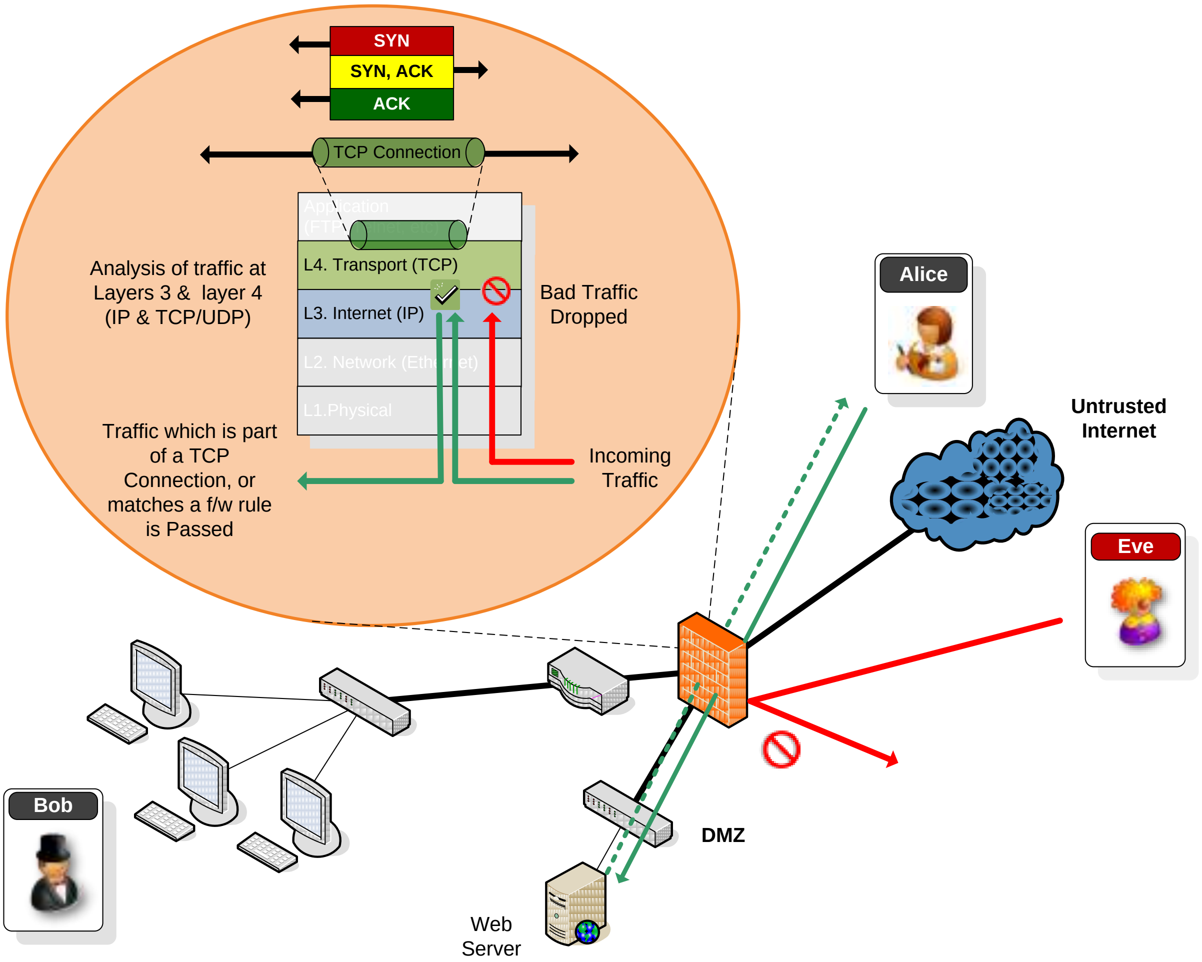


Figure 14 Stateful Packet Filtering Firewall

A disadvantage to statefull firewalls, is the problem of the state table becoming full. If this happens while a connection is still in use the return packets will simply be dropped. This can occur if the firewall runs out of memory fro the state cache. This could be due to large traffic loads, (possibly having to store hundreds of thousands of connections) but more likely as a result of a direct Denial of Service (DoS) attack on the firewall itself. If an attacker floods the firewall with bogus connections, such as a SYN Flood attack, the firewall may drop legitimate connections.

Stateful Packet Filtering Process

The stateful packet filtering process is shown in Figure 15. The operations in the process are numbered, and described below:

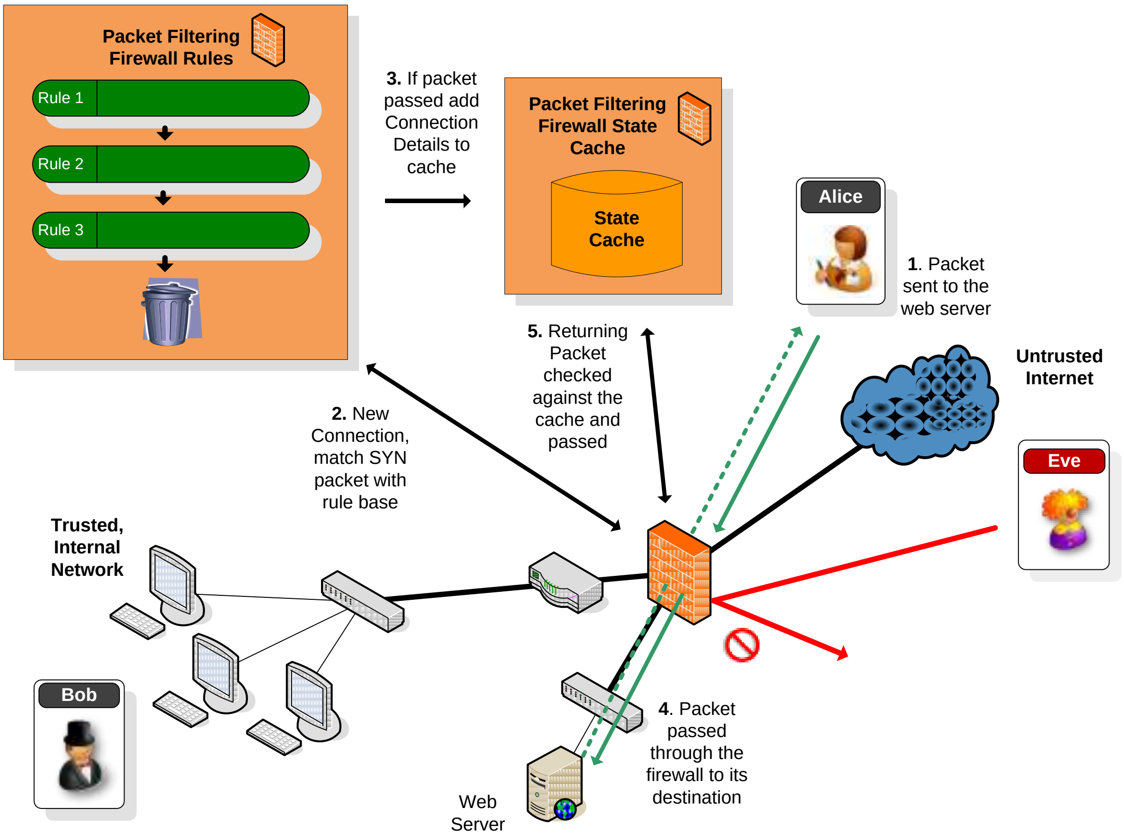

Packet Filtering Firewall Rules
Rule 1
Rule 2
Rule 3
3. If packet passed add Connection Details to cache
Packet Filtering Firewall State Cache
State Cache
Alice
1. Packet sent to the web server
5. Returning Packet checked against the cache and passed
Untrusted Internet
Eve
2. New Connection, match SYN packet with rule base
Trusted, Internal Network
Bob
4. Packet passed through the firewall to its destination
Web Server

Figure 15 Stateful Packet Filtering Process

1. A packet is sent by Alice. When it reaches the firewall, and has been decapsulated up to layer 4, the firewall then examines the TCP header flags field for SYN, RST, ACK, FIN flags being set. From these, it can determine the state of the connection the packet is part of.
2. If it is the first packet in a TCP connection the SYN flag will be set. In this case the firewall matches it against its rule base.
3. If the packet is passed, details of the connection are entered into the state cache. The Client and Server IP Addresses, the Protocol, and Port Numbers are entered (its run time so we know the Client Port number, unlike the problem with static packet filtering).
4. The firewall forwards the packet to the destination, in this case Bob's web server.
5. When inspecting further packets, the firewall compares the packet data with the state cache data to determine if the packet is part of an existing connection or not. If the packet is part of an existing connection it is passed through the firewall immediately, without having to be checked against the rule set. If no cache information exists, the packet is matched against the firewalls ruleset, and the appropriate action is taken.

The connection data is removed from the state cache when the connection is torn down (once the FIN handshake has been completed). For UDP packets, the state cache simulates a session, and then removes the connection details based on a time out, as there is no UDP tear down. A time out would also be used to remove TCP sessions which have not been torn down. An example of a state cache is shown in

Table 3 below.

Table 3 Example of a Stateful Firewall State Cache

Source Address	Source Port	Destination Address	Destination Port	Connection State
192.168.1.1	1025	146.178.10.5	80	Initiating
192.168.1.4	1030	178.147.1.4	80	Established
192.168.1.1	1035	146.178.11.7	25	Established
192.168.1.2	1042	146.178.10.5	80	Established

To summarise: the first packet of a Connection effectively opens a hole in the firewall, and the state cache allows packets in both directions to flow through the hole, without the need to consult the rule base again, making the process very efficient.

Stateful Firewall Packet Flows

To return to the packet flow diagrams used earlier in the chapter, we can again look at the single DMZ topology, and hopefully now the pairs of flows are fully understood.

In Figure 15, Bob accesses a web server out on the Internet, and the firewall allows the outbound connection, storing the details in the state table. When the web server responds to Bob, the state table is checked and the packets returning from the web server are allowed. From then on, packets from the connection are allowed through the firewall based on the state table and do not need to be filtered using firewall rules. These same pairs of flows through the firewall exist between different network trust zones. One pair from the, Internal private network to the external public network, another from the Internal private network to the DMZ, and another from the external public network to the DMZ.

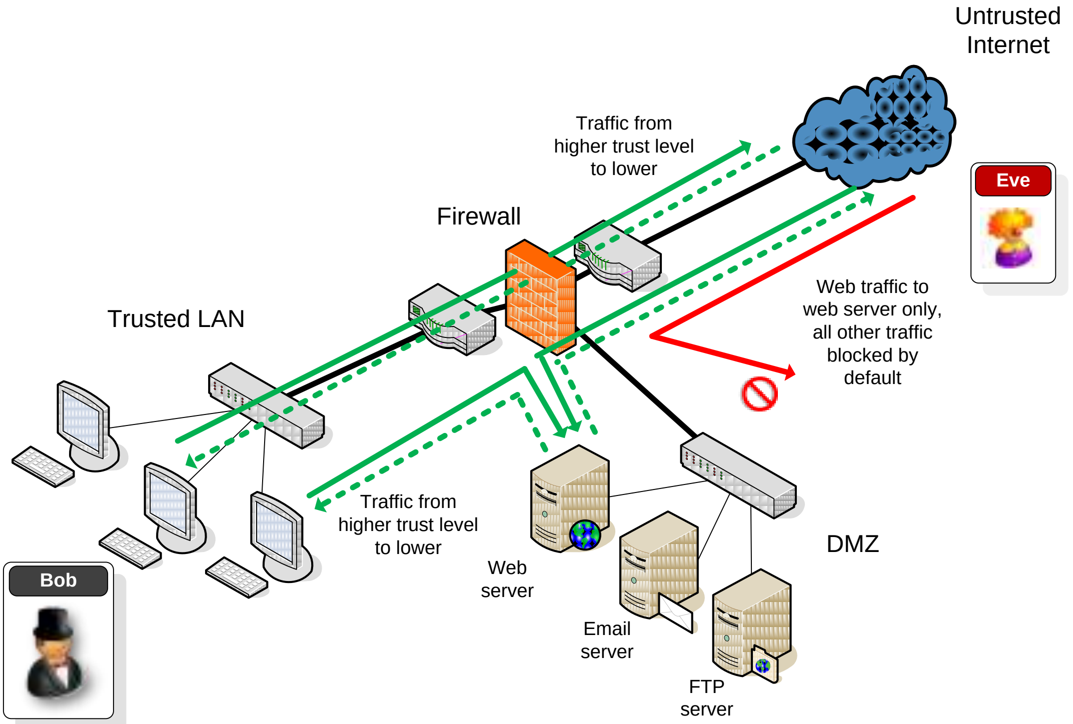


Figure 16 Packet flows for Stateful Firewall in a Single DMZ Topology

Stateful firewalls do have some limitations, such as if an attack is contained in application layer data, such as a virus. Similar to the situation described earlier, in the section on Network Perimeters, a malicious attack could be hidden inside an application layer protocol, such as HTTP. This is a common technique used by attackers nowadays, for attacks using malicious software, or buffer overflows, and it is generally accepted that application layer headers and application data have to be inspected to mitigate these threats. This can be done by application layer firewalls, Application proxies, and IDPS sensors, or a combination of all three.

Another limitation of stateful firewalls is that some applications open multiple communications channels – sometimes called **multi-channel protocols**. These services, such as **peer to peer applications** and **FTP** use standard ports to set up a communications channel for control, but use dynamic ports for communication sessions to send data. Both static and dynamic packet filtering firewalls have a problem with dynamic port selection, as the information on which ports are to be used is passed in the application layer data.

Application Protocol Inspection Firewalls

Sometimes called **deep packet inspection**, this type of firewall is normally an extension of the stateful packet filtering firewall. They operate at the network, transport and application OSI layers 3, 4 and 7, providing protection to applications and services, as well as providing special support for multi-channel services, such as FTP.

An application layer firewall essentially provides some Intrusion Detection System functionality. An inspection engine can analyse the network, transport and application protocols and compare their behaviour with vendor defined standards that the protocols should follow. This conformity checking of the protocols prevents misuse of the commands in a protocol, and identifies unexpected sequences of commands, such as repeated commands, common in attacks like buffer overflows and DoS attacks. Because the firewall is validating the protocol commands, it is also checking whether the protocol is what it says it is, such as if it really is HTTP content in an HTTP packet. This means the firewall can also block hidden, or protocols **tunnelled** within others protocols, such as peer to peer protocols tunnelled inside HTTP.

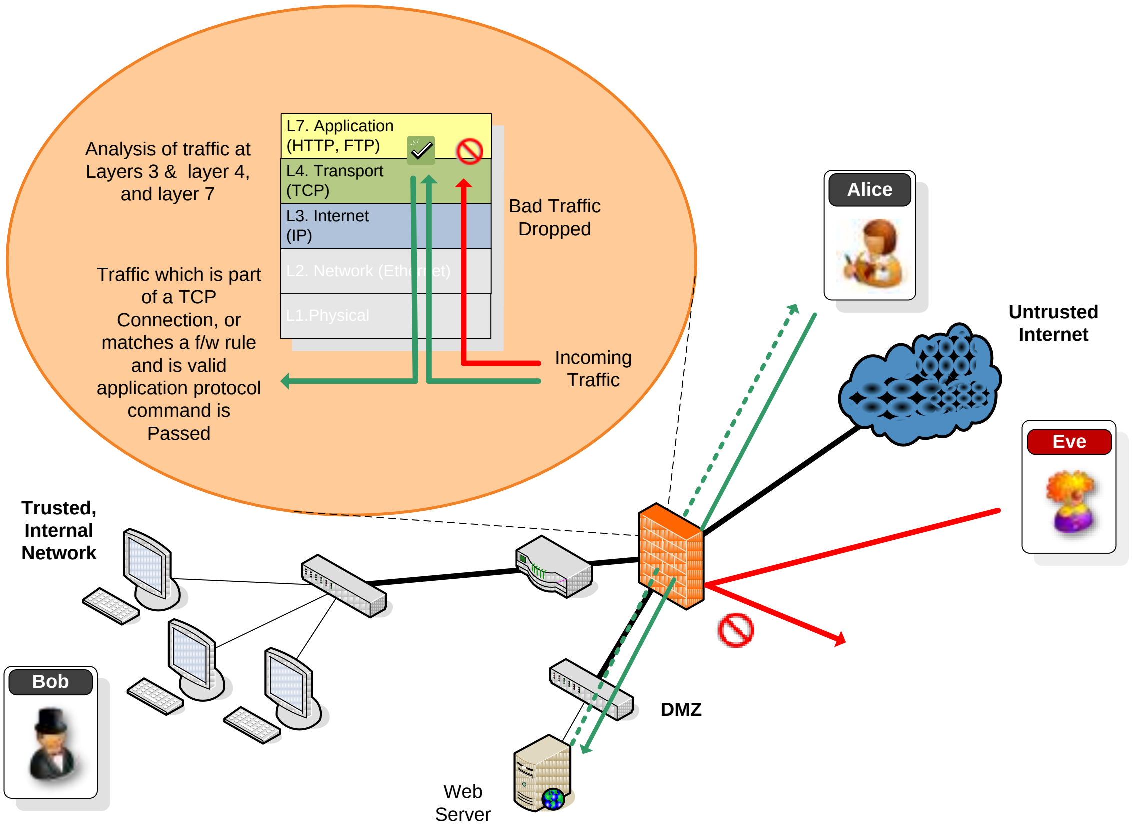

Analysis of traffic at Layers 3 & layer 4, and layer 7
L7. Application (HTTP, FTP)
L4. Transport (TCP)
L3. Internet (IP)
L2. Network (Ethernet)
L1.Physical
Bad Traffic Dropped
Incoming Traffic
Traffic which is part of a TCP Connection, or matches a f/w rule and is valid application protocol command is Passed
Alice
Untrusted Internet
Eve
Trusted, Internal Network
Bob
DMZ
Web Server

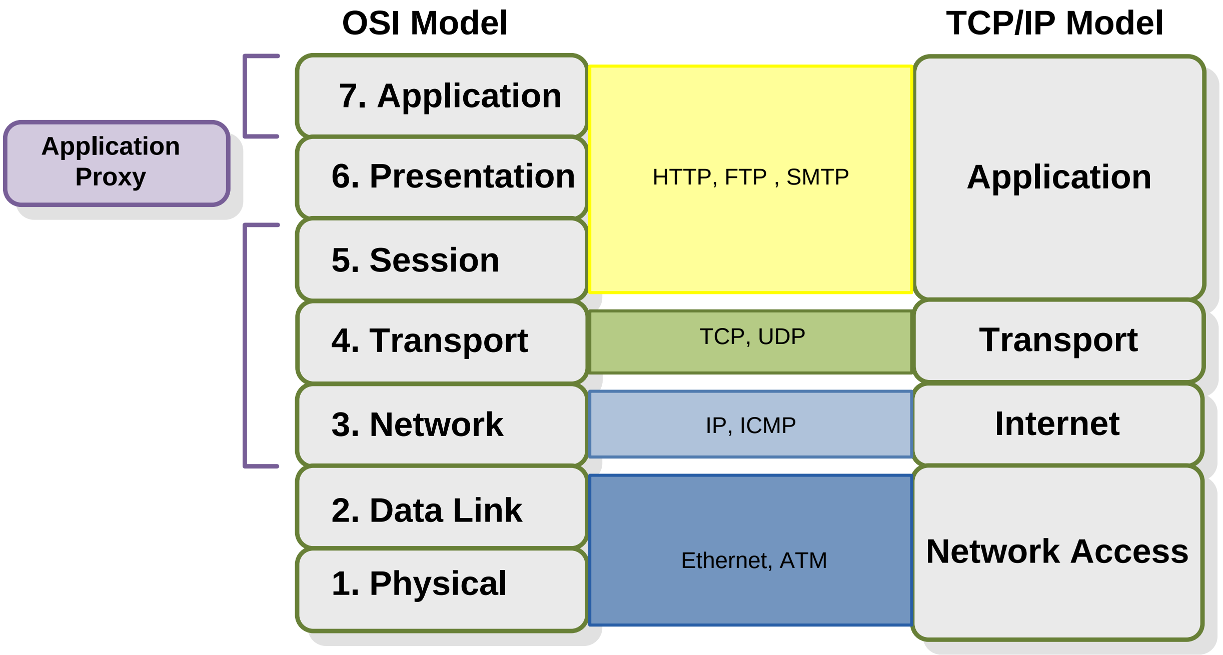
Figure 17 Application Protocol Inspection Firewall

Application inspection can also monitor for multi-channel services, and determine the dynamic port numbers for the data channels. Many protocols now open secondary TCP sessions for data communication, after the primary session – which is created using a well known port. The firewall would then allow data transfer via the dynamic ports for the duration of the connection.

Firewalls with stateful , and protocol inspection are not to be confused with fully functional Intrusion Detection and Prevention System (IDPS) systems, which will be discussed in a later chapter. These IDPS systems provide significantly more in the way of intrusion detection than protocol conformity checks. They use complex signature and anomaly based intrusion detection techniques.

Application Proxy Firewalls

Sometimes called Application Gateways or Application Layer Firewalls, these operate at pretty much all the layers in the OSI model, filtering traffic at layers 3, 4, 5 & 7, including the application layer information and the application data payload. They offer the greatest level of security of all the different types of firewall.



The firewall contains a proxy agent, which acts as intermediary between networks, typically the trusted internet network and the Internet. In a communication between two hosts, there is no connection directly between the systems. On a successful connection to the proxy, two separate connections are created by the proxy, one between the client and the proxy, and one between the proxy and the destination. As shown in Figure 18.

> “Think of two guards, one inside the walls and the other outside the walls. The guard outside knows nothing about the insides of the castle. The guard inside knows nothing about the world outside the castle. But the guards pass packets to each other”

Bruce Schneier (7).

Security experts love their medieval metaphors! From the point of view of the two hosts there is one direct connection, and the proxy is transparent. The IP Addresses of internal host machines are not made public, and the external hosts only communicate with the proxy, so the proxy firewalls is the only IP Address visible on the Internet. A proxy agent program checks the firewall ruleset and passes or drops packets. It also has the ability to perform user authentication, via techniques like Username/Password, a token such as an Active Directory token, or source address.

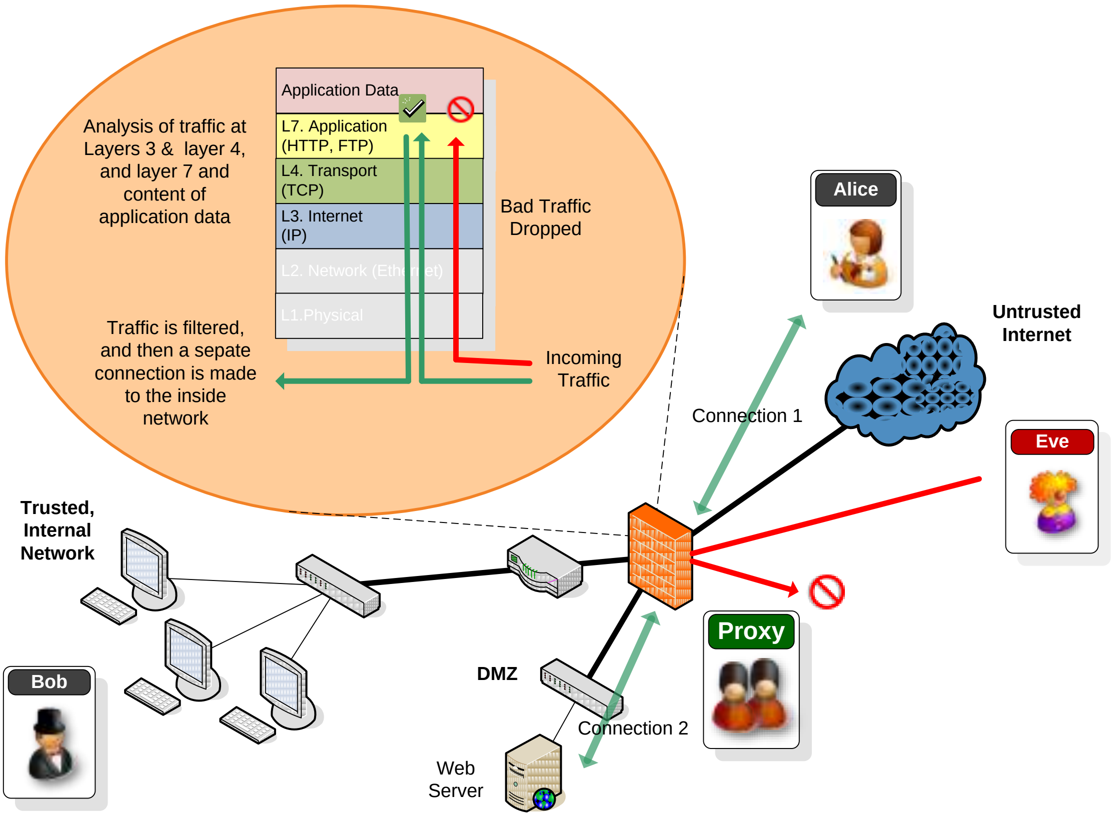

Analysis of traffic at Layers 3 & layer 4, and layer 7 and content of application data
Application Data
L7. Application (HTTP, FTP)
L4. Transport (TCP)
L3. Internet (IP)
L2. Network (Ethernet)
L1.Physical
Bad Traffic Dropped
Traffic is filtered, and then a sepate connection is made to the inside network
Incoming Traffic
Alice
Untrusted Internet
Connection 1
Eve
Trusted, Internal Network
Proxy
DMZ
Bob
Connection 2
Web Server

Figure 18 Application Proxy Firewall

The proxy firewall works at the application layer, and can inspect the application header, and also the actual content of the application data. Unlike an Application Protocol Analysis Firewall which mainly checks for compliance with protocol definitions, the proxy can thoroughly check the packet contents, checking for abnormal protocol behaviour (usually a sign of an attack), and drop these packets. The proxy can check for breaches of the security policy, such as executables as email attachments, or active content, such as JavaScript or ActiveX, being downloaded within HTTP traffic, and either drop the packets, or remove the offending items. Another feature of a proxy is that it can be configured to only allow certain actions to be performed for specific applications. For example, the FTP 'put' command – which allows files to be written to the FTP server - could be blocked. If this command is sent to the proxy it would simply not forward the packet onto the FTP server, as that command is not allowed.

> "From the beginning, proxy firewalls were recognized as being more secure, because they effectively are implementing a correctness check upon the application protocols they gateway. This is still an important property of proxy firewalls. For example, when the author first implemented the FTP proxy in the DEC SEAL firewall, he simply left out unused FTP protocol commands that allowed users to issue remote commands to the FTP server. Years later, when hackers discovered those commands and attempted to exploit them, they simply did not work against proxy-protected networks because the proxy refused to gateway the command through to the target. Sites behind packet filtering firewalls were vulnerable, if the reachable systems behind the firewall were themselves vulnerable."

Marcus Ranum (8).

Advantages of proxies over packet filtering firewalls and stateful firewalls are numerous. The proxy offers a degree of security the other cannot, as it prevents direct connections between two hosts. The inside host can be made invisible to the outside host. Also, detailed logging can be done as the firewall inspects the entire packet, including the data payload. Logging of application commands can be done, which may help detect or analyse new attacks. User authentication can also be performed by the proxy, and not just authentication of the device. Inspecting the application layer means attacks such as buffer overflows, malformed URLs, and unauthorised accesses can be detected. Authorisation at the application command level can also be performed by the proxy, based on the user authentication. IP Address Spoofing attacks and DoS attacks can also be mitigated by the proxy, via inspection of the application payload, for malicious content such as repeated commands.

The Disadvantages over other firewall technologies, are because of the inspection of the entire packet as well (the same reason the proxy firewall is so secure). One disadvantage is the limited support for new protocols, as a specific proxy agent is needed for each type of network traffic – for each application protocol - crossing the firewall. Another disadvantage is that the firewall spends a lot of time inspecting all the different part of each packet, and so performance is not as good as packet filtering firewalls. A solution to this is to deploy application specific **dedicated proxy servers**, just behind the firewall These can be used to support less time sensitive services, typically email and web traffic filtering.

Dedicated Proxy Servers

Dedicated proxy servers perform the proxy services of a proxy firewall, but have no inherent firewalling capabilities. They are typically application specific, for example a proxy server might be used to validate and filter HTPP web traffic only. They are usually deployed just behind a perimeter firewall, and normally the firewall would pass the traffic for the specific application to the dedicated proxy to deal with it, for example an email proxy. The proxy would then perform filtering or logging on the traffic, and drop or pass it onto the internal server. For example the email proxy would then pass on the valid email traffic to the internal email server. An example of an email proxy is shown in Figure 19. As mentioned previously, the type of application protocols which dedicated proxies are typically used for are less time sensitive, like Web traffic and email, and not real time application such as Voice over IP (VoIP).

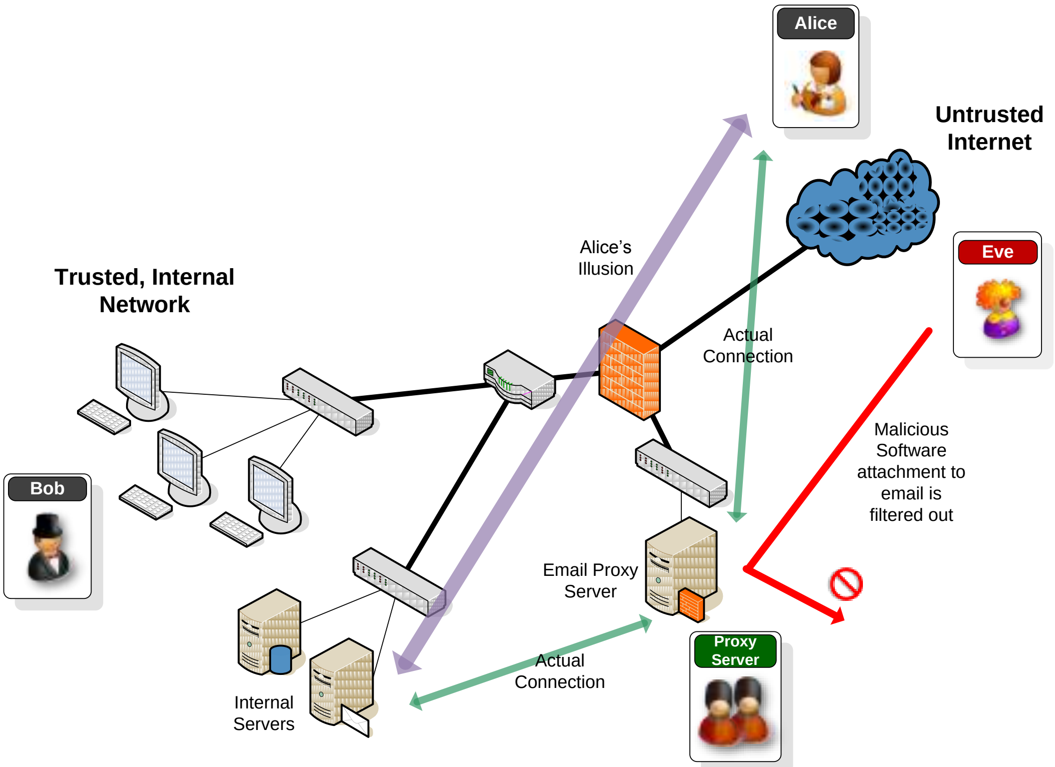


Figure 19 Dedicated Proxy Server

A proxy server can also be used for proxying outbound traffic, originating within the internal network, and filter or log the traffic, before passing it onto the firewall for delivery to its external destination server. A common example is an HTTP proxy server, deployed behind the firewall, which user's web traffic would flow through on its way to the destination, external, web server. The dedicated HTTP web proxy would decrease the load on the firewall and allow detailed validation and filtering, in both directions, for attacks and policy violations. These could be users breaking the organisations web browsing policy, malicious software in the return traffic, and black and white list URL filtering. Black list URL filtering simply blocks access to a black list of URLs (open security stance), and white list filtering which only allows access to a certain list of URLs (closed security stance).

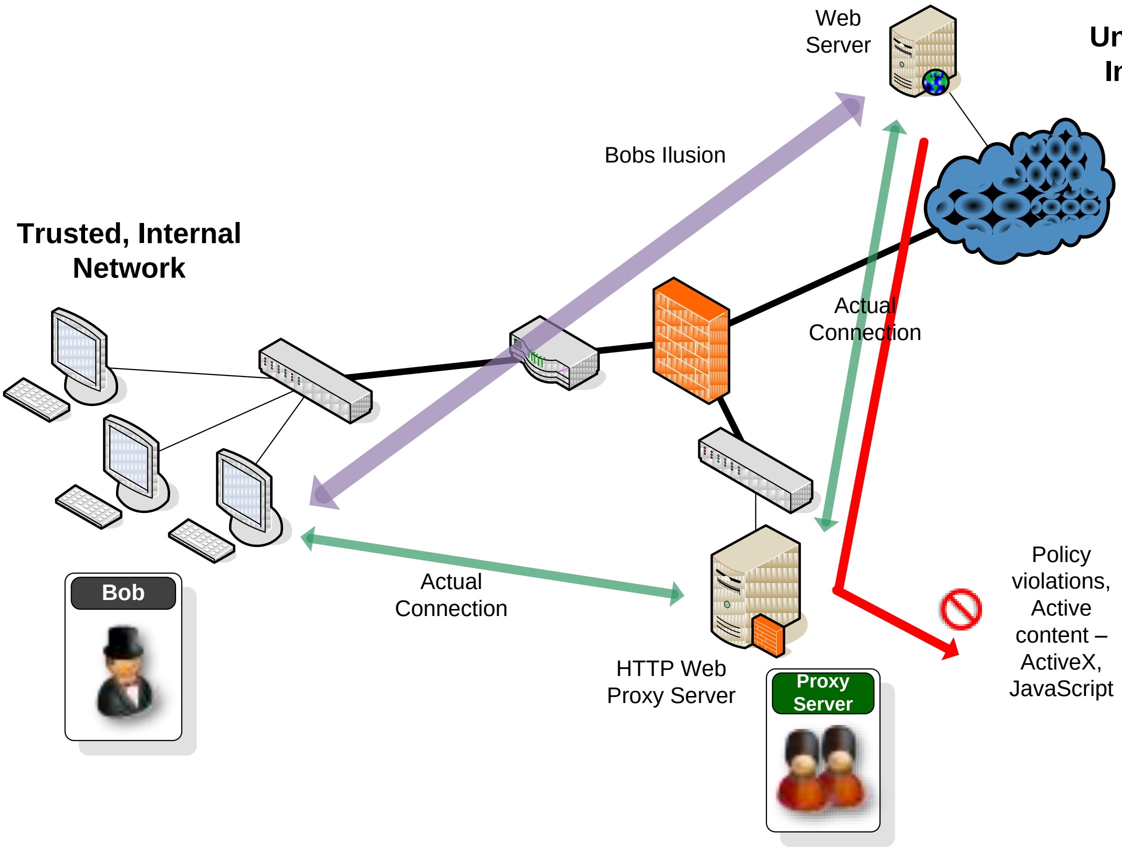


Figure 20 Outbound HTTP Proxy Server

The use of inbound proxy servers has decreased as the actual application servers they protect now implement a lot of the logging and filtering that proxies used to do. Proxy servers are still extensively used for *outbound filtering*, particularly for web HTTP traffic filtering. Many organisations also use proxies for caching web pages to save bandwidth and speed up response times. In large organisations many proxies may be used in parallel, to share the load, so several servers might be used for HTTP and several for email. Tiered proxy topologies are also common in larger organisations, were filtering proxies might be deployed behind the firewall, and a user authentication and web page caching proxy might be deployed behind that, inside the network.

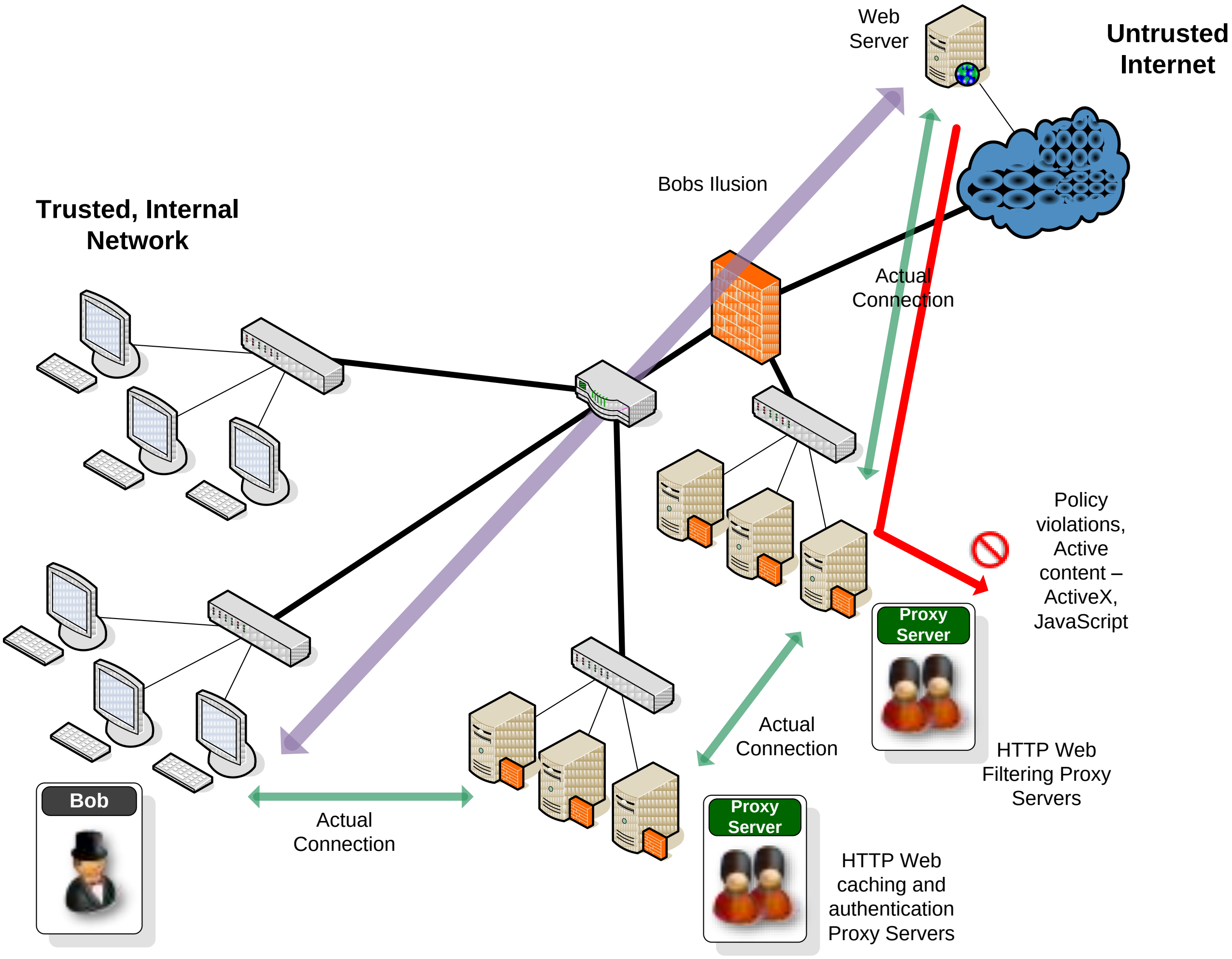

Web Server
Untrusted Internet
Bobs Ilusion
Trusted, Internal Network
Actual Connection
Policy violations, Active content – ActiveX, JavaScript
Proxy Server
HTTP Web Filtering Proxy Servers
Actual Connection
Bob
Actual Connection
Proxy Server
HTTP Web caching and authentication Proxy Servers

Figure 21 Multi-Tier HTTP Proxy Server Deployment

Examples of products which might be used in this type of deployment, would be Bloxx servers (9) for the web filtering and behind those, Microsoft Internet Security and Acceleration (ISA) (10) Servers for the Web page caching and authentication, maybe using Active Directory tokens. Authentication methods including Active Directory are covered in the Access Control Chapter. These proxies would typically be deployed behind one or more enterprise firewall and security appliances, which would also perform VPN termination and Network Address translation.

4.5 References

1. *The Design of a Secure Internet Gateway.* **Cheswick, Bill AT&T Labs.** 1990.

2. **Badham, John.** *Wargames - http://www.imdb.com/title/tt0086567/.* Leonard Goldberg Production, 1983.

3. War dialing gets an upgrade. *SecurityFocus .* [Online] April 2009. http://www.securityfocus.com/brief/918.

4. **Stuart McClure, Joel Scambray, George Kurtz.** *Hacking Exposed 6.* s.l. : McGraw Hill, 2009.

5. Checkpoint Software Firewall. *Checkpoint.* [Online] Checkpoint. http://www.checkpoint.com/products/softwareblades/firewall.html.

6. **IANA.** IANA - Port numbers. *IANA.* [Online] http://www.iana.org/assignments/port-numbers.

7. **Hamed, Ehab Al-Shaer and Hazem.** Taxonomy of Conflicts in Network Security Policies. *IEEE Communications Magazine.* 2006, Vol. 44.

8. **Alfaro, Frederic Cuppens and Nora Cuppens-Boulahia and Joaquin G.** Detection and Removal of Firewall Misconfiguration. *Conference On Communication, Network, and Information Security (CNIS).* 2005.

9. **Schneier, Bruce.** *Secrets and Lies - Digital Security in a Networked World.* s.l. : Wiley, 2000.

10. **Ranum, Marcus.** What is "Deep Inspection". *Marcus Ranum.* [Online] http://www.ranum.com/security/computer_security/editorials/deepinspect/.

11. **BLOXX.** BLOXX Products. *BLOXX Web Site.* [Online]

12. **Microsoft.** ISA Server Home Page. *ISA Server.* [Online] http://www.microsoft.com/forefront/edgesecurity/isaserver/en/us/default.aspx.

13. **Schneier, Bruce.** Crypto-Gram Newsletter - May 15, 2000. *Bruce Schneier.* [Online] http://www.schneier.com/crypto-gram-0005.html.

14. RFC 1918. *IETF.* [Online] http://tools.ietf.org/html/rfc1918.

15. **IANA.** IANA - ICMP. *IANA.* [Online] http://iana.org/assignments/icmp-parameters.

16. **IETF.** RFC 4890. *IETF.* [Online] http://www.ietf.org/rfc/rfc4890.txt.

17. **Paquet, Catherine.** *Implementing Cisco IOS Network Security (IINS): Authorized Self-study Guide: CCNA.* s.l. : Cisco Press, 2009.

18. **Cisco.** Cisco ASA 5500 Series. *Cisco.* [Online] http://www.cisco.com/en/US/products/ps6120/index.html.

19. —. ASA Application Protocol Inspection. *Cisco Web Site.* [Online] http://www.cisco.com/en/US/docs/security/asa/asa72/configuration/guide/inspect.html#wp1383691.

20. —. Cisco IOS Security Command Reference. *Cisco Web Site.* [Online] http://www.cisco.com/en/US/docs/ios/security/command/reference/sec_book.html.

21. **Sedayao, Jeff.** *Cisco IOS access lists.* s.l. : O'Reilly, 2001.

22. **Bellovin, SM and Cheswick, WR.** Network Firewalls. *IEEE Communications Magazine.* 1994, Vol. 32, 9.

23. **Elizabeth D. Zwicky, Simon Cooper, D. Brent Chapman.** *Building Internet firewalls 2nd Edition.* s.l. : O'Reilly, 2000.