## Firewalls

In this video, we're going to take a look at firewalls. Firewalls are a key part of our perimeter defense in organisations. If you consider a company which has a large number of computers, it's incredibly likely that these computers aren't going to be connected in an internal local area network.

But equally it's very likely that they are going to be connected to the internet. So much of modern practice involves making use of cloud-based software and software as a service. As a result, a connection to the internet is a key component of modern working.

However, in doing so this introduces the possibility of a number of attacks through that network connection. So it's important to look at the different mechanisms we have which allow us to protect our networks. Firewalls effectively are a security concept, they can be implemented either as software or as hardware.

There's two general approaches to firewalls. We have a packet filtering approach or a proxy-based approach. In this video we will break down each of those, looking at how they work. The aim of a firewall is to protect your network and the devices connected to it. It's a combination of security mechanisms such as monitoring and filtering of traffic packets, which are either entering or leaving a network. The aim is to provide a private network some protection from an unsecured external network, like the internet.

It typically uses a combination of packet filters and/or proxies. We'll take a look at each of these in turn. A firewall can put conditions on inbound and outward bound traffic. These are referred to as a firewall rule. In packet filtering, rules are created which evaluate a packet to determine whether it should be allowed to pass through the firewall or not, whether that's outbound or inbound.

This can involve looking at things such as where it comes from, where it's intending to go to, and the protocol being used. The rules can also be customised based on whether the packets are incoming or outgoing. For example, it may be that a company is less concerned about outgoing packets and more concerned about incoming packets, which might contain things such as malware. Depending on these values a packet is either accepted, denied, or dropped.

Accepted means that it's allowed to pass through the firewall. Denied means it's sent back to the sender. Due to the cost of bandwidth, this isn't often done. Drop, on the other hand, means the packet is simply removed from existence, or deleted. To implement this, the security administrator, or other colleagues who are responsible for the firewall, first have to decide what the policies are.

For example, a policy could be to deny all traffic coming in using FTP. Another could be to allow traffic to a specific web server, but deny all other incoming traffic. These policies are then translated into technical statements called rule sets.

This typically comprises a name, a protocol such as FTP or TCP, a source IP and the port destination, and whether it should be allowed or denied. Here is an example GUI construction of a firewall interface. As already mentioned, the rule has things like a name or a description, a protocol, a source IP, the destination port, and the destination IP. This policy then looks to allow or deny based on specific requirements. Take a moment to pause the video and look at the two rules represented here. Recall that client applications can use random, uncommon port numbers, but typically port 80 is HTTP.

Why is the good rule good, and why is the bad rule bad? Hopefully you've paused and had a think about this. The bad one only looks at requests for port 80 to port 80. But we could feasibly have HTTP requests from any port between 1,024 to 65,535. So traffic could still feasibly be let through.

The good one is better because it examines all possible source ports. Other things to watch out for when creating such rules includes trying to avoid conflating policies. For example, if one rule says to allow something and another one says to deny it for the same configuration, then this would clearly be contradictory and cause issues.

The same port numbers, IP addresses, and so forth, are the kind of configurations you might want to consider. You may also wish to consider having a default rule, that is, what do you do if none of the other rules are broken or applied, and ensure that it's reflective of what you truly want to happen. Two approaches to policies can be to deny all within a particular range, or to allow all within a given range.

For example, certain IP addresses might be preapproved, or specific IP addresses may be blocked. If you have a Windows machine, then you can explore the firewall which is in built within the system. An example of this is shown here. There are, of course, other alternatives. For example, Unix-based systems often use IP tables.

Either way, it might be worth having a dig around and seeing if you can look at what this is like for the machine that you're using. Moving on to a proxy-based approach. In this approach, we have a proxy server. These act as a gateway, or intermediary, from one network to another. For example, from the internet into an internal local area network. A request is made to the proxy to gain access to a given service. The proxy server checks this and passes it on, assuming it is satisfied. It acts like a server to the client, and like a client to the server, hence the name proxy.

It allows you to implement policies based on user IDs, and hide information about the structure of an internal network. This means that the proxy server is the only entity seen by the outside world. Note that for every service that you provide, there needs to be a proxy server to deal with that service.

The kind of things that the proxy server is going to ask the client who is making the access request is; what is it they are looking to access, who are they, and it's going to require some form of authentication for that. If it is happy with the information provided, then it can pass on the request to an internal router, which can then route that request to the server.

If we look at outbound packets when using proxy servers, if the proxy server sends out the packets from the internal network it has the potential to be sniffed. And the original IP address of the internal server could be revealed. In order to stop that, a modern proxy firewall will add its own IP header to the packets, and then the sniffer would only see the IP address of the proxy, hence a hiding internal hosts.

Another way to stop sniffing of the IP header is to employ header destruction. In this approach the firewall proxy destroys the packet header completely, and replaces it with its own IP header. A possible limitation of a firewall which is using a packet filtering approach is that port numbers could potentially be spoofed.

This means that when looking at the filtering rules, if they are looking for specific port numbers, then this could potentially be exploited. However, with a proxy firewall this isn't so easy to do, since it enforces particular protocols. For example, it ensures that port 80 is in fact HTTP, and deals with only one application per server. Getting firewalls right can be a challenge in itself.

But hopefully you have a better understanding of the kind of things that need to be taken into consideration. Well, that's us for this video, where we've broken down the two general approaches to firewalls. Why not take the time to look at your own firewall on your home network, and see how it's configured? I hope you've enjoyed the video, and I'll see you next time.