1495 UNIVERSITY OF ABERDEEN

CELEBRATING
525 YEARS
1495 – 2020

ABERDEEN 2040

Network Security Technology

Firewalls, internet technology, routing and filtering, and packet filters

September 2025

Outline of lecture

1. Threat environment.
2. Firewalls.
3. Zero trust architecture.
4. Internet technologies.
5. Routing and filtering.
6. Packet filters.

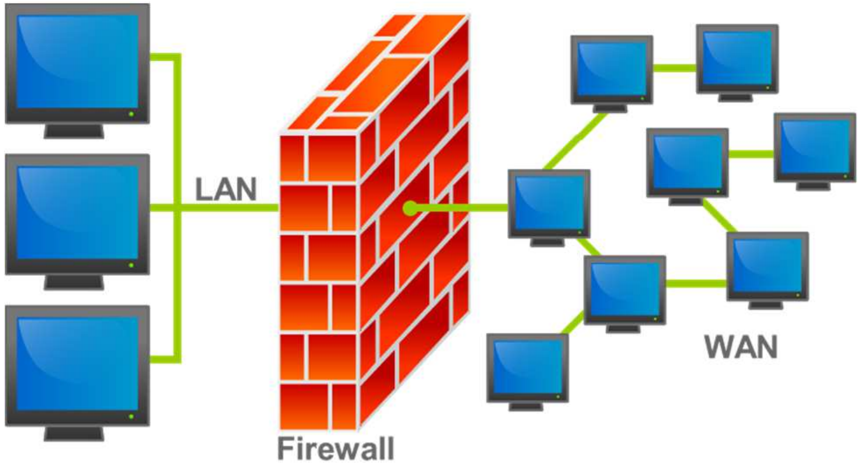
Threat environment

1. A modern network (call it *P* for protected) is typically connected to the internet, *I*.
2. All kinds of remote intrusion, sniffing, scanning, malware, protocol attacks will be thrown at *P*.
    1. Need to control access to *P*’s resources from those outside.
3. Eventually some attacks will get through, but to complete they may need to communicate back to the outside again.
    1. E.g., some entity has got in and taken over an account and wishes to exfiltrate sensitive data.
    2. Moreover, there may be insiders that try to use resources of *I* inappropriately.
    3. Need to control access to *I* from inside *P*.

Firewalls: objectives and policy

1. A certain network is connected to a larger external network (often the internet).
    1. This means that there is a notion of a boundary of the network.
    2. Is it obvious what this is?
2. We need policies to say how to protect it and manage the security of its information resources (remembering C.I.A triad).
3. Idea is to use:
    1. (Automated) policies to defend at the boundary
    2. Mechanisms to do it.

Firewalls



“Firewall" by Bruno Pedrozo - Feito por mim. Licensed under CC BY-SA 3.0 via Commons

1. A **firewall** is a network security device that is used to control traffic between two parts of a network.
2. **Ingress filtering**: control what traffic enters
3. **Egress filtering**: control what traffic leaves.

Common firewall locations

1. Common locations:
    1. At the boundary of an organizational network
    2. At the boundary of a subnet (e.g., for some subnet belonging to a department or physical site).
2. Here, and below, we are talking about firewalls for the entire network, i.e. **network firewalls**.
3. There are also firewalls to protect individual end-points, e.g. **personal firewalls/host-based firewalls**.

Three requirements for controlling traffic

1. **Ingress filtering**: All traffic entering the protected network needs to enter via the firewall.
2. **Egress filtering**: All traffic leaving the protected network needs to leave via the firewall.
3. The firewall itself must be protected against the effects of attack.
4. Recall that in access control, all access to the protected resource had to be controlled by the reference monitor (guard) in the trusted computing base. This is similar.

Firewall policies

1. **Permissive policies**:
    1. ``Everything which is not forbidden is permitted’’
    2. **Default allow:**
        1. The default is to allow all traffic, except that which is explicitly blocked (**blacklisted**).
        2. If you forget to block something, then it will be allowed, and this might subsequently be exploited.
2. **Restrictive policies**:
    1. ``Everything which is not permitted is forbidden.’’
    2. **Default deny:**
        1. The default is to block traffic, except that which is explicitly allowed (**whitelisted**).
        2. Typically, only that which meets a useful purpose is allowed (e.g. HTTP, POP3, SMTP, SSH).

Firewall policies – trade off

1. There is trade-off between absolute security and ease-of-use (at an organizational level).
2. Restrictive policies provide more of what I previously called *absolute* security, since anything not listed will be blocked.
3. On the other-hand, if something useful is blocked, it may lead to a loss of productivity until someone complains and the issue is eventually resolved.
4. As usual, if the policy is too restrictive it may lead to users developing workarounds that can compromise security:
    1. Working outside the firewall (at home etc.)
    2. Using various combinations of proxies, port forwarding, ssh tunnels, IP spoofing, source routing ….

Firewall policies examples

1. Policies are often implemented in ruleset that look a bit like ACLs with positive and negative entries. Examples including the following:
    1. Allow from internal network to Internet: HTTP, FTP, SSH, DNS
    2. Allow from anywhere to mail server: SMTP only
    3. Allow from mail server to internet: SMTP, DNS
    4. Allow from inside to mail server: SMTP, POP3
    5. Allow reply packets
    6. Block everything else.
2. Defining and managing rulesets is important (and complicated) when deploying firewalls in practice.

Types of firewalls

1. We discuss four standard types below:
    1. (Stateless) packet filters
    2. Stateful packet filters
    3. Circuit-level proxies
    4. Application-level proxies.
2. This taxonomy is not perfect.
3. There are many different names, subtle distinctions and overlaps between categories.

Analogy for firewall types

1. Packet filters are like telephone call barring by number.
2. Application filters are like telephone call monitoring (and disconnection) by listening to the conversation.
3. Modern challenges:
    1. Virtual networks and cloud services make it hard to define a simple boundary/perimeter of the network.
    2. Compromises within a large and complex network likely at some point in time.

Zero trust architecture

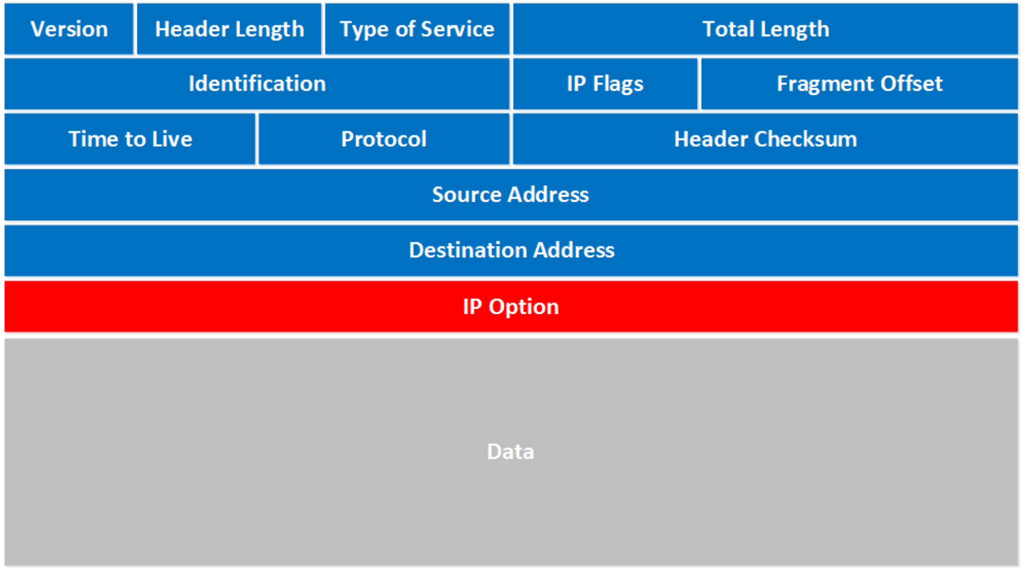
1. **Zero trust** is an approach to the design and implementation of networks that says that hosts and traffic should not be trusted simply because they are in the network.

2. *Everything* must be authenticated and authorized.

Packets/Protocol Data Unit (PDU) at IP layer

1. The header for the packet contains:
   1. IP source address,
   2. IP destination address,
   3. Protocol (TCP, UDP, ICMP,…),
   4. Other attributes.
2. The packet also has data in the payload.

IPv4 packet header



Version | Header Length | Type of Service | Total Length

Identification | IP Flags | Fragment Offset

Time to Live | Protocol | Header Checksum

Source Address

Destination Address

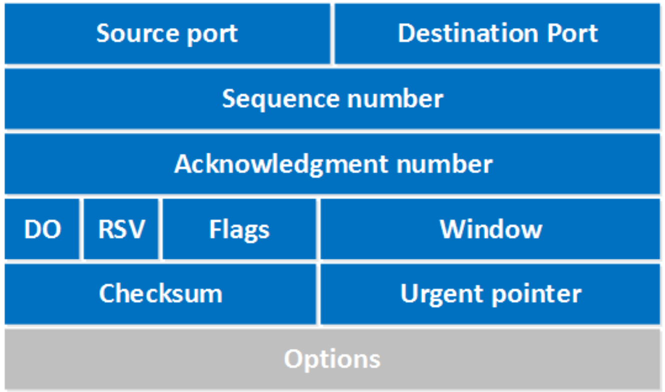
IP Option

Data

Source: https://networklessons.com/cisco/ccna-routing-switching-icnd1-100-105/ipv4-packet-header

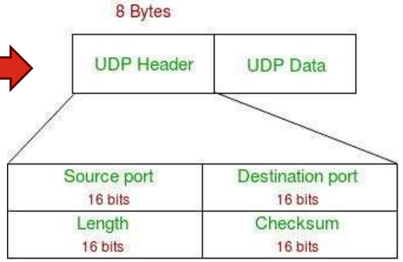
Transport layer PDUs

1. TCP header contains:
    1. Source port
    2. Destination port
    3. Flags (including SYN, ACK)
    4. Sequence Number
    5. Acknowledgement Number
    6. Other information



Source: https://networklessons.com/cisco/ccie-routing-switching-written/tcp-header

2. UDP header's have:
    1. Source port
    2. Destination port
    3. Other information.



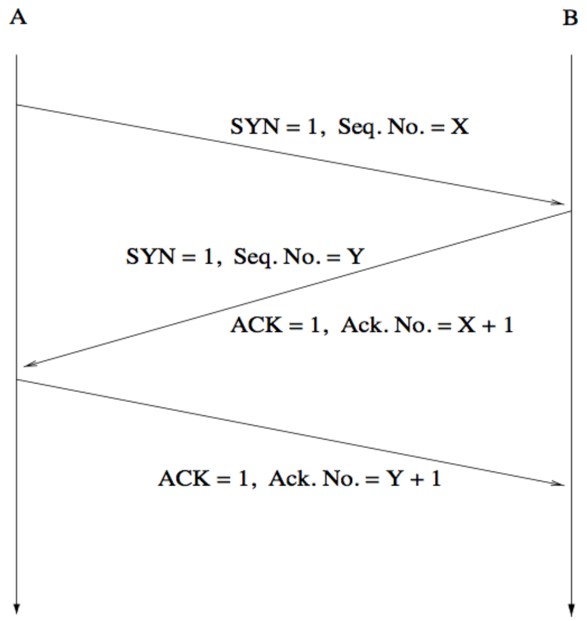
Source: https://www.geeksforgeeks.org/user-datagram-protocol-udp/

3. Transport layer PDUs are packaged into the PDU payload at the IP layer.

TCP flag bits

1. The TCP Header includes certain 1-bit fields called flag (or control) bits. We need to know about two of these:
    1. **SYN:** Synchronize sequence numbers
        1. Sent in the first packet when initiating a connection
        2. Only the first packet sent from each end should have this flag set.
    2. **ACK:** Acknowledgement Number is valid.
        1. All packets after the initial SYN packet sent by the client should have this flag set.

TCP handshake



1. TCP connections start with a handshake.
   Server is listening at some port.
2. Client machine opens a TCP session on server B with the three-way handshake protocol:
   1. A -> B: SYN, X
   2. B -> A: SYN|ACK, Y, X+1
   3. A -> B: ACK, Y+1
3. X and Y are 32-bit sequence numbers.
   1. Note that in the messages, these are the only identifiers of A and B. Vulnerabilities arise from their discovery by attackers.
4. X+1 and Y+1 are acknowledgement numbers.

Handshake

1. The handshake is designed to set-up TCP connections for general reliability and integrity of transmission.
2. It is not for security purposes.
3. The relevant fact for today, is that the SYN and ACK flags enable us to know which end is initiating the connection and which is responding.

Routers

1. Forward and find routes for IP packets:
2. From the packet, the router could:
    1. use the packet header.
    2. use the data in the packet: e.g., see name of web page requested
    3. check for validity of the packet: e.g., size is legal and matches true size
3. Router knows other things about the packet:
    1. Interface (link layer) it arrived on
    2. Interface it will leave on.

Routers (cont’d)

1. Some routers may be stateful:
    1. they may keep track of some of the history of packets that they have seen.
2. Such a router may be able to determine:
    1. Whether a given packet appears to be a response to another packet
    2. How many other packets have recently been seen to or from the same host
    3. Whether a given packet is identical to another, recently seen packet
    4. If a given packet is a fragment of a larger packet.
3. Some such devices may be able to look at transport-level information (e.g. TCP segment header information).

Routers and screening routers

1. An ordinary router looks at the destination address of each packet and picks the best way it knows to send the packet towards its destination.
   1. The decision is based solely upon the destination.
   2. If it does not know a route, then it sends back an ICMP ``destination unreachable'' message.
2. A **screening router** additionally determines whether packets *should* be forwarded to their destinations.
   1. This is done using the configuration of the router which is intended to enforce security policy.

Possible filtering actions for routers

1. Basic actions:
   1. Forward packet to destination
   2. Drop the packet (just forget it)
   3. Reject packet (& send notification to sender)
   4. Log information about the packet
   5. Set off an alarm
2. Sophisticated routers can also:
   1. Modify the packet (e.g., do Network Address Translation)
   2. Send the packet to a destination other than the one it was intended for (e.g. to proxy, or for load balancing)
   3. Adapt filtering rules (e.g. to accept replies to a UDP packet, or to deny all traffic from a site that has sent hostile packets).

Example policies programme into screening router

1. Block all incoming connections from systems outside the network, except for incoming SMTP connections (for email).
2. Block all connections from certain systems that you distrust.
3. Allow email and FTP services, but block dangerous services like TFTP, X Windows (outside a tunnel), RPC and the r-services (*rlogin*, *rsh*, *rcp*, etc.).
4. Usually, screening routers are based on packet filters.
5. The two terms may be synonymous for some people.

Packet filters

1. Uses information at IP and Transport layers.
    1. A packet filter applies rules specifying which packets are allowed through the firewall, and which are to be blocked are applied to packets individually:
    2. It does not care about connection state;
    3. It does not care about streams of packets.
2. Typical rules specify:
    1. source and destination IP addresses, and
    2. source/destination TCP and UDP port numbers.

Packet filters (cont'd)

1. Rules for traffic in both directions can be defined.
2. Can be implemented by a router which examines the IP and Transport Layer (TCP/UDP) headers of every packet going through and can drop them.

Stateless packet filter example

Rule Name	Direction	Source Address	Source Port	Destination Address	Destination Port	Action
1	Inbound	98.103.1.156	80	192.168.10.1	60033	Permit
2	Outbound	192.168.10.2	60124	133.142.27.3	20	Deny

1. Rules might also be conditioned to filter on other TCP information: SYN flag, ACK flag.
2. The internal addresses here are `private’ rather than `public’ (under NAT).

Packet filter limitations

1. Only static rules can be enforced.
2. Certain common protocols are difficult to handle. For example:
    1. When a client sends an outbound FTP request to an FTP server, the firewall cannot link the data packets coming back in (on a different port) from the server to this request.
3. Some examples to address the above:
    1. We can have a blanket rule for all packets coming in from port 20, say, or
    2. We can have a blanket rule for all packets from port 20 from an IP address nominated in advance, but
    3. We can't have dynamically defined rules.

Stateful packet filters

1. Packet filters that keep track of the state of connections.
2. These can understand requests and replies.
   1. E.g., they know about the TCP-handshake (SYN, SYN-ACK, ACK).
3. They typically adapt their handling of packets dynamically, depending on the traffic seen.
   1. Sometimes called **dynamic packet filters**.
4. Rules usually only specified for the first packet in one direction.
   1. A new rule is created dynamically after the first outbound packet.
   2. Further packets are then processed automatically.

Stateful packet filters

1. Stateful firewalls can support policies for a wider range of protocols than a simpler packet filter, including FTP for example.
2. Packet filtering can be done by routers:
   1. Gives high performance at low cost.
   2. It is easier to configure for security, platforms with limited functionality.
3. The filtering policies that can be enforced are limited by the parameters that can be observed in the TCP and IP headers.

Advantages of stateful packet filters

1. Using only a few screening routers can protect an entire network.
2. Simple packet filtering is extremely efficient.
   1. Just routers examining headers
   2. More sophisticated filtering work reduces performance.
3. Packet filtering is widely available.
4. Transparent to users.

Disadvantages of stateful packet filters

1. Filtering capabilities are limited:
   1. Hard to configure: too many rules possible,
   2. Filtering rules can be hard to test,
   3. Many desirable rules difficult or impossible to implement on existing products,
   4. Contain bugs and may wrongly allow packets.
2. Can reduce performance:
   1. Can be a problem for certain performance-sensitive uses of routers.

Disadvantages of stateful packet filters (cont’d)

3. Some policies can’t be easily enforced. Examples include the following:
   3. Can’t filter packets by user, or by application.
   4. Might filter by host and port, but the relationship between applications and ports can be many-to-many.
   5. Could accidentally block an additional application through some port
   6. A malicious insider could run an application over a non-standard port.
4. There are more sophisticated `intelligent’ packet filters, and those that force authentication by users.