Advanced Network Security

Attacks at Different Layers

Thomas Martin

t.martin@mmu.ac.uk

March 6, 2019

Outline

Outline

Link Layer Attacks

The link layer is used to connect physically adjacent devices. This can be in a wired or wireless network. We have already discussed how ARP (Address Resolution Protocol) is used to boot-strap network access for new devices.

One important distinction for wired networks is the use of hubs vs. switches. Both connect multiple devices. A hub will simply broadcast all received frames to all other connected devices. A switch will learn which MAC addresses are connected to which port and forward frames intelligently. This improves the efficiency of the network as well as limiting the exposure of sensitive information, and is much more commonly used.

ARP Spoofing[1]

ARP spoofing is quite simple. An attacker joins a network and wishes to intercept traffic between devices A and B. The attacker broadcasts ARP responses to A claiming to be the MAC address for B's IP address, and similarly one to B claiming to be the MAC address for A's IP address. Now when either device wishes to send frames to the other, they will actually be forwarded to the attacker.

The properties that make ARP very useful for managing a network are exactly the reasons why this type of attack is simple. ARP is stateless, devices accept unsolicited responses, and there is no authentication.

[1] https://www.sans.org/reading-room/whitepapers/detection/detecting-responding-data-link-layer-attacks-33513

Ettercap[2]

Ettercap is a powerful Kali tool for executing man-in-the-middle attacks using ARP, as well as a number of other protocols. It has two main modes: UNIFIED and BRIDGED. Unified sniffs the packets that are received (and if promiscuous mode is enabled, this will include frames intended for other devices). Bridged uses two network interfaces and forwards traffic from one to the other (completely stealthy).

Ettercap can poison the ARP cache of victims on the network to become a MitM. Another method is to send spoofed ICMP redirect messages to a victim to claim to be a better route for the internet. It can also attempt a DHCP starvation attack. These last two methods only create a half-duplex mitm. Once established, there are a number of plug-ins that can make use of the mitm position.

[2] See ettercap man page for more details.

CAM Table Exhaustion attack[3]

The Content Address Memory (CAM) table maps the switch's ports to specific MAC addresses. This table is what provides the security benefit of a switch over a hub, traffic is delivered only to the intended recipient, not broadcast to the whole network. However, since the table is learned by observation of traffic, it can be manipulated.

An attacker can flood the CAM table with new MAC-to-port mappings by sending spoofed frames. Since the table is limited in size, eventually the legitimate entries will be overwritten. At this point, network traffic cannot be directly delivered and will need to be broadcast to all ports.

[3] http://www.cisco.com/warp/public/cc/so/cuso/epso/sqfr/sfblu_wp.pdf

DHCP Starvation

DHCP is designed to allow new devices to connect to the network. Each device will need a unique IP address, and to save the effort of having them manually configured, a DHCP server allocates them on a first-come, first-served basis.

Typical DHCP involves four messages: Discovery, Offer, Request, Acknowledge (using UDP on ports 67 and 68). In a DHCP starvation attack, only the first message is sent, multiple times. Addresses will still be reserved for each request, and since the pool is limited (e.g. 253 in a /24 network) they will quickly become exhausted. At that point the valid DHCP server cannot accept any requests, leaving the attacker the chance to run a rogue DHCP server.

Mitigations

There are possible mitigations to these attacks. Validation of ARP responses can limit ARP spoofing[4]. Some switches can inspect the responses and make sure no MAC address is claiming to have more than one IP address (this is stored in a trusted binding table).
Port security can limit CAM exhaustion. In a very strict setting, the MAC address can be specified for each port. Slightly more flexible is limiting the number of MAC addresses that can be learned by a switch port.
To prevent DHCP starvation, all traffic except DHCP packets should be blocked from a client. Once an IP address is received, this should be used to create an ACL in the switch port, restricting spoofed packets or additional DHCP requests.

[4] https://www.cisco.com/c/en/us/products/collateral/switches/catalyst-6500-series-switches/white_paper_c11_603839.html

Outline

Ping of Death

An early method of DoS was the so called "Ping of Death". Originally discovered in 1996[5], it crashed systems by sending ICMP packets larger than the maximum allowed, causing an overflow.
IP packets can be upto $2^{16} - 1 = 65535$ bytes in length, with 20 bytes for the header. An ICMP header has a further 8 bytes, leaving 65507. Sending an ICMP message longer than that should be impossible, due to the length field only having two bytes. However, packets can be fragmented (if certain links have constraints), and a set of fragments can be maliciously crafted such that when they are reformed at the receiver, the total size is greater than 65535. A similar problem was recently discovered with ICMPv6[6]

[5] http://insecure.org/sploits/ping-o-death.html
[6] https://gcn.com/Blogs/CyberEye/2013/08/Microsoft-patch-ping-of-death-IPv6.aspx

Teardrop Attack

Another security issue that arose from fragmentation was the Teardrop Attack[7]. In the normal operation, each fragment will have an offset and size so that when reassembled end-to-end (after the removal of the headers) the original packet is reformed.

The Teardrop Attack is where the offsets are malformed so as to imply the fragments overlap. Not expecting such nonsensical data, the system crashes (instead of discarding the received data). This vulnerability has long since been patched in modern systems.

[7] https://www3.physnet.uni-hamburg.de/physnet/security/vulnerability/teardrop.html

DNS Attacks

Strictly speaking, DNS is an application layer protocol. However, it is vital to the correct working of traffic at the Network layer.

We will discuss attacks on DNS later in the course.
`http://unixwiz.net/techtips/iguide-kaminsky-dns-vuln.html`
`https://www.grc.com/dns/gallery.htm`

Outline

Port Scan

A port scan[8] is part of any attacker's arsenal, but it would not be entirely correct to call it an attack. It is certainly not something that exists because of a vulnerability that can be fixed. A port scan is the sending of packets to a remote system and interpreting the responses to determine which ports are open, which are closed, and which are filtered.

If a host has a (TCP) port open, and a public IP address, then by definition it will respond to any TCP SYN segment. Port scanning software[9] can send many such packets to any number of hosts and ports and collect the results. The most popular type of scan is a stealth scan, where only the first handshake message is send.

[8] http://tart.wikispaces.com/Transport+Layer+Risks+and+Prevention

[9] For example: nmap.

Port Scan

A full TCP connect scan is slower and more likely to be detected, and would only be performed if the attacker did not have the required permissions.

A X-mas scan sets various flag bits to 1 (FIN, PSH, URG) in an attempt to confuse any firewall and IDS in-between the attacker and target.

UDP scanning can also be performed, but because UDP does not have the initial three-way handshake, it is a lot less reliable. It depends on either the server running a commonly known protocol or automatically responding to unsolicited, incorrectly formatted requests.

SYN Flood

When a server receives a SYN request, it needs to allocate a small amount of resources dedicated to that connection (port numbers, sequence numbers, state, buffers, etc.) An attacker can attempt to crash the server by inundating it with requests to try to exhaust its memory.

The attacker does not need to store any state on their end, as they have no intention of actually establishing the connection. They do not even need to give their correct IP address, making it harder to block the attack. Often the only defence is to have sufficient bandwidth and server memory/processing power to handle the flood.

Session Hi-jacking

The Internet protocol stack was created to facilitate communication between trusting peers. As such, no attempts were made to authenticate participants. There are some difficulties, but it is not impossible for an attacker to inject traffic into an already established connection[10].

In order for a maliciously crafted packet to be accepted, the IP addresses, port numbers, and sequence/acknowledgement numbers all have to be correct. Some of these will be publicly known, but the rest would either require guessing or being placed somewhere where the traffic can be eavesdropped. Injecting a packet is then simply a matter of spoofing the source address and updating the sequence number based on the payload size. So long as the attackers packet arrive first it will be accepted.

[10] https://technet.microsoft.com/en-us/library/2005.01.sessionhijacking.aspx

Outline

Application Layer Attacks

The largest attack surface at the Application Layer is HTTP. The OWASP Top 10 is a widely acknowledged list of important weaknesses (which we will cover in depth later).

One change observed in attacks is DDoS shifting to the Application Layer[11]. Denial of service attacks at this layer can be more complex, but also more devastating and difficult to distinguish.

If the attacker has good intelligence on the site, he can inundate it with GET requests for large files to exhaust the bandwidth or POST requests that will exhaust the processing power (e.g. searches).

[11] http://www.pcworld.com/article/2056805/applicationlayer-ddos-attacks-are-becoming-increasingly-sophisticated.html

In Class Tasks

Much of our discussion around the various attacks at different layers have implicitly been considering devices (desktops and laptops) on wired networks.

- Are these attacks equally applicable to smartphones?
- Are these attacks equally applicable to wireless networks (e.g. mobile data and WiFi?)
- Do smartphones and wireless networks introduce new vulnerabilities?

Thank you

Any Questions?