CS 161 Computer Security
Spring 2010 Paxson/Wagner
Discussion 4

February 16, 2010

1. **Protocol Layers** At which network layer does each of the following operate (physical, link, network, transport, or application)?

   - Ethernet **(Link - 2)**
   - SMTP **(App - 7)**
   - SYN packet **(Transport - 4)**
   - UDP **(Transport - 4)**
   - Fiber optics **(Physical - 1)**
   - FTP **(App - 7)**
   - DNS request **(App - 7)**
   - BitTorrent **(App - 7)**
   - TTL field **(Network - 3)**
   - Hub **(Physical - 1)**
   - 127.0.0.1 **(Network - 3)**
   - 802.11n WiFi **(Physical, Link - 1, 2)**

2. **TCP and UDP** Transmission control protocol (TCP) and user datagram protocol (UDP) are two of the primary protocols of the Internet protocol suite.

   (a) How do TCP and UDP relate to IP (Internet protocol)? Which of these protocols are encapsulated within (or layered atop) one another? Could all three be used simultaneously?

   (b) What are the differences between TCP and UDP? Which is considered "best effort"? What does that mean?

   (c) Which is easier to spoof, and why?

   Answer:

   (a) TCP and UDP both exist within the transport layer, which is one layer above IP (network layer). Either can be encapsulated in IP, referred to as TCP/IP and UDP/IP. TCP and UDP are alternatives; neither would normally be encapsulated within the other.

   (b) TCP provides a *connection-oriented*, *reliable*, *bytestream* service. It also includes sophisticated rate-control enabling it to achieve high performance but also respond to changes in network capacity. UDP provides a *datagram-oriented*, *unreliable* service. (Datagrams are essentially individual packets.) It is lightweight. "Best effort" refers to a delivery service that simply makes a single attempt to deliver a

packet, but with no guarantees. The network-layer Internet protocol IP provides such a service, and because UDP simply encapsulates its datagrams directly into IP packets with very little additional delivery properties, it too provides "best effort" service.

(c) Spoofing a UDP packet requires determining the correct port numbers to use, but no more. Spoofing a TCP packet requires both correct port numbers and also correct sequence numbers, making it significantly more difficult.

3. IP Spoofing You are the network administrator for a large company.

(a) Your company will be held liable for any spoofing attacks that originate from within your network. What can you do to prevent spoofing attacks by your own employees?

(b) Is there anything you can do to prevent others from sending your employees spoofed packets?

Answer:

(a) Inspect the source IP address of all outgoing packets. If a packet has an address from outside the range assigned to your network, block the packet. This is called "egress filtering."

(b) It is highly dependent on how your system is setup. If you have many links to other networks from your company, and you know what IP addresses are associated with those networks, if a packet comes in on a link and it does not match with the IP addresses associated with that network, you can filter out those packets. This is called "ingress filtering." However, if you only have one link, this does not work. Furthermore, it is not always possible to associate a set of IP addresses with a link connection. ISPs can do this for traffic coming from their edge customers (on separate autonomous systems). this, however.

4. Sniffer detection As the security officer for your company, your network monitoring has observed a download of a "sniffer" tool. This tool passively eavesdrops on traffic, and whenever it sees a web session going to a server in a `*.yahoo.com` domain, it extracts the user's session cookie. It then uses the cookie to create a new connection that automatically logs in as the user and dumps their `*.yahoo.com` settings.

You become concerned that one of your employees may have installed a network "tap" somewhere among the hundreds of links inside your building, and will use it to run this tool. How might you determine whether such a sniffer is in operation?

Answer: One approach is to send customized web traffic along each of the network's links, as follows. The traffic connects to a remote server with the address `A.B.C.D`, which you know none of your systems would normally connect to. Because the sniffer needs to determine whether a given connection goes to a `*.yahoo.com` domain, it will need to look up the address $\rightarrow$ domain mapping for `A.B.C.D`. By monitoring for such lookups, you can determine that a sniffer appears to be in operation, since none of your normal systems should have reason to make the lookup.

Another approach is again using customized web traffic, this time with connections to a `*.yahoo.com` domain. You "seed" the connections with a unique (fake) session cookie and monitor your outbound traffic for any *additional* connection that uses the fake cookie.

5. DHCP Spoofing After physically connecting to a local network, a client may use the dynamic host configuration protocol (DHCP) to establish the network settings necessary to communicate over the Internet. To do so, a client first broadcasts the *DHCP discover* message. Upon receiving a *DHCP offer* message from a local DHCP server in response, the client will send a *DHCP request* message back to the server. If the client then receives a *DHCP ACK*, they may begin using the network.

(a) To attack this protocol, which of these messages must an attacker spoof?

(b) Is it necessary for the attacker to be connected to the same local network as the victim, or can the attack be carried out remotely?

(c) DHCP provides (among other things) the following configuration information to the connecting client: IP address offered to the client, lease time, IP address of the DHCP server, subnet mask, default gateway, IP address of a DNS server. Which of these values might the attacker wish to overwrite?

Answer:

(a) They would normally spoof the DHCP offer and DHCP ACK messages to ensure that the client obtains the modified network settings and begins to use them.

(b) It would not normally be possible for this attack to be carried out remotely, since an attacker must see the DHCP discover message to begin the attack.

(c) The most useful values to overwrite would be the default gateway and the IP address of a DNS server. If the default gateway is changed to the address of the attacker, they could observe and tamper with all subsequent (unencrypted) traffic to and from the client. If the DNS server's address is changed to that of the attacker, they could do essentially the same thing, but only for connections beginning with a DNS lookup.

6. **More on TCP and UDP** Suppose you are developing a voice over IP (VoIP) application to allow users to make telephone calls over the Internet. Would you use TCP or UDP to stream the audio samples? Why?

   What if your audio streaming application will instead be used to listen to podcasts (e.g., recorded radio shows)? Would that change your decision? Why?

   Answer: UDP is overwhelming chosen over TCP for real-time / interactive streaming audio applications, primarily due to the latency requirements. Typically, a single packet will carry 10-30ms of audio, and the largest acceptable (one-way) latency is about 150ms. If a packet is dropped, by the time TCP will have successfully retransmitted it, it will be typically be too old to be useful. An application using UDP, on the other hand, will focus its efforts on sending the newest samples as quickly as possible. A dropped packet may result in a moment of static, but it will be much less noticeable than the delay introduced by waiting for the packet to be retransmitted. The TCP throttling mechanisms are also inappropriate for real-time audio.

   In the latter case, TCP would be a more natural choice. Since the audio being streamed is prerecorded, the application can buffer as many samples as it wants before playing them back. Without the latency issues of interactive audio, there will be enough time to retransmit any missing packets, ensuring clear audio without any glitches.