Paxson
Spring 2017

CS 161
Computer Security

Discussion 8

Week of March 20, 2017

Question 1 ***IP Spoofing*** **(15 min)**

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 and are sent out to the global Internet. What can you do to prevent spoofing attacks by your own employees?

> **Solution:** 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." Note that it does not eliminate *all* spoofing, as a host within your network can still spoof a source address belonging to a different host in your network; but it does severely limit the potential damage due to such spoofing.

(b) You now want to evaluate the risk your employees face from spoofed IP packets originating from outside the network. Assess the likelihood and dangers of spoofed IP packets that use TCP as the transport layer protocol. What applications might be vulnerable to such attacks? How does this change with UDP?

> **Solution:** Recall that TCP uses a 3-way handshake to establish a connection. As part of that handshake each side must agree upon a pair of valid sequence numbers. In order to successfully spoof a new connection, or to inject a spoofed packet into an existing connection, the attacker must either know or correctly guess the valid sequence numbers.
>
> For a *blind* spoofing attack, without the use of any other attack techniques, the likelihood of correctly guessing the sequence numbers is quite small.
>
> However, if the attacker is on-path (can eavesdrop) then spoofing TCP connections is quite easy.
>
> An example of an application that would be vulnerable to such an attack would be HTTP. Later in the semester we will discuss TLS (SSL) and how that could help mitigate this attack.
>
> In contrast to TCP, UDP does not require a 3-way handshake. The attacker must only know the right source port from which the victim has started an outgoing connection. Thus, spoofing UDP packets requires much less effort.

If application-layer protocols wish to defend against this kind of attack, they must develop their own defenses. An example of an application that can be vulnerable to such an attack is DNS.

(c) What can be done to prevent parties outside your network from sending your employees spoofed traffic that impersonates your own employees?

Solution: Packets originating from outside your network should never have a source IP address from inside your network. Using this fact, you can filter (block) incoming packets that contain source IP addresses that belong to your own network. This is a form of *ingress filtering*.

Question 2 *DNS* (15 min)

Recall that in a *blind* DNS spoofing attack, the attacker tries to guess the identification number of the DNS request sent by the victim.

(a) For the following, assume that the victim's DNS resolver cache does not contain a record for the domain the attacker is targeting:

i. In a blind spoofing attack, the attacker must know when a DNS request is about to be made by the victim so that the attacker can respond with their attack responses. Recall from lecture how an attacker might learn this information.

ii. What can an attacker do if they successfully get a victim to believe their bogus DNS mapping?

iii. How can an attacker avoid having to carry out this attack for every time the victim visits the targeted domain?

(b) Now assume that the victim's DNS cache has a genuine `NS` record for the domain the attacker is targeting.

i. Can the attacker still be successful at poisoning the `A` records for some of the names belonging to the domain?

ii. Can the attacker poison the `NS` record of this domain? If yes, how?

Solution:

(a) i. Lure a victim to your web site, which contains an image pointing to the site whose DNS record you would like to attack (e.g., `google.com`). When you see that a victim has contacted your site, you know they are about to make a DNS request for `google.com`, so you initiate the attack at this point.

ii. The attacker is in control of the content of any request made to a hostname whose DNS record has been successfully spoofed. For example, if the attacker managed to get the victim to accept a bogus DNS record for `google.com`, then any subsequent request to `google.com` will actually go to an IP address of the attacker's choosing (this will probably be the address of a server under the attacker's control). The attacker might get you to reveal your password to the fake `google.com` at that point. DNSSEC and TLS (SSL) are useful techniques to mitigate the impact of this attack. We will learn about both of these topics later in the semester.

iii. DNS records are cached, so the attacker might set a long TTL (time-to-live) so that the bogus DNS record will reside in the cache for a long time. The attack will succeed for as long as the bogus DNS record lives in the cache.

(b) i. The attacker can still poison `A` records of subdomains. For example, if the victim has an `NS` record for `foo.com` in the cache that points to `ns.foo.com`, then the attacker can simply force the client to lookup subdomains such as `www.foo.com`. The victim only knows who to ask about the subdomain, namely the IP address provided in the `NS` record for `ns.foo.com`, but does not know the answer for that query.

ii. The attacker can also poison the `NS` record using the Kaminsky attack we covered in lecture. In this case, the attacker can specify the same `NS` record in the *Authority section* of the reply to tell the victim that `ns.foo.com` is responsible for the domain `foo.com`. Moreover, the attacker also provides an `A` record for `ns.foo.com` in the *Additional section* pointing to a bogus address. (This extra information is also known as the *glue record*.) Most resolvers will replace the corresponding existing records in their cache with this new information.

Question 3 *Sniffer detection* (10 min)

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 exfiltrates all of their `*.yahoo.com` activity, such as their emails if they use a `yahoo.com` email account.

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?

Solution: 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 → domain mapping for `A.B.C.D`, which, like lookups for domain $\rightarrow$ address, is done using DNS. 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.