cs261 Scribe Notes

Joel Weinberger
2008-10-14

Problems with packet filters:

1. Low level attacks may get through
   - Forged IP src address (against IP based authentication)
   - Stealth port scanning to find services that are running on a host
     - Send packet with ACK bit set
     - If no service is running, RST is returned
     - Otherwise, nothing is returned. Either way, knowledge about the service has been gained.
   - IP ID scanning
     - Send packet with ACK bit set to target, but set src address to some patsy host.
     - Attacker then pings patsy continuously
     - If IP ID is incremented by one, then the increment was only due to the attacker's own ping
     - If, however, the IP ID is incremeneted by two, then there must have been some other message received, most likely a RST in response to the earlier packet sent to the target. Thus, the service must be running on the specific port.
     - If after some sufficiently long time interval the IP ID hasn't been incremeneted by two, the service is not running on the target.

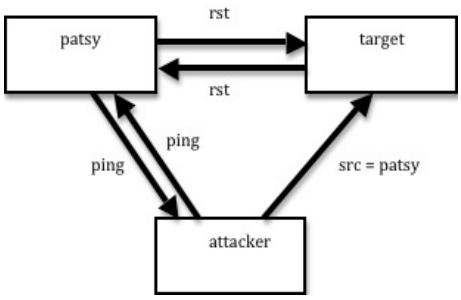

Figure 1: IP ID scanning attack

   - Ping of death: An IP stack bug once had a short represent an IP packet length. However, the IP stack added the packet's header length. Thus, if the packet was long enough, the packet length plus

the header length would overflow the short. Eventually, this would cause a kernel packet on the host. A ping that causes a kernel packet? Perhaps a security vulnerability?

These examples only work when no state is assumed. Otherwise, things like sequence number guessing are necessary.

2. Content based filtering does not work

   - Stateful packet reconstruction necessary
   - One idea is to throw away all packets where the data contains a blacklisted string
     - Very slow because it requires reconstructing a full packet before it can be sent forward to the destination
     - Has to be a blacklist which easily leads to security holes. Whitelists FTW!
     - Even if a blacklist was given, it would be tough to enforce such a policy
   - This idea is very similar to the ideas of intrusion detection, but is an attempt to *p*revent attacks rather than *d*etect them.
   - There is also the problem of a DoS attack by sending fragmented packets. This can easily end up taking all of the firewall's memory since it must retain packet information until the packet has been completed, with no holes.
   - Another problem is that the firewall does not know how the host will interpret packets. Imagine the host gets two packets with the same range:

     ```
     ... filename = foo.jpg
     ... filename = foo.exe
     ```

     Some operating systems will always accept the 1st packet. Some versions of Windows, however, wikll take the second. How can the firewall make a decision about the packet?

     Some intrusion detection systems will raise an alarm if two packets arrive with different content in overlapping ranges. However, our desire is to prevent, not detect, attacks.

     One possible solution is to simply drop the second packet. But what if the time to live is different, such that the first packet will not make it to the destination?

     ```
     (TTL=3) ... filename = foo.jpg
     (TTL=7) ... filename = foo.exe
     ```

One solution is to raise the TTL on all packets to a sufficiently high value that it is guranteed to arrive at the destination behind the firewall. Of course, this still does not solve the problem of which packet was the correct packet to deliver (or if either packet should have been delivered).

3. Application Firewalls

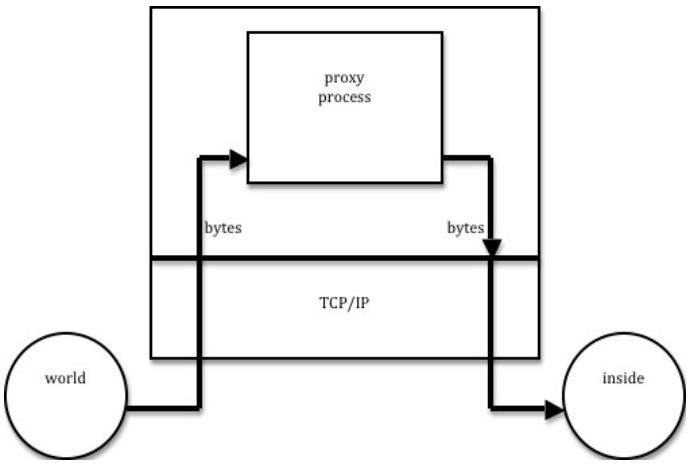

Figure 2: Application Firewall Flow

- Defends against 1st set of problems because outside packets are terminated at the application firewall which provides a better single point of contact.
- Because the connection is terminated at the firewall, there cannot be malicious headers going to the victimss since the headers are created fresh by the firewall. This leaves only malicious data to reach the victims. Effectively, the application firewal takes packets and transforms them into a harmless state
- At this point, Dave gets excited about a Sci-Fi book named, "A Fire Upon the Deep," exclaiming, "Okay, I'm a Sci-Fi dork." Much jumping and hand waving ensues. This ends with Dave insisting that we all need to read the book (see Figure 3).
- There are three basic possibilities for how this might work:
  (a) Insert a reference monitor that replies allow or deny for the packets. However, this relies on maintaining shadow state which is a bad idea.
  (b) Create an *e*mbeded reference monitor. This avoids shadow state, but if there's no mechanism to plug in to, you're stuck!

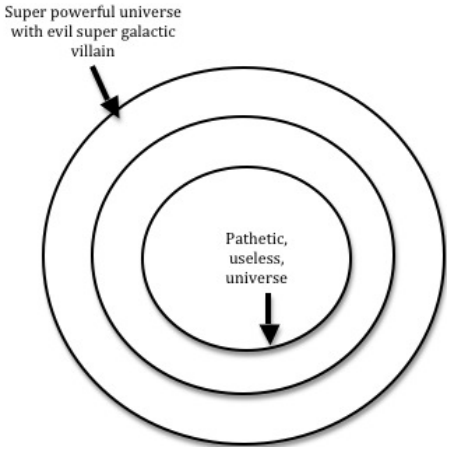

Figure 3: Dave's Crazy Sci-Fi Universe

(c) Transform before the packets reach the end host, such that you parse, filter and unparse, ensuring that all packets are provably safe.

- Almost everything goes over HTTP or mail, both of which are exceedingly hard for Firewalls to deal with, and also require a large group of exceptions such that tunneling and similar mechanisms are always useable. This mitigates a lot of the usefulness of firewalls. Hence why firewalls haven't played a big part in modern security!