Network Security, Part II

ICMP security

The *Internet Control Message Protocol* (ICMP) is used for sending error and status messages through the internet. Each ICMP message has a type, which is a numerical code specifying what type of message it is. For us, the most important types are 8 (echo request), 0 (echo reply), 11 (time exceeded), 3 (destination unreachabe) and 5 (redirect).

An echo request message is more commonly known as a *ping*. This is a usually short message you can send to another machine to see if you can reach it. If it is reachable and is responding to pings, it will return an echo reply. There is a ping tool built into most common operating systems. That tool has options to set things like the number of pings, the amount of data to send, and much more. Below we will look at some of the security implications of ping and ICMP.

Ping flood Pings, like many types of traffic, have been used in denial of service attacks. A simple example is a *ping flood*, which involves sending a lot of pings to the target to flood it with traffic and use up its resources as it tries to respond to each ping.

Ping of Death A older attack is the *Ping of Death*. It exploited a bug in Windows systems of the late 1990s. Most pings only involve sending a few bytes of data, but it is possible to send pings with up to 65,000 bytes of data. The programmers of those older Windows systems did not anticipate pings so large. The large ping would cause a *buffer overflow*. Buffer overflows are one of the most important types of software bugs. A buffer is a region of memory that holds some data, usually a string or an array. If a programmer allocates a certain amount of space for the buffer and later the program tries to store more stuff in the buffer than it can hold, if the programmer was not careful, the extra data can overflow into adjacent memory locations, overwriting whatever is there. In the case of the Ping of Death, that overflow would overwrite important system data that would cause the system to crash.

It was so easy to do. It's just `ping <target name or IP> -l 65000`. That won't work anymore because modern systems have been programmed to avoid this bug. Interestingly, a similar bug showed up in the Bluetooth protocol in the early 2000s. Those who don't know their history are doomed to repeat it.

Smurf attack Another older attack is the *Smurf attack*. This is a type of amplification DoS attack. The highest address on a network is the broadcast address. Things sent to this address will go to all the hosts on the network. In the Smurf attack, an attacker sends a ping to the broadcast address. Everyone on the network will respond to the ping. This, of course, would just send a bunch of traffic back at the attacker, so instead the attacker spoofs the source IP on the ping to be the IP of the target. On a network with 100 hosts, a short 75-byte ping from the attacker translates into $100 \cdot 75 = 7500$ bytes at the target. Repeated enough times, this can overwhelm the target. The defense against this attack is simple: don't allow pings to the broadcast address. Most modern systems do this.

ICMP tunneling When you send a ping, you can send up to around 65,000 bytes of data. There's not really much restriction on what that data can be, so people sometimes use ping data to sneak data into and out of a network. This is called *ICMP tunneling*. Tunneling refers to any type of situation where you put one type of traffic inside another. Sometimes people use the term "encapsulation" for this. If you're on a network that doesn't allow a particular type of traffic but does allow pings, then you can put the disallowed traffic into a ping. For instance, there is a tool that allows you to run an SSH connection using pings.

Ping sweep A ping sweep is where you send pings to some or all of the addresses on a network to find out what other machines are on the network. Nmap can do this with a command like `nmap -sP 10.0.0.0/24`, which would send pings to all the addresses from 10.0.0.0 to 10.0.0.255.

Traceroute *Traceroute* is another useful networking tool that is available on most modern operating systems. It is used to see the path a packet takes from a source to a destination. Here is a traceroute I ran from my home computer to Google.

```
 1     1 ms     2 ms    <1 ms  192.168.1.1
 2    15 ms    13 ms    12 ms  96.120.9.9
 3    38 ms    16 ms    27 ms  162.151.69.213
 4    24 ms    24 ms    14 ms  96.108.5.201
 5    21 ms    22 ms    23 ms  be-34-ar01.mckeesport.pa.pitt.comcast.net [69.139.168.141]
 6    26 ms    23 ms    25 ms  be-31641-cs04.pittsburgh.pa.ibone.comcast.net [96.110.42.173]
 7    24 ms    22 ms    24 ms  be-1411-cr11.pittsburgh.pa.ibone.comcast.net [96.110.38.142]
 8    42 ms    28 ms    29 ms  be-302-cr12.ashburn.va.ibone.comcast.net [96.110.32.101]
 9    29 ms    28 ms    40 ms  be-1112-cs01.ashburn.va.ibone.comcast.net [96.110.32.201]
10    28 ms    36 ms    28 ms  be-2111-pe11.ashburn.va.ibone.comcast.net [96.110.32.122]
11    34 ms    30 ms    27 ms  50.248.119.106
12    27 ms    27 ms    27 ms  108.170.229.246
13    30 ms    31 ms    32 ms  209.85.251.83
14    26 ms    26 ms    27 ms  iad23s63-in-f14.1e100.net [172.217.15.78]
```

This shows all the routers the packet travels through from my computer to Google. In some cases those routers have names, which traceroute will look up via DNS.

Traceroute works by making clever use of the TTL field in the IP header. The TTL or time-to-live field is decreased by 1 by each router along the path from source to destination. If the TTL ever reaches 0, the packet is dropped and an ICMP message is sent back to the sender indicating that the packet's time was exceeded. The purpose of the TTL is to give each packet a finite lifetime on the network. It's possible for packets to get caught in infinite routing loops and the TTL makes it so that eventually those packets will be dropped from the network.

What traceroute does is it first sends a packet with TTL 1. This packet's TTL will decrease to 0 at the first router, and that router will send back a Time Exceeded message to the sender. The sender will be able to see the IP address of the router in that message, so that's how it knows what the first router is. The next thing traceroute does is to send a packet with TTL 2. This packet's TTL will decrease to 1 at the first router and then to 0 at the second router. That second router will send back a Time Exceeded message, and the sender will be able to see the second router's IP in the message. This process continues with TTLs of 3, 4, 5, etc.

If you want to use traceroute, it's called `traceroute` on Mac and Linux and `tracert` on Windows. Traceroute can be used for both good and bad purposes. The good is in diagnosing networking problems. The bad is that it can be used by an attacker to map out what a network looks like.

ICMP redirect ICMP has a redirect message that can be used to tell someone that they have a direct route to a destination. An attacker can use this to mislead someone into routing their traffic through the attacker.

Disabling ICMP Many network administrators disable ICMP messages. The reason is to stop the attacks mentioned above, as well as to make it harder to map out the network. But this makes legitimate networking applications more difficult.

Denial of service attacks

A denial of service (DoS) attack is any attack that denies service to someone. A simple example would be repeatedly trying to log in as someone at a site until the site stops allowing login attempts. Then the real user would not be able to get in for a while, and their service would be denied. Probably smashing someone's computer with a sledgehammer could qualify as a DoS attack. But mostly when we talk about DoS attacks, we are referring to attacks where a site or a server is flooded with so much traffic that it either goes down or real users can't get through because of all the traffic.

We have seen a number of denial of service attacks already. Below we will briefly discuss them and cover a few other attacks. Note that in all of these we will refer to the "attacker". The attacker is usually someone who has botnet and instructs the machines in the botnet to do what the "attacker" is doing in the descriptions below. These types of DoS attacks, where multiple machines are doing the attacking, are called *distributed denial of service attacks* or DDoS attacks.

DNS amplification This is also called DNS reflection. It involves an attacker sending a request to a misconfigured open DNS resolver. That resolver should only answer queries from its own network, but it's misconfigured to answer queries from anyone. An attacker will ask it a query that results in a large answer, usually either a large TXT record or a DNSSEC entry. The attacker spoofs the source IP address to be that of their target so that the big reply goes to the target instead of the attacker. The amplification comes from the fact that the attacker makes a short DNS query that generates a big response.

NTP amplification This is very similar to DNS amplification except that it uses flaws in the NTP protocol.

SYN flood This involves an attacker sending a bunch of SYNs to a target. Each SYN has source IP address spoofed to be that of a real IP address at which there is no one home. The target will send off SYN ACKs in response to the SYNs and those SYN ACKs will not be answered. The target will sit there waiting for replies that aren't coming, and the resources it devotes to these bogus connection attempts will not be available for legitimate users.

Ping flood This is where an attacker sends a whole bunch of pings to a target.

Smurf attack In this attack, the attacker sends a ping to the broadcast address of a network. The ping goes to all the hosts on the network who all send replies. The attacker spoofs the source IP of the ping so that all the replies go to the target.

UDP flood This is where an attacker sends a whole bunch of UDP traffic to the target.

HTTP flood Here an attacker sends a bunch of HTTP traffic to the target. The attacker will often make HTTP requests that require the server to do a lot of work, like maybe do a complicated database lookup. The attacker will also ask for rarely accessed items, which won't be cached, making the server have to work harder. HTTP floods often look like real traffic, which makes them harder to defend against than other attacks. Note also, that unlike most of the attacks, in an HTTP flood, the IP address cannot be spoofed. A full TCP connection has to be made before making the HTTP request.

HTTP floods can also happen by accident, like when a news site like Reddit links to a small site and all of a sudden thousands of people are trying to get to it.

Echo-chargen attack This is an old attack, but it's interesting and worth at least looking at briefly. The echo service is a simple service that echoes, or repeats back whatever is sent to it. The chargen service replies to any message sent to it with a stream of characters. Both of these services are used for testing.

An attacker can exploit this if their target is running echo. The attacker sends a message to a machine running the chargen service (often a networked printer) with the source address spoofed to be the target, with the source port spoofed to be the echo port. The chargen machine will send the characters to the target. The target then echoes those characters back to the chargen machine, which responds by sending more characters to the target. This back-and-forth continues, creating an infinite loop of traffic on the target.

A simple defense against this attack is to make sure the echo service is not running on the server. In general, only essential services should be running.

Motivations for denial of service attacks

- Fun — Some people enjoy taking things down.
- Shady business practices — Some companies try to get a business advantage by taking down their competition at strategic times.
- Profit — An attacker can extract money from some organizations by threatening to DoS them.
- Activism — Some people and organizations will use DoS attacks to take down the sites of organizations they don't agree with.
- Distraction — Sometimes an attacker will use a DoS attack as a distraction while they do something bigger, like hack into a network.

Mitigations for DoS attacks

- One solution is to have a lot of resources and bandwidth. However, this can be expensive.
- There are companies out there that provide DDoS protection.
- Go serverless. If you don't have a server and rely on companies like Google and Amazon to handle things for you, then you have less to worry about with DDoS attacks. However, depending on the type of site you're running, this might not be viable.

If your site is hosted by a hosting company and you are under a DDoS attack, the hosting company may *null-route* your site. That is, they will advertise to other routers that your site is unreachable. That makes it unreachable to everyone. The hosting company does this to protect themselves and their other customers.

One thing to note about DoS attacks is that most of them rely on IP address spoofing. If network administrators perform *egress filtering*, where they don't allow any traffic to leave their network that has a source IP spoofed to be something not on their network, then many of these DoS attacks would be significantly harder.

BGP

The Border Gateway Protocol (BGP) is a routing protocol that *autonomous systems* (ASes) use to talk to each other. An AS is an organization that has a network consisting of a block of real (not private) IP addresses. These are usually fairly large organizations like ISPs and phone companies. Smaller ASes usually pay larger ASes to carry their traffic. The biggest ASes usually *peer* with one another, which means they agree to carry each other's traffic without charging the other.

BGP is what is used to route traffic over the internet and from network to network. Because the ASes are often in competition with each other, and because money is involved, BGP is rather complex. But the basic idea of it is that BGP routers will advertise to other routers which IP addresses they are in charge of, and they tell others whether or not they have a route to get to a specific destination. The problem is that there is not much authentication to this. Other routers implicity trust the information they are given.

So if a BGP router claims that it has a certain block of IP addresses that it really doesn't, the other routers will start routing packets destined for those IPs to the router claiming ownership. This can make large portions of the internet go offline for a while. It has happened multiple times where a country uses BGP to block access to certain sites within the country, and it ended up affecting other countries as well. This can also be used to "steal" IP addresses for a while, which can be used in DoS attacks or to view all the traffic destined for those addresses. People have used this along with DNS cache poisoning to steal cryptocurrency.

It takes some work to pull off an attack like this, as not just anyone has access to these BGP routers. Generally, you would either have to locate one and hack it or convince an employee of an ISP to do your dirty work. Sometimes those employees have accidentally broken parts of the internet by typing things in wrong.

MAC addresses

Every machine connected to the internet has an IP address. In the OSI model, this is at layer 3, the network layer. Below that, at the data link layer (layer 2), we have a different address called a MAC address. MAC is short for Media Access Control. This address is needed in order for a device to send a packet directly to another device. IP addresses are used for routing things around the internet, while MAC addresses are what is actually used when it comes time for one device to send things to another directly.

MAC addresses are written in six groups of two hex digits, usually separated by dashes or colons. For instance, AB:CD:EF:12:34:56 is a typical MAC address. The first three groups are the manufacturer code. In order to make sure no two devices in the world are assigned the same MAC address, each manufacturer of network devices can apply for a block of these addresses. For instance, C0:C1:C0 belongs to Cisco. They can assign MAC addresses starting with these three blocks to each individual device they manufacture. There are sites online that you can use to look up the codes. Wireshark will also show them.

MAC addresses can easily be changed in software. Usually it's just a short command at the terminal. Devices also have recently started using MAC address randomization. This is in response to MAC addresses being used to track people. The way that works is someone could set up a WiFi access point in public or in a business. Whenever someone walks by with their phone, when their phone sees the access point, it by default sends its information to the access point, which includes the phone's MAC address. Someone setting up multiple of these access points can use it to see where someone has been. It does require an additional step of figuring out who belongs to that MAC address, but that can be done in various ways.

ARP

The *Address Resolution Protocol* (ARP) is used to tie IP addresses to MAC addresses. Picture a local network with a router that is connected to the internet. If someone on the local network, say with address 10.0.0.2 requests a web page, the request passes through the router to the website, and the request comes back through the router. In order for the router to send the reply to 10.0.0.2, it needs to know 10.0.0.2's MAC address. ARP is what it uses to do that. The router broadcasts a message to its network asking "Who has 10.0.0.2?" The machine that has it is supposed to respond to the router with its MAC address. The router will then cache the answer for a while to avoid having to do ARP queries with every single packet that comes in, which would get tedious. The answer is cached in something called the *ARP table*.

The problem with ARP is that there is no authentication of replies. Anyone on the network could respond to an ARP reply with their own MAC address, and if their reply gets there before the real reply, then theirs will be taken as correct. This is called *ARP spoofing* or *ARP cache poisoning*. Attackers can use this to read traffic destined for others or to drop traffic destined for others, creating a denial of service.

ARP also has a feature called *gratuitous ARP*, where a machine can basically claim to have an IP address without an ARP query ever having been done. Like most things, gratuitous ARP has a legitimate use, which is when a machine first comes online or when its MAC changes (like if it switched from WiFi to ethernet), it can send this gratuitous ARP to let others know. But any machine can use this feature to claim to own an IP they don't.

There are a few things that network administrators can use to catch ARP spoofing.

- Check the ARP tables to see if the same IP shows up for multiple MAC addresses.
- Set an "alarm" to go off whenever an ARP table entry changes, where the MAC corresponding to an IP changes from what it used to be.
- Send out a "honeypot" fake ARP query, asking who has a particular IP, where that IP is not a real IP on the network. Anyone who answers is likely performing ARP spoofing.

To make ARP spoofing impossible, a network administrator could turn off ARP and configure their ARP tables with only *static* entries. These are fixed entries that the administrator (or their intern) has to manually enter, specifying which IP corresponds to which MAC address. This is tedious and would not work well on networks where new devices are constantly coming online.