https://www.barracuda.com/glossary/email-spoofing

Dr. Gaurav Varshney, IIT Jammu

Email Spoofing Attacks

Email spoofing is the fabrication of an email header in the hopes of duping the recipient into thinking the email originated from someone or somewhere other than the intended source. Because core email protocols do not have a built-in method of authentication, it is commonplace for spam and phishing emails to use said spoofing to trick the recipient into trusting the origin of the message.

The easiest way to spoof mails is for the attacker finds a mail server with an open SMTP (Simple Mail Transfer Protocol) port. SMTP lacks any authentication so servers that are poorly configured have no protection against prospective cyber criminals. It's also the case that there is nothing stopping a determined attackers from setting up their own email servers. This is very common in In cases of CEO/CFO fraud. Attackers will register domains easily confused for the company they are impersonating, where the email is originating from – e.g. "@exarnple.com" instead of "@example.com". Depending on the formatting of the email, it might be extremely difficult for a regular user to notice the difference.

Although most well-known for phishing purposes, there are actually several reasons for spoofing sender addresses. These reasons can include:

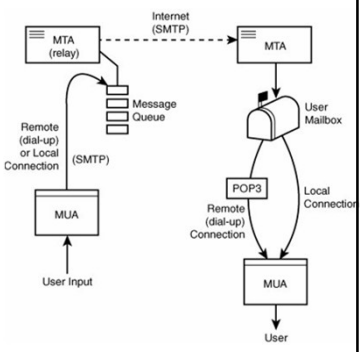
- **Hiding the sender's true identity –** though if this is the only goal, it can be achieved more easily by registering anonymous mail addresses.
- **Avoiding spam block lists.** If a sender is spamming, they are bound to be block listed quickly. A simple solution to this problem is to switch email addresses.
- **Pretending to be someone the recipient knows,** in order to, for example, ask for sensitive information or access to personal assets.
- **Pretending to be from a business the recipient has a relationship with,** as means of getting ahold of bank login details or other personal data.
- **Tarnishing the image of the assumed sender,** a character attack that places the so-called sender in a bad light.
- **Sending messages in someone's name can also be used to commit identity theft,** for example, by requesting information from the victims financial or healthcare accounts.

Email Spoofing: SMTP vs ESMTP

nmu

```
S: 220 smtp.server.com Simple Mail Transfer Service Ready
C: HELO client.example.com
S: 250 Hello client.example.com
C: MAIL FROM:<mail@samlogic.com>
S: 250 OK
C: RCPT TO:<john@mail.com>
S: 250 OK
C: DATA
S: 354 Send message content; end with <CRLF>.<CRLF>
C: <The message data (body text, subject, e-mail header, attachments etc) is sent>
C: .
S: 250 OK, message accepted for delivery: queued as 12345
C: QUIT
S: 221 Bye
```

```
S: 220 smtp.server.com Simple Mail Transfer Service Ready
C: EHLO client.example.com
S: 250-smtp.server.com Hello client.example.com
S: 250-SIZE 1000000
S: 250-AUTH LOGIN PLAIN CRAM-MD5
S: 250-STARTTLS
S: 250 HELP
C: STARTTLS
S: 220 TLS go ahead
C: EHLO client.example.com *
S: 250-smtp.server.com Hello client.example.com
S: 250-SIZE 1000000
S: 250-AUTH LOGIN PLAIN CRAM-MD5
S: 250 HELP
C: AUTH LOGIN
S: 334 VXNlcm5hbWU6
C: adlxdkej
S: 334 UGFzc3dvcmQ6
C: lkujsefxlj
S: 235 2.7.0 Authentication successful
C: MAIL FROM:<mail@samlogic.com>
S: 250 OK
C: RCPT TO:<john@mail.com>
S: 250 OK
C: DATA
S: 354 Send message, end with a "." on a line by itself
C: <The message data (body text, subject, e-mail header, attachments etc) is sent>
S: .
S: 250 OK, message accepted for delivery: queued as 12345
C: QUIT
S: 221 Bye
```



Email Security Policies: Sender Policy Framework (postmarkapp.org)

Sender Policy Framework or SPF

- SPF is an open standard so that the owner of a domain can provide a public list of approved senders.
- This way, receiving mail servers can cross-check that the email originated from a server that has permission to send on your behalf.
- If the message originates from a server that's not on your list, then the receiving server can consider it a fake and treat it accordingly.
- An important aspect to understand about SPF is that it does not validate against the From domain.
- Instead, SPF looks at the Return-Path value to validate the originating server.
- Furthermore, even if a message fails SPF, there's no guarantee it won't be delivered. That final decision about delivery is up to the receiving mail server.

Dr. Gaurav Varshney, IIT Jammu

Email Security Policies: Sender Policy Framework (postmarkapp.org)

•Implementing SPF for your domain

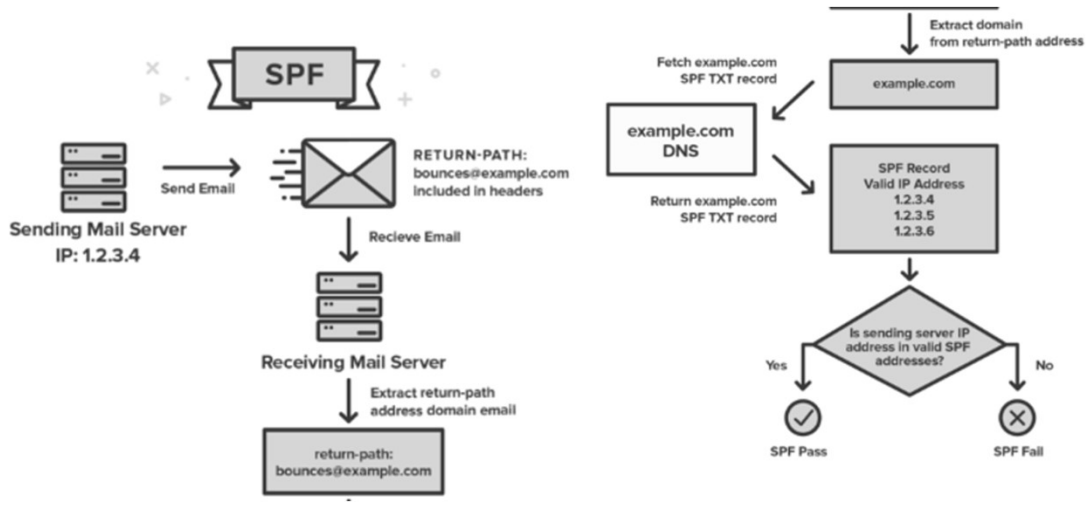
- Implementing SPF only requires a TXT entry in DNS
- The most common mistake when setting up SPF is having multiple SPF TXT entries in your DNS. If you do, the receiving server won't know which SPF TXT entry is the definitive entry. This can result in valid servers failing SPF.
- Example SPF TXT record in DNS

"v=spf1 ip4:169.226.0.0/16 ip4:66.151.109.0/24 include:spf.protection.outlook.com include:verifymyfafsa.com include:_spf.acquia.com ~all"

https://support.google.com/a/answer/10683907?hl=en

- `v=spf1` This states which version of SPF is being used.
- `a` This states that if the domain includes an address record (A or AAAA) for the sender's address, it will match. So, if the IP address of your A record is used to send email, it will pass.
- `mx` The short version is that as long as the email originates from an IP address of the domain's incoming mail servers, then it's a match. The recipient server will check the MX record with the highest priority first.
- `include:` The include statements essentially say to include the values for the SPF records at the specified domain. These records generally specify a set of IP addresses for the service. In this case `spf.mtasv.net` contains the SPF entry for Postmark and `_spf.creatsend.com` represents Campaign Monitor's SPF entry. (For extra credit, you can look at these using the `dig` command in your terminal. Just type `dig txt spf.mtasv.net` and you'll see the Postmark SPF record and the specified IP addresses.)
- `~all` This specifies that everything else should be a "Soft" fail. That means that the message should be accepted but tagged as a soft fail, and the receiving ISP can use that as an additional factor in scoring the message's likeliness of being spam. You could replace the `~` with a `-` and that would indicate that the message should be rejected. However, this is more aggressive and is known to create more issues than it solves (we don't recommend it).

Email Security Policies: Sender Policy Framework (postmarkapp.org)



Dr. Gaurav Varshney, IIT Jammu

Email Security Policies: Domain Keys Identified Email [DKIM]

•**DKIM (Domain Keys Identified Mail)** is an email security standard designed to make sure messages weren't altered in transit between the sending and recipient servers.

•It uses public-key cryptography to sign email with a private key as it leaves a sending server.

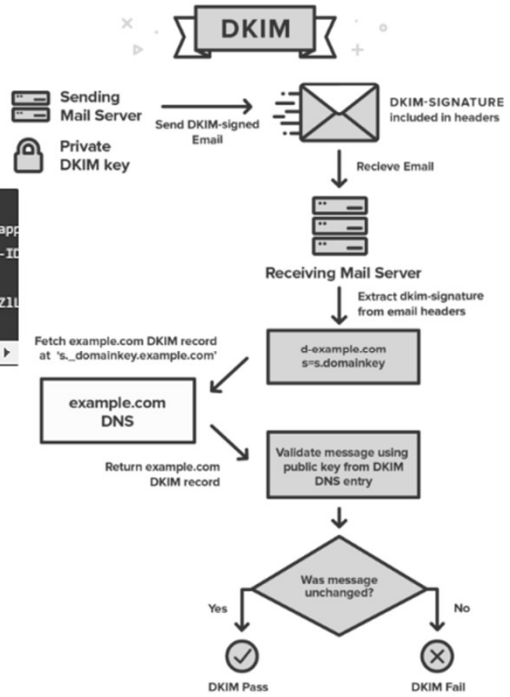
•Recipient servers can then use a public key published to a domain's DNS to verify the source of the message, and that the body of the message hasn't changed during transit.

•Mail servers that don't support DKIM signatures are still able to receive signed messages without any problems.

Email Security Policies: Domain Keys Identified Email [DKIM]

Dr. Gaurav Varshney, IIT Jammu

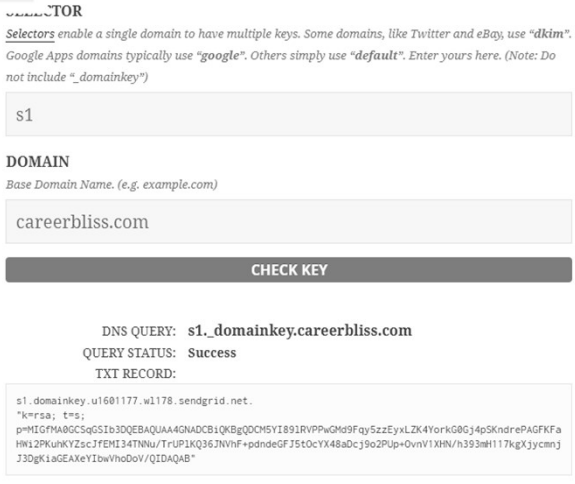
```
DKIM-Signature: v=1; a=rsa-sha1; c=relaxed/relaxed; s=20130519032151pm; d=postmarkapp
h=From:Date:Subject:MIME-Version:Content-Type:Content-Transfer-Encoding:To:Message-ID
i=support@postmarkapp.com; bh=vYFvy46eesUDGJ45hyBTH30JfN4=;
b=iHeFQ+7rCiSQs3DPjR2eUSZSv4i/Kp+sipURfVH7BGf+SxcwOkX7X8R1RVObMQsFcbIxnrq7Ba2QCf0YZlL
```



Email Security Policies: Domain Keys Identified Email [DKIM]

Dr. Gaurav Varshney, IIT Jammu

```
20130519032151pm._domainkey.postmarkapp.com. 3599 IN TXT "k=rsa\;
p=MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCSGLvJw6sZ5YEeFe5tl6D7h6I1E1J5kmM46TEpG99BIMDuX
```



https://www.metaspike.com/leveraging-dkim-email-forensics/#:~:text=The%20recipient%20can%20then%20retrieve,signed%20by%20the%20transmitting%20domain!

Email Security Policies: Domain Keys Identified Email [DKIM] [rfc4871]

Dr. Gaurav Varshney, IIT Jammu

- s= indicates the selector record name used with the domain to locate the public key in DNS. The value is a name or number created by the sender.
- Here is an example of a DNS selector record. The tags shown in this example only appear in this record within DNS and ***not*** in the email header itself: <selector(s=)._domainkey.domain(d=)>. TXT v=DKIM1; k=rsa; p=<public key>

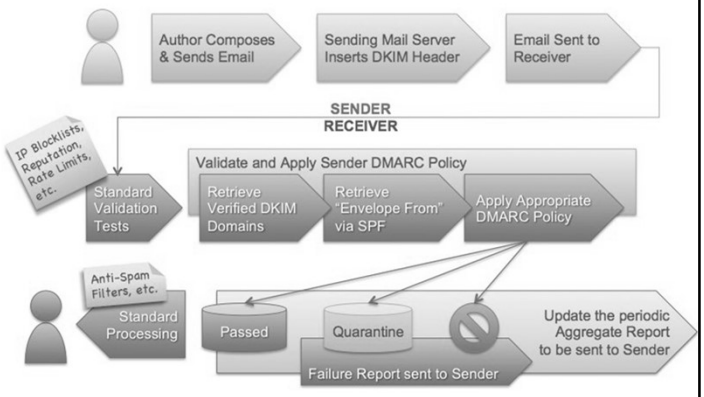
```
In hash step 1, the signer/verifier MUST hash the message body,
canonicalized using the body canonicalization algorithm specified in
the "c=" tag and then truncated to the length specified in the "l="
tag.  That hash value is then converted to base64 form and inserted
into (signers) or compared to (verifiers) the "bh=" tag of the DKIM-
Signature header field.

In hash step 2, the signer/verifier MUST pass the following to the
hash algorithm in the indicated order.

1.  The header fields specified by the "h=" tag, in the order
    specified in that tag, and canonicalized using the header
    canonicalization algorithm specified in the "c=" tag.  Each
    header field MUST be terminated with a single CRLF.

2.  The DKIM-Signature header field that exists (verifying) or will
    be inserted (signing) in the message, with the value of the "b="
    tag deleted (i.e., treated as the empty string), canonicalized
    using the header canonicalization algorithm specified in the "c="
    tag, and without a trailing CRLF.
```

- `DKIM-Signature:` is the header registered for DKIM signed messages.
- `v=1;` the version of DKIM being used by the sending server.
- `a=rsa-sha1;` the algorithm used to generate the hash for the private/public key. There are two officially supported signature algorithms for this hash, rsa-sha1 and rsa-sha256.
- `c=relaxed/relaxed;` sets the canonicalization posture for the sending domain. This regulates whitespace and text wrapping changes in a message. There are two canonicalized postures. `Simple` doesn't allow any changes, and `relaxed` allows common changes to whitespace and header line wrapping. Canonicalization in the header and body can be managed individually, and uses a header/body format.
- `s=20130519032151pm;` is used as a selector for the public DKIM key for verification. Domains can have multiple public DKIM keys, and the selector value makes sure recipient servers are using the correct public key.
- `d=postmarkapp.com;` is the email domain that signed the message. It's important that your DKIM signature use your domain name here, because this bolsters your domain's reputation with ISPs as you send valid email, regardless of the Email Service Provider you use.
- `From:Date:Subject:MIME-Version:Content-Type:Content-Transfer-Encoding:To:Message-ID;` are the headers included with the message when it was cryptographically signed.
- `i=support@postmarkapp.com;` is the identity of the signer and is usually provided as an email address.
- `bh=vYFvy46eesUDGJ45hyBTH30JfN4=;` is the value of a body hash generated before the message headers are signed.
- `b=iHeFQ+7rCiSQs3DPjR2eUSZSv4i/Kp+sipURfVH7BGf+SxcwOkX7X8R1RVObMQsFcbIxnrq78a2QCf0Y2l19iqJf32V+baDI8IykuDztuoNUF2Kk0pawZkbSPNHYRtLxV2CTOtc+x4eIeSeYptaiu7g7GupekLZ2DE1ODHhuP4I=` is the cryptographic signature of all the preceding information from the DKIM-Signature field. This entry is treated as an empty string during the verification process.

Email Security Policies: DMARC [dmark.org]

Dr. Gaurav Varshney, IIT Jammu

DMARC, which stands for "Domain-based Message Authentication, Reporting & Conformance", is an email authentication, policy, and reporting protocol. It builds on the widely deployed SPF and DKIM protocols, adding linkage to the author domain name, published policies for recipient handling of authentication failures, and reporting from receivers to senders, to improve and monitor protection of the domain from fraudulent email.



Email Security Policies: DMARC [dmark.org]


```
"v=DMARC1;p=reject;pct=100;rua=mailto:postmaster@dmarcdomain.com"
```

Tag Name	Purpose	Sample
v	Protocol version	v=DMARC1
pct	Percentage of messages subjected to filtering	pct=20
ruf	Reporting URI for forensic reports	ruf=mailto:authfail@example.com
rua	Reporting URI of aggregate reports	rua=mailto:aggrep@example.com
p	Policy for organizational domain	p=quarantine
sp	Policy for subdomains of the OD	sp=reject
adkim	Alignment mode for DKIM	adkim=s
aspf	Alignment mode for SPF	aspf=r

Relaxed DKIM Alignment tag in a DMARC record (adkim=r)

v=DMARC1; p=quarantine; pct=25 ; rua=mailto:dmarcreports@mxtoolbox.com; ruf=mailto:dmarcfailurereports@mxtoolbox.com; adkim=r;

Strict DKIM Alignment tag in DMARC record (adkim=s)

v=DMARC1; p=quarantine; pct=25 ; rua=mailto:dmarcreports@mxtoolbox.com; ruf=mailto:dmarcfailurereports@mxtoolbox.com; adkim=s;

Pass / Fail Scenarios in Relaxed Alignment

DKIM Domain is	Header From Domain is	Result is
mail.example.com	*example.com*	PASS
example.mail.com	*example.com*	FAIL

DKIM Domain is	Header From Domain is	Result is
mail.example.com	*mail.example.com*	PASS
mail.example.com	*example.com*	FAIL

There are four types of DMARC failure reports that can be sent using the "fo" tag:

- **fo=0**: Generate a DMARC failure report if all underlying authentication mechanisms (SPF and DKIM) fail to produce an aligned "pass" result. (Default)
- **fo=1**: Generate a DMARC failure report if any underlying authentication mechanism (SPF or DKIM) produced something other than an aligned "pass" result. (Recommended)
- **fo=d**: Generate a DKIM failure report if the message had a signature that failed evaluation, regardless of its alignment.
- **fo=s**: Generate an SPF failure report if the message failed SPF evaluation, regardless of its alignment.

Example 1: SPF in alignment

```
MAIL FROM: <sender@example.com>
From: sender@example.com
Date: Fri, Feb 15 2002 16:54:30 -0800
To: receiver@example.org
Subject: here's a sample
```

In Example 1, SPF passes alignment. Why? The RFC5321.MailFrom parameter and the RFC5322.From field (both highlighted above) have identical DNS domains. Therefore, the domain matches (aligns) and will generate a pass result.

Example 2: SPF in alignment (parent)

```
MAIL FROM: <sender@child.example.com>  *This is the RFC5321.MailFrom domain.
From: sender@example.com  *This is the RFC5322.From domain.
Date: Fri, Feb 15 2002 16:54:30 -0800
To: receiver@example.org
Subject: here's a sample
```

In Example 2, the RFC5322.From address (second highlight) includes the Organizational domain of example.com and the RFC5321.MailFrom / Return-Path (first highlight) includes a subdomain of example.com (child.example.com). In default relaxed SPF mode, this message passes SPF alignment because the domains match due to the loose parent / child matching criteria (alignment).

https://mxtoolbox.com/dmarc/spf/spf-alignment#:~:text=SPF%20Alignment%20is%20the%20alignment,found%20in%20the%20other%20header.

Scanning: Introduction

- Scanning is the process of probing and engaging a target network with the intent of revealing useful information.
  - *The information obtained is used in later stages of pen test.*
- Require knowledge of network fundamentals

Dr. Gaurav V

Scanning: Types of Scans

- Port Scans
  - *Sending intentionally crafted packets to know more about a system*
  - *Probes are associated with well known port numbers or those less than or equal to 1024*
- Network Scan
  - *Network scanning is designed to locate all the live hosts on the network*
  - *Ping sweeps, Nmap, Angry IP*
- Vulnerability Scan
  - *Used to identify vulnerability or weakness on a target system and its services.*
  - *Tenable Nessus, Rapid 7's Nexpose.*

Checking for Live Systems on a Network

- Pingining
  - *Ping <target IP>*
  - *Ping <target hostname>*
    - Requires DNS resolution
- A very strong tool to know is NMAP
  - *Nmap –sP –v <target IP Address>*

Dr. G

Scanning: NMAP

- Nmap ("Network Mapper") is a free and open source (license) utility for network discovery and security auditing.
- Nmap uses raw IP packets in novel ways to determine
  - *what hosts are available on the network,*
  - *what services (application name and version) those hosts are offering,*
  - *what operating systems (and OS versions) they are running,*
  - *what type of packet filters/firewalls are in use, and dozens of other characteristics.*
- The output from Nmap is a list of scanned targets, with supplemental information on each depending on the options used.

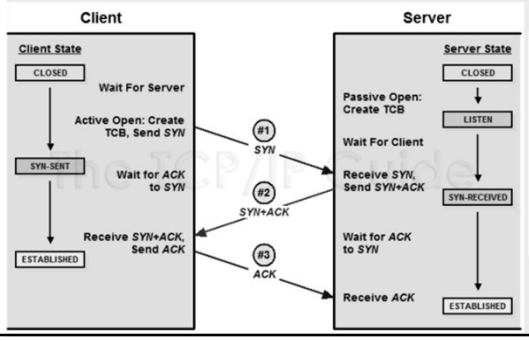
Scanning: Hping3

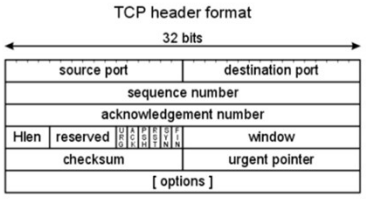
- Another packet crafting and network scanning tool is Hping3
  - *It is a command line based TCP/IP packet crafter.*
  - *Hping3 –1 <domain name> it is a way to ping the system by sending a ICMP request message*
  - *Hping3 –c 1 –V –p 80 –s 5050 –A <domain name> ACK packet is sent to check whether the port is open*

Dr. G

Scanning: Understanding TCP port scanning

- Three Way Handshake





Dr. Gaurav Varshney, IIT Jammu

Scanning: Tree of TCP Scans

- Full Open Scan
  - *Complete the full TCP handshake process*
    - Nmap –sT <ip address or range>
    - Return report of all ports where a TCP connection is established
    - A closed port will return RST flag in the packet.
- Half Open Scan
  - *Send RST to the received SYN+ACK packet*
  - *Server has less to log as a full TCP connection did not happened.*
    - Nmap –sS <ip address or range>
- Xmas Tree Scan
  - *A single packet with URG, PSH and FIN all set to on.*
  - *RST packet response means port is closed*
  - *Lack of response means port is open.*
    - Nmap –sX –v <target IP Address>

Dr. Gaurav Varshney, IIT Jammu

Scanning: Tree of TCP Scans

- FIN Scan
  - *Does not get detected by Firewalls and routers*
    - Send a FIN Packet
      - *No Response means the port is open*
      - *RST means port is closed*
      - *Nmap –sF <target IP address>*
- NULL Scan
  - *Set no flags in the packet*
  - *No response means the port is open*
  - *RST means the port is closed*
    - Nmap –sN <target IP address>
- ACK Scanning
- Stateful and Stateless Firewalls
  - *Nmap –sS –T4 –A –f –v <targetIPAddress>*

Scanning: UDP Scans

- UDP
  - *Connectionless Protocol*
  - *Send UDP packet to a port*
    - Open will send no response
    - Closed will send ICMP port unreachable message

Dr. Gaur

Scanning: OS Fingerprinting

- OS fingerprinting
  - *It tries to identify the operating system by the unique fingerprint it returns.*
  - *Those fingerprints can be compared to a database of known fingerprints to determine what OS the target is using.*
  - *Common techniques are based on analyzing the following:*
    - IP TTL Values: Linux 64, FreeBSD 64 Windows XP 128, Cisco Router (iOS 12.4) 255
    - TCP Window Size Linux 5840, FreeBSD 65535, Windows XP 65535, Cisco Router 4128
    - TCP options (generally, in TCP SYN and SYN+ACK packets)
    - DHCP requests
    - ICMP requests
    - HTTP packets (the User-Agent field)
    - Running services
    - Open port patterns

Scanning: OS Fingerprinting

- OS fingerprinting
  - *Active Fingerprinting with Nmap*
    - Nmap fires a range of TCP and UDP packets at a target system
    - Looks for responses to be returned
    - The responses are analyzed in depth to look for clues as to the nature of the OS.
    - Nmap compares the findings with the OS fingerprint database.
      - *Nmap –O <ip address>*
    - It provides the OS information that it guessed as well as the confidence with which it has guessed that information.
  - *Passive fingerprinting*
    - Just see the TTL and Window size in packets via eavesdropping.
    - In Linux systems p0f is a packet sniffing tool which can be used for such type of fingerprinting.
      - *Sudo p0f –i eth0*
    - Passive fingerprinting takes more time than active fingerprinting

TCP SYN Flooding Attack

Dr. Gaurav Varshney, IIT Jammu

When the server receives the initial SYN packet it uses a special data structure called Transmission Control Block (TCB) to store the information about this connection.

Therefore, the server stores the TCB in a queue that is only for the half-open connections. After the server gets the ACK packet from the client, it will take this TCB out of the queue, and store it in a different place.

If the final ACK packet does not come, the server will resend its SYN + ACK packet. If the final ACK packet never comes, the TCB stored in the half-open connection queue will eventually time out, and be discarded.

There are several common events that can lead to the dequeue of a TCB record. First, if the client finishes the three-way handshake process, the record will be dequeued, because it is not half-open anymore.

Second, if a record stays inside for two long, it will be timed out, and removed from the queue. The timeout period can be quite long (e.g., 40 seconds).

Third, if the server receives a RST packet for a half-open connection, the corresponding TCB record will be dequeued.

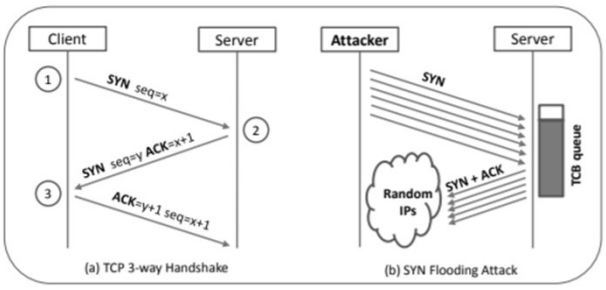


Figure 13.3: TCP Three-way Handshake Protocol and SYN Flooding

TCP SYN Flooding Attack

Dr. Gaurav Varshney, IIT Jammu

When flooding the target server with SYN packets, attackers need to use random source IP addresses; otherwise, their attacks can be easily blocked by firewalls.

When the server replies with SYN + ACK packets, chances are that the replies may be dropped somewhere in the Internet because the forged IP address may not be assigned to any machine or the machine may not be up at the moment.

If attackers can fill up this queue quickly, there will be no space to store the TCB for any new half-open connection;

basically, the server will not be able to accept new SYN packets.

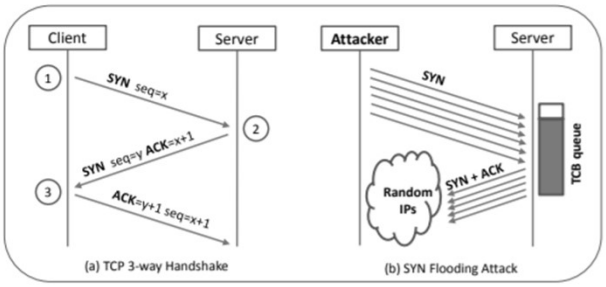


Figure 13.3: TCP Three-way Handshake Protocol and SYN Flooding

https://web.ecs.syr.edu/~wedu/seed/Book/book_sample_tcp.pdf

TCP SYN Flooding Attack: Solution

Dr. Gaurav Varshney, IIT Jammu

An effective way to defend against SYN flooding attacks is a technique called SYN cookies, which was originally invented by Daniel J. Bernstein in September 1996

The idea of the SYN cookies mechanism is to not allocate resources at all after the server has only received the SYN packet; resources will be allocated only if the server has received the final ACK packet.

SYN Cookies are constructed and sent as initial sequence number in the TCP packet. “SYN cookies are particular choices of initial TCP sequence numbers by TCP servers”

After a server has received a SYN packet, it calculates a keyed hash from the information in the packet, including the IP addresses, port number, and sequence number, using a secret key that is only known to the server.

This hash value H will be used as the initial sequence number placed in the server’s SYN+ACK packet sent back to the client. The value H is called SYN cookies.

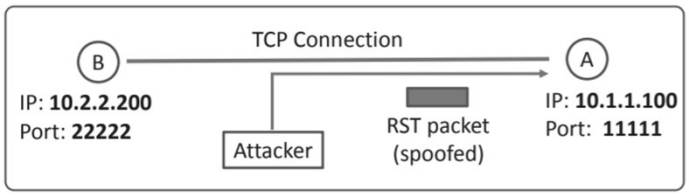
The server sends the initial sequence number and does not reserves any resources for the connection at the server and waits for an ACK-1 that can match the initial sequence number sent on the connection to do the final connection establishment and allocation of resources.

https://web.ecs.syr.edu/~wedu/seed/Book/book_sample_tcp.pdf

https://www.net.in.tum.de/fileadmin/TUM/NET/NET-2019-06-1/NET-2019-06-1_14.pdf

Dr. Gaurav Varshney, IIT Jammu

TCP RESET ATTACK



(a) Attack diagram

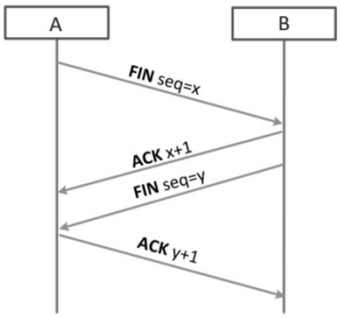


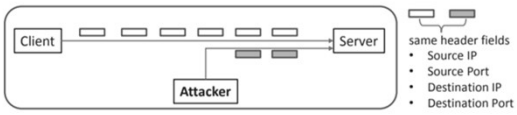
Figure 13.4: TCP FIN Protocol

It should be noted that the success of the attack is very sensitive to the sequence number. The number that we put in the spoofed packet should be exactly the number that the server is waiting for.

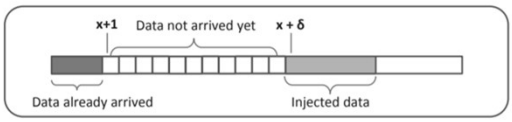
If the number is too small, it will not work. If the number is large, according to RFC 793 [Postel, 1981], it should be valid as long as it is within the receiver's window size,

TCP Session Hijacking ATTACK

When a connection is established between two hosts, the connection is supposed to be used only by these two hosts. If an attacker can inject his/her own data into this connection, the connection can essentially be hijacked by the attacker, and its integrity will be compromised. In this section, we discuss how such an attack works.



(a) Injecting data into a TCP connection



(b) Receiver's TCP buffer and sequence numbers

Figure 13.7: TCP Session Hijacking Attack

If the δ is too large, it may fall out of the boundary of the buffer. In this case, the spoofed packet will be discarded.

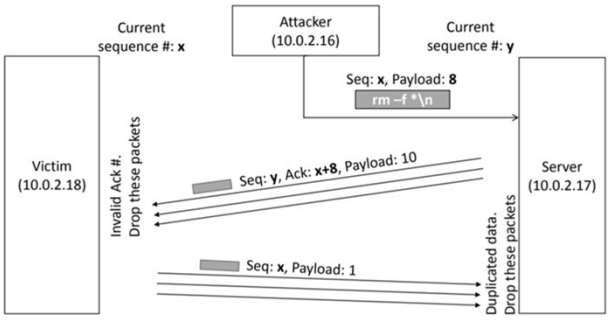
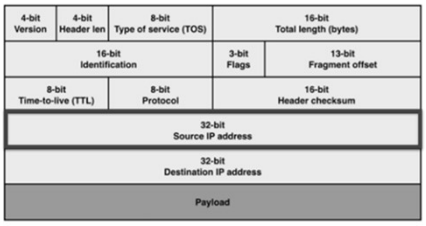


Figure 13.10: Why the connection freezes

Attacks on IP

Dr. Gaurav Varshney, IIT Jammu

Attacks on IP



Source-spoof

There is nothing in IP that enforces that your source IP address is really "yours"

Source spoofing

- Why source-spoof?
  - Consider DoS attacks: generate as much traffic as possible to congest the victim's network
  - Easy defense: block all traffic from a given source near the edge of your network
  - Easy countermeasure: spoof the source address
- Challenges won't help here; the damage has been done by the time the packets reach the core of our network
- Ideally, detect such spoofing *near the source*

Egress filtering

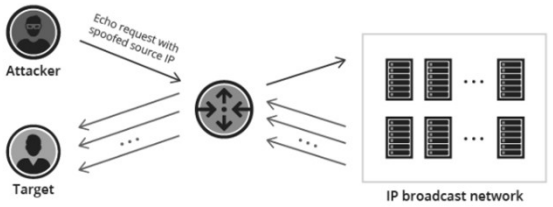
- The point (router/switch) at which traffic *enters* your network is the *ingress point*
- The point (router/switch) at which traffic *leaves* your network is the *egress point*
- You don't know who owns all IP addresses in the world, but you *do* know who in *your own network* gets what IP addresses
  - If you see a packet with a source IP address that doesn't belong to your network trying to cross your egress point, then *drop it*

https://www.cs.umd.edu/class/spring2017/cmsc414/s17-stuff/14-s17-internet-ip-layer.pdf

Smurf DDOS Attack

https://www.imperva.com/learn/ddos/smurf-attack-ddos/

In an IP broadcast network, an ping request is sent to every host, prompting a response from each of the recipients. With Smurf attacks, perpetrators take advantage of this function to amplify their attack traffic.



A Smurf attack scenario can be broken down as follows:

1. Smurf malware is used to generate a fake Echo request containing a spoofed source IP, which is actually the target server address.
2. The request is sent to an intermediate IP broadcast network.
3. The request is transmitted to all of the network hosts on the network.
4. Each host sends an ICMP response to the spoofed source address.
5. With enough ICMP responses forwarded, the target server is brought down.

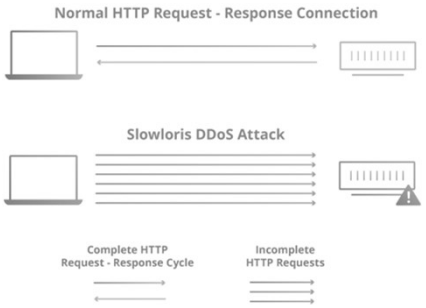
- Disable IP-directed broadcasts on your router.
- Reconfigure your operating system to disallow ICMP responses to IP broadcast requests.
- Reconfigure the perimeter firewall to disallow pings originating from outside your network.

Read about
Ping of Death – Oversized ICMP packets
Teardrop – Overlapping fragmented packets

HTTP Slowloris Attack: Low and Slow DoS

- Read about it.
- https://nmap.org/nsedoc/scripts/http-slowloris-check.html
- https://www.imperva.com/learn/ddos/slowloris/

```
IDs:  CVE:CVE-2007-6750
  Slowloris tries to keep many connections to the target web server open and hold
  them open as long as possible.  It accomplishes this by opening connections to
  the target web server and sending a partial request. By doing so, it starves
  the http server's resources causing Denial Of Service.

Disclosure date: 2009-09-17
References:
  http://ha.ckers.org/slowloris/
  http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2007-6750
```



HTTP Slowloris Attack

An analysis of an *HTTP GET* request helps further explain how and why a slow HTTP DoS attack is possible. A complete *HTTP GET* request resembles the following:

```
GET /index.php HTTP/1.1[CRLF]
Pragma: no-cache[CRLF]
Cache-Control: no-cache[CRLF]
Host: testphp.vulnweb.com[CRLF]
Connection: Keep-alive[CRLF]
Accept-Encoding: gzip,deflate[CRLF]
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/28.0.1500.63 Safari/537.36[CRLF]
Accept: */*[CRLF][CRLF]
```

The CRLF (*Carriage Return + Line Feed*) is a sequence of non-printable characters that is used to denote the end of a line. Similar to text editors, an HTTP request uses a [CRLF] at the end of a line to start a fresh line and two [CRLF] sequences to denote a blank line. The HTTP protocol defines a blank line as the completion of a header. A slow HTTP DoS attack takes advantage of this by never sending a finishing blank line to complete the HTTP header.

While some thread-based servers such as Apache use a *timeout* when they wait for incomplete HTTP requests, it is set to 300 seconds by default and reset as soon as the client sends the rest of the data. To make matters worse, a slow HTTP DoS attack is not commonly detected by *Intrusion Detection Systems* (IDS) because it does not contain any malformed requests. The HTTP request will seem legitimate to the IDS.

https://www.acunetix.com/blog/articles/slow-http-dos-attacks-mitigate-apache-http-server/

Network Attacks [IP/DHCP/ICMP/TCP]

Dr. Gaurav Varshney, IIT Jammu

Smurf Attack: Sending ICMP Echo ping request from spoofed source IP address to a broadcast destination address in the network.

- Sudo hping3 –icmp 10.0.2.255 –a spoofedaddress(victimip)

IP fragmentation based attacks

- Sudo hping3 10.0.2.15 –f –morefrag

ICMP Source Quench Message Spoofed

- Sudo hping3 - - icmp - - icmptype 4 - - icmpcode 0 192.168.1.2 –a 192.168.1.12

RESET a TCP connection

- Sudo hping3 10.0.2.15 –setseq 1234 –setack 1201 –R

- ICMP Redirect Spoofing
  - Sudo hping3 –icmp –icmptype 5 –icmpcode o www.iitjammu.ac.in –a 192.168.0.1
- SYN Floooding
  - Sudo hping3 –S www.example.com –p 80 –c 100
  - Solution SYN Cookies
- ARP cache poisoning
- MAC Flooding Attack
- Port Stealing Attack