Anatomy of a DNS Cache Poisoning Attack
Introduction
The purpose of this presentation is to dissect the “Domain Name System (DNS) Cache Poisoning” Cyber attack. We will cover:

- Real-world examples of DNS cache poisoning
- A definition of DNS
- An overview of the technology behind DNS
- The role played by DNS Cache within DNS
- Ways in which Cyber-criminals exploit vulnerabilities in the DNS to steal information from unsuspecting victims
- Best practices in preventing attacks
About the presenter:

Name: Boyan Lazarevski
Profession: IT Operations Specialist
Experience: System Administration, Network Security
Interests: Cybersecurity, Computer Hardware, Retro-computing
Defining the Problem
On January 26, 2015, a hacker group managed to redirect visitors of the Malaysia Airlines (MAS) official website to another site displaying malicious content.

MAS denied that their systems had been hacked and claimed that their web servers were intact despite news reports indicating it was a hacking incident.
MAS was right. Their own hardware/servers were not actually “hacked”. Instead, they had fallen victim to a hack attack that indirectly affected them.

This attack is known as “DNS Cache Poisoning”.

The attackers (or Cyber-criminals) abused the cached IP address in the DNS server to redirect their web site visitors to a completely different web page.
- April 2018, a major DNS cache poisoning attack compromised Amazon’s DNS servers, redirecting users to malicious web sites.

- November 2011, a large-scale attack on ISPs in Brazil rerouted traffic from popular sites (including Google, Gmail and Hotmail) to a web page that installs malicious Java applets.

- December 2009, hackers redirect traffic from Twitter to their own web site.

- July 2008, a major DNS cache poisoning attack on AT&T DNS servers. Many websites become unavailable to millions of web users.
What is the DNS?
- Domain Name System (DNS)
- Communications Protocol, part of the “TCP/IP suite”
- A critical “building block” of the Internet.
- Web browsers, e-mail services, and social networks rely 24/7 on its availability.

- However… it is also a critical attack vector!
(Perhaps the most overlooked internet service in terms of Cybersecurity)
The DNS Resolution Process
The user types "www.owasp.org" into a web browser, which in turn creates a DNS query and sends it to the default DNS Recursive Resolver server.
The DNS Recursive Resolver is the first stop for the DNS query. In most cases it is a server hosted by the Internet Service Provider (ISP).
After receiving a DNS query from a host computer’s web browser, the DNS resolver will send a request to a DNS Root Server.
The DNS Root server responds to the DNS resolver with the address of a DNS Top Level Domain (TLD) server that stores information for .org domains.
The DNS resolver makes a request to the .org DNS TLD server.
The DNS TLD server responds with the IP address of an DNS Authoritative Name server.
The DNS Resolver sends a query to the DNS Authoritative Name server.
The DNS Authoritative Name server holds the actual DNS IP records, and sends the IP address for “owasp.org” to the DNS Resolver.
The DNS Resolver sends to host computer a DNS response that contains the IP address of the website initially requested.
The user’s web browser uses the IP address received from the DNS response to start a Transport Layer Security (TLS) encrypted connection session with OWASP’s Web Server.
- Server responds
- A secure connection over port 443 is established
- The Web Server begins transmitting thousands of packets of data containing web page resources.
The user’s (client’s) computer receives the individual TCP packets, assembles them, and reconstructs the HTTP data sent by the Web Server over the secure connection.
- OWASP’s web page is displayed on the user’s computer monitor.

- This concludes the DNS Resolution process.
Structure of DNS Messages
DNS Query

Frame 185: 80 bytes on wire (640 bits), 80 bytes captured (640 bits) on interface 0
Ethernet II, Src: VMware_cc:79:b8 (00:0c:29:cc:79:b8), Dst: VMware_fa:b7:c0 (00:58:56:fa:b7:c0)
User Datagram Protocol, Src Port: 36611, Dst Port: 53
Domain Name System (query)
Transaction ID: Bxbd63

Flags: 0x0100 Standard query
0... ....... ... = Response: Message is a query
.000 0... ....... = Opcode: Standard query (0)
.... .0 ... .... = Truncated: Message is not truncated
.... ....1 .... = Recursion desired: Do query recursively
.... ........ = Z: reserved (0)
.... .......0 .... = Non-authenticated data: Unacceptable

Questions: 1
Answer RRs: 0
Authority RRs: 0
Additional RRs: 1

Queries
owasp.org: type A, class IN
Name: owasp.org
[Name Length: 9]
[Label Count: 2]
Type: A (Host Address) (1)
Class: IN (0x0001)

[Additional records]

0000 00 50 56 fa b7 c0 00 0c 29 cc 79 b8 00 00 45 00 .PV......... y....
0010 00 42 40 c4 40 60 49 11 b7 c1 c6 a8 e9 d1 c0 a8 .80 0@.........
0020 e0 02 8f 03 00 35 08 2e 42 65 0b 68 b1 00 00 01 .5. Bec......
0030 00 00 00 00 00 00 01 05 6f 77 61 73 70 03 6f 72 67 .0 wasp.org
0040 00 00 01 00 01 00 00 00 00 00 00 00 00 00 00 00 ...........
DNS Response

Frame 187: 96 bytes on wire (768 bits), 96 bytes captured (768 bits) on interface 0
Ethernet II, Src: Vmware_fa:b7:c0 (00:50:56:fa:b7:c0), Dst: Vmware_cc:79:b8 (00:0c:29:cc:79:b8)
User Datagram Protocol, Src Port: 53, Dst Port: 36611

Domain Name System (response)

Transaction ID: 0xbd63

Flags: 0x8100 Standard query response, No error
1... ---- ---- ---- = Response: Message is a response
.000 0... ---- ---- = Opcode: Standard query (0)
.... .0... ---- ---- = Authoritative: Server is not an authority for domain
.... .0. ---- ---- = Truncated: Message is not truncated
.... .0. ---- ---- = Recursion desired: Do query recursively
.... .0. ---- ---- = Recursion available: Server can do recursive queries
.... .0. ---- ---- = Z: reserved (0)
.... ---- .0... ---- = Answer authenticated: Answer/authority portion was not authenticated by the server
.... ---- .0000 ---- = Non-authenticated data: Unacceptable

Questions: 1
Answer RRs: 1
Authority RRs: 0
Additional RRs: 1

Queries

Answers

owasp.org: type A, class IN, addr 104.130.219.202
  Name: owasp.org
  Type: A (Host Address) (1)
  Class: IN (0x0001)
  Time to live: 5
  Data length: 4
  Address: 104.130.219.202

Additional records

[Request In: 185]
[Time: 0.125611149 seconds]
- There are many public DNS servers that the DNS Resolver can use to speed up the resolution process.

- However it's much faster to have a local copy (even a temporary one) of the DNS "phone book." This is exactly where DNS caches come into play.

- Each operating system (OS) (Windows and MAC OS by default, and UNIX via a Daemon) stores a temporary DNS cache database that contains a list of all recently accessed domain names and the addresses that DNS calculated for them the first time a request was made.
In a local DNS cache entry, the "A" record contains the IPv4 address for the given website name. IPv6 addresses use the “AAAA” record. The DNS cache stores this address, the requested website name, and several other parameters from the host DNS entry.
How does the “Poisoning” of the DNS Cache occur?
A DNS cache becomes “poisoned” or polluted when unauthorized domain names or IP addresses are inserted into it. The corruption of the DNS cache can be achieved either by:

- Computer malware, or
- Network attacks that insert invalid DNS entries into the cache.
Reminder: when a user tries to browse to a website, the computer queries its local DNS cache for the IP address. If the DNS cache has a copy of the record, it replies. If not, it queries an “upstream” DNS server, relays the results back to the end user, and caches them for next time.
Attackers have devised a way to “spoof” DNS responses - to forge DNS responses that look as if they are coming from legitimate DNS servers. If an attacker successfully spoofs a DNS response, it can make the receiving DNS server cache a poisoned record.

But how does that help the attackers?
By being redirected to a wrong destination, we may end up suffering from a phishing attack – which is the ultimate goal of this type of Man-in-the-Middle attacks!
For example:

- An attacker learns that the Department of Computer Science at the University of Ghana regularly visits the OWASP website to check the most updated vulnerability databases and get up to speed with the latest development in web app security.

- The attacker poisons the University’s DNS Resolver, sending users to the attacker’s web site.

- The attacker creates a legitimate looking OWASP login page to get users to enter their credentials.

- The attacker could have also relayed website traffic to the real server (“Man-in-the-middle” style), so no one notices.

- This approach can be used to obtain bank account information...
But wait a moment...

- What about the “Transaction ID” that we mentioned previously?
- Don’t DNS Queries and Responses contain Transaction IDs that are read by the Application layer of the user’s computer?

Well yes, but there are two problems:

- The Transaction is a 16-bit binary number \(2^{16} = \text{any number between 0 and 65536}\).
- DNS servers accept near-simultaneous responses to requests, allowing attackers to make multiple guesses about the transaction ID (something like a brute force attack against a password).
DNS Spoofing Demonstration
The demonstration is carried on a LAN network composed of the following three elements:

- Default Gateway (IP address 192.168.224.2)
- Attacker computer (IP address 192.168.224.13)
- Target computer (IP address 192.168.224.211)

The application used to carry out the DNS Spoofing is **Ettercap**: a free and open source network security tool for man-in-the-middle attacks.
Prepare for the attack by configuring the attack parameters:

- Step 1: Make a fake OWASP HTML web-page (phishing web-page). Set it up on an Apache Web Server hosted on the Attacker computer (the fake web-site will be accessed by typing the IP address of the Attacker computer onto a browser).

- Step 2: Go to the Ettercap directory and open the “etter.dns” using a text editor. At the bottom of the file, add the name to the website that we want to attack (in this case, “www.owasp.org”) and also add the IP that we want the Target computer to be redirected to (in this case, the IP address of the Attacker computer, hosting the fake web-page). See the following screenshot for illustration.
The attack parameters that were added manually are marked with the red square:

```
LAB-PC*  WINS  127.0.0.1

owasp.org   A   192.168.224.13
*.owasp.org  A   192.168.224.13
www.owasp.org A   192.168.224.13
```

# vim:ts=8:noexpandtab
Step 3: Open Ettercap in sudo mode and select Sniff>Unified Sniffing.
Step 4:
Go to Hosts>Scan for Hosts to find devices connected to the LAN
Step 5:
Go to Hosts>Hosts List to display the list of devices
Step 6:
From the list, select the IP address of the Target computer and add it to Target 1 and also select the IP address of the default gateway and add it to Target 2.
Step 7:
Go to Plugins>Manage the Plugins
Step 8:
From the list of plugins select “dns_spoof”
Step 9:
The plugin activates
the process of
bombarding the
target machine with
fake DNS responses
that resolve
owasp.org to IP
address
192.168.224.13
(where the fake web-
page is hosted by
web server on the
Attacker machine)
As a result, instead of being directed to the real web-page...
Welcome to OWASP's Official Web Page

We are sorry to inform you that we are currently undergoing a company-wide rebranding process, which also affects the design of our website. Please send any important information and correspondence to the following e-mail address

Sincerely yours, the OWASP Team

Copyright ©2019 All rights reserved

...the Victim is directed to the fake web-page (notice that the browser displays the same URL!)
Please Note:

- We have discussed DNS Cache Poisoning methodology used to compromise DNS Cache records stored on users’ computers.

- However, these principles also apply for tampering with the cached DNS records on DNS Resolver servers!

The following slide shows a topological illustration of the attack.
Avoiding DNS Cache Poisoning
Although DNS Poisoning sounds scary, there are ways to (try to) prevent it. Let’s first look at some common measures that mainly involve vigilance while browsing the internet:

- Keep your Antivirus/Anti-malware apps “Cooking and Up-to-Date”
- If you have the possibility, browse the internet via a Virtual Machine
- Don’t download suspicious files. If you insist, do it through a “sand-boxed app” or a Virtual Machine
- Use a respected DNS Server and a reputable ISP
- Always double-check websites you visit (check if there is HTTPS encryption)
- Flush computer DNS cache as well as DNS cache stored in the router
Additional long-term prevention measures include:

- **Virtual Private Network (VPN):** A service that encrypts all the internet traffic going to and from a device and routes it through an intermediary server in a location of the user’s choosing.

- **Encrypted DNS:** Apps that encrypt DNS traffic between the user and an OpenDNS nameserver (similar to how SSL encrypts traffic to websites that use HTTPS).
Security mechanisms developed for DNS server operators:

- **UDP Source Port Randomization** (UDP SPR): What this does is setting the UDP source port randomly, so an attacker would have to guess both the transaction ID and the source port in a short time window - which is usually not feasible (since they would need to make $2^{32}$ combinations).

- **DNS Security Extensions** (DNSSEC): It is a protocol designed to create a unique cryptographic signature and store it alongside other DNS records. Thus, DNSSEC provides DNS with an additional methods of verification by digitally signing the DNS information. This is done on all levels of the DNS Resolution process.
The End ?
References

https://www.pcworld.com/article/149126/dns_attack_writer.html
https://medium.com/metacert/major-dns-spoofing-hack-affects-amazon-web-services-157e3565c844
https://www.lifewire.com/what-is-a-dns-cache-817514
https://www.varonis.com/blog/what-is-dns
https://www.makeuseof.com/tag/what-is-dns-cache-poisoning
https://www.cloudflare.com/learning/dns/dns-security