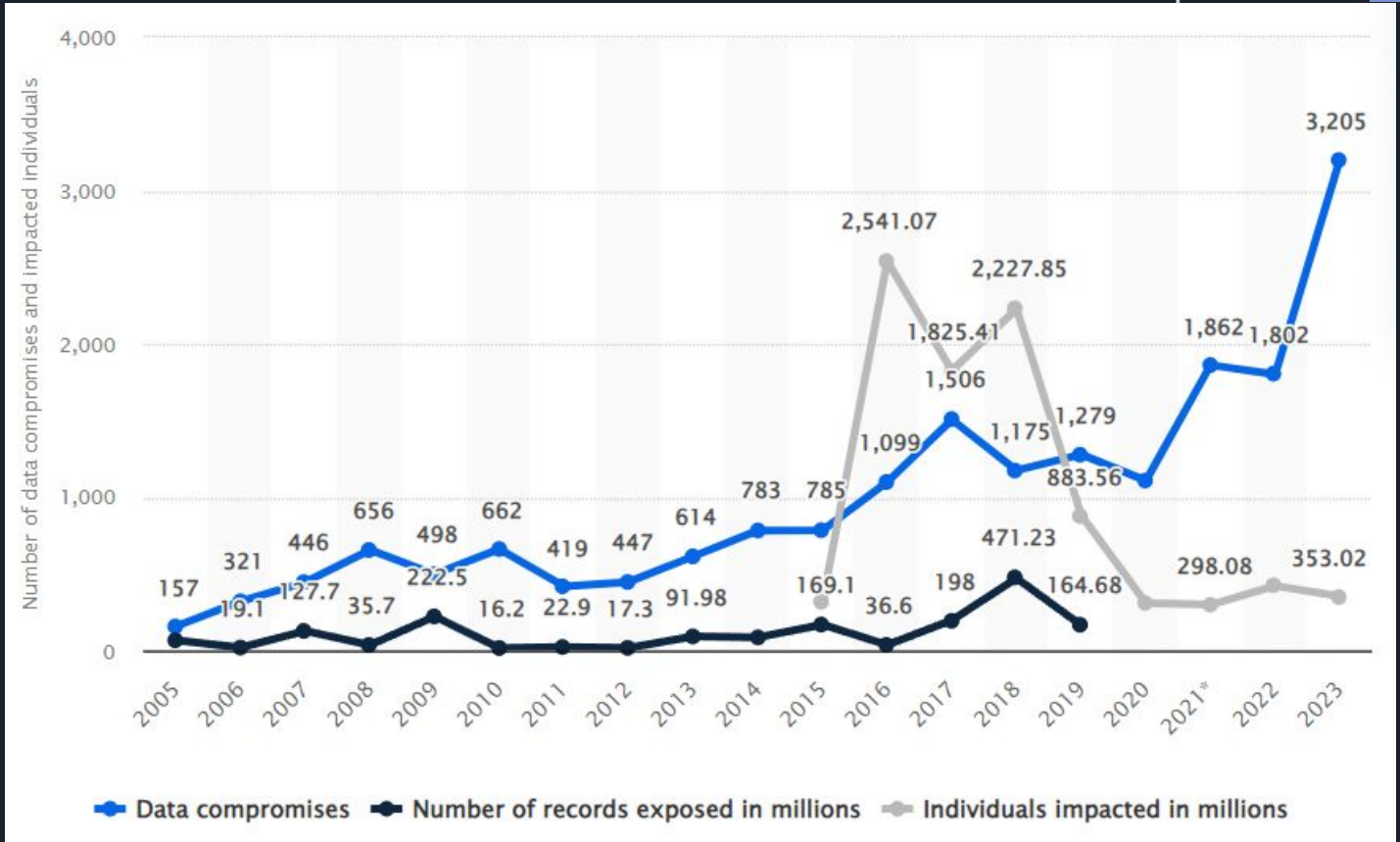
Web Security

Ethan Greene and Brett Widholm

Data compromises from 2005 - 2023

Statistic acquired from Statista



What web security?

For clients:

- talk about hesitation to what sites you visit/sign up for/give your data to
- Regularly checking sites such as haveibeenpwned to see if you are involved in data breaches or leaks
- Choose smart passwords and differentiate between different accounts you own
- Update passwords if you know you have been compromised

For hosts:

- Necessity to keep client data confidential and secure
- Protect hardware and software from attackers
- Ensure service is available to clients

Importance of web security

- Confidentiality
  - Sensitive data being leaked such as passwords, bank information, etc.
- Integrity
  - Loss of trust and credibility for either an individual or a company.
- Availability
  - Damages reputation and can lead to clients picking different options.
  - Loss of sales and resources in the time shut down.

Data leaks and data breaches

Data breaches

- Requires intent to do harm
- Attackers exploit vulnerabilities within the service in order to gain access to sensitive data

Data leaks

- Does not always require intent to do harm
- Requires the exposure of sensitive data to an audience
- Could be due to the host's incompetence or could be a result of someone trying to cause damage

EQUIFAX Data Breach, 2017

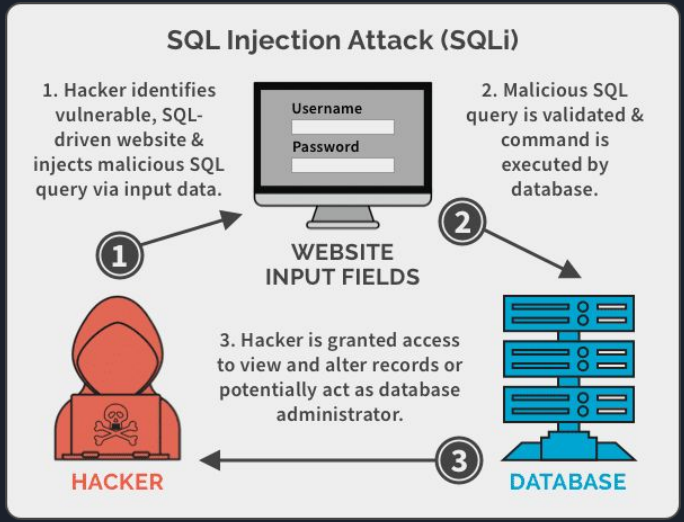
- Company that gathers credit history information in order to formulate credit reports.
- In September of 2017 Equifax announces a leak of which 147 million of their users personal information was exposed.
- 4 members of the chinese military later charged.
- Settlement of $425 million dollars for victims affected by this hack.

SQL injections

- An SQL injection is a series of strange characters that cause the code to run in unexpected ways
- These strange characters modify the behavior of the code which could grand unauthorized users access when they otherwise should not

Any database management system is vulnerable to this attack

- Examples include: MySQL, Oracle, Protage, etc.



SQL injections

- SQL injections have been known to be a problem since 1998 when initially discovered by Jeff Forristal
- His findings were published to Phrack magazine but was largely overlooked until 2002
- Due to the rise in computer viruses and worms prior to 2002, a demand for improved cybersecurity gave exposure to the attack

Companies affected by SQL injections:

- Yahoo! (2012) - over 450,000 emails and passwords leaked online
- Epic Games (2016) - over 800,000 accounts compromised
- Sony (2011) - over 1 million passwords stolen and leaked
- LinkedIn (2012) - around 167 million accounts compromised

SQL injection example

```
1  > // {…
16   app.post('/login', function (req, res) {
17       var username = req.body.username;
18       var password = req.body.password;
19       var query = "SELECT name FROM user where username = '" + username + "' and password = '" + password + "'";
20
21       console.log("username: " + username);
22       console.log("password: " + password);
23       console.log('query: ' + query);
24
25       db.get(query , function(err, row) {
26
27           if(err) {
28               console.log('ERROR', err);
29               res.redirect("/index.html#error");
30           } else if (!row) {
31               res.redirect("/index.html#unauthorized");
32           } else {
33               res.send('Hello <b>' + row.name + '</b><br /><a href="/index.html">Go back to login</a>');
34           }
35       });
36
37   });
38
```

Invalid Username or Password

Hello **App Administrator**
Go back to login

SQL injection example

```
username: hi "' or '1' = '1'; --
password: thispassworddoesnotmatter
query: SELECT name FROM user where username = 'hi "' or '1' = '1'; --' and password =
'thispassworddoesnotmatter'
```

SQL injection example

While this was a basic example, SQL injections come in many forms

- Boolean based injection
  - Uses boolean operations to manipulate the statement
  - Ex: hi' or '1' = '1'--;
- Error based injection
  - Uses information from error codes produced to gain knowledge about the database
  - See figure 1
- Union based injection
  - Uses union command to retrieve data from various columns in a table within the database
  - Can retrieve data if you know the table name as well as column names

Figure 1

```
1 GET / HTTP/1.1
2 Host: 0a2d00b5049d812583dad372002b00bd.web-security-academy.net
3 Cookie: TrackingId='||cast((select username from users limit 1) AS INT)--; session=
  3JCKmjbonsoE3mtTlwM5F71vMMjytgXd

    ERROR: invalid input syntax for type integer: "administrator"
  </h4>
```

```
User ID:

hation_Schema.tables;# Submit

ID: a' UNION select table_schema,table_name FROM information_Schema.tables;#
First name: information_schema
Surname: CHARACTER_SETS
••••••
ID: a' UNION select table_schema,table_name FROM information_Schema.tables;#
First name: dvwa
Surname: guestbook

ID: a' UNION select table_schema,table_name FROM information_Schema.tables;#
First name: dvwa
Surname: users
••••••
ID: a' UNION select table_schema,table_name FROM information_Schema.tables;#
First name: mysql
Surname: user
```

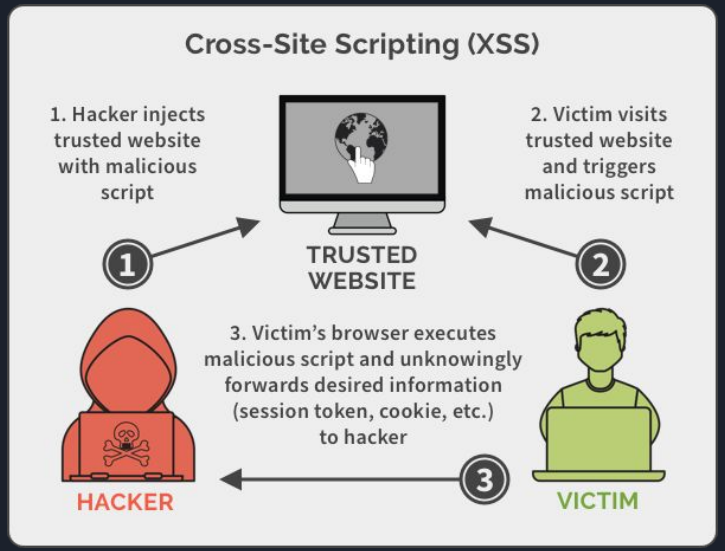
Prevention of SQL injections

- Input validation
  - Cleansing the inputs allowed within an entry box
  - Restrict certain characters from being allowed to be entered
- Parameterized queries
  - User input is passed as a parameter rather than as a literal
  - Aims to separate the query from the user input
  - Code can no longer be manipulated to become executable
- Regularly updating databases
  - New vulnerabilities and methodology are discovered frequently and it is the host's job to ensure they stay up to date on the latest discoveries to roll out patches

Cross-site scripting (XSS)

Malicious scripts injected into a secure service in order to taint the user's experience

- Consequences of XSS include:
  - Logging keystrokes
  - Malicious website redirects/phishing
  - Modify presented website data
  - Obtaining cookie information
  - Install malicious software



Cross-site scripting (XSS)

- The first cases XSS appeared in 1999 when employees at Microsoft noticed irregular data within various HTML pages
- The findings were not published until 2000 and was the first time the term “cross-site scripting” was used

Companies affected by XSS:

- Yahoo!	(2013 & 2017)	- Allowed two different types to affect their service
- Uber	(2019)	- Exposed the account details of thousands of drivers
- Twitter	(2009)	- Hovering mouse over link forced accounts to retweet the infected link

Cross-site scripting (XSS) Types

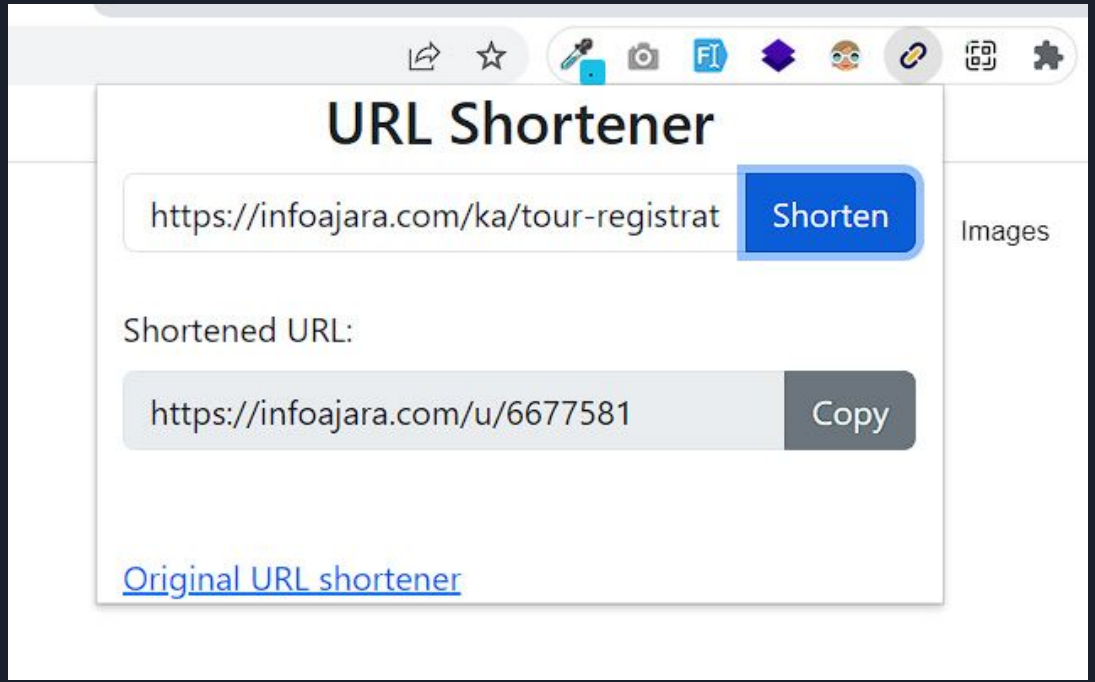
Non-persistent XSS

- Most common form of XSS
- Script embedded within a link that when opened executes the malicious code
- Often hidden within shortened URLs in order to trick users into believing the link is safe
- Danger is limited to those who click the link

Persistent XSS

- Less common form of XSS but much more dangerous
- Malicious script becomes stored on the host's page making anyone who accesses the page vulnerable
- Harder for users to detect as a supposedly trustworthy service implicitly hides malicious code

Cross-site scripting (XSS) Examples



Prevention of cross-site scripting (XSS)

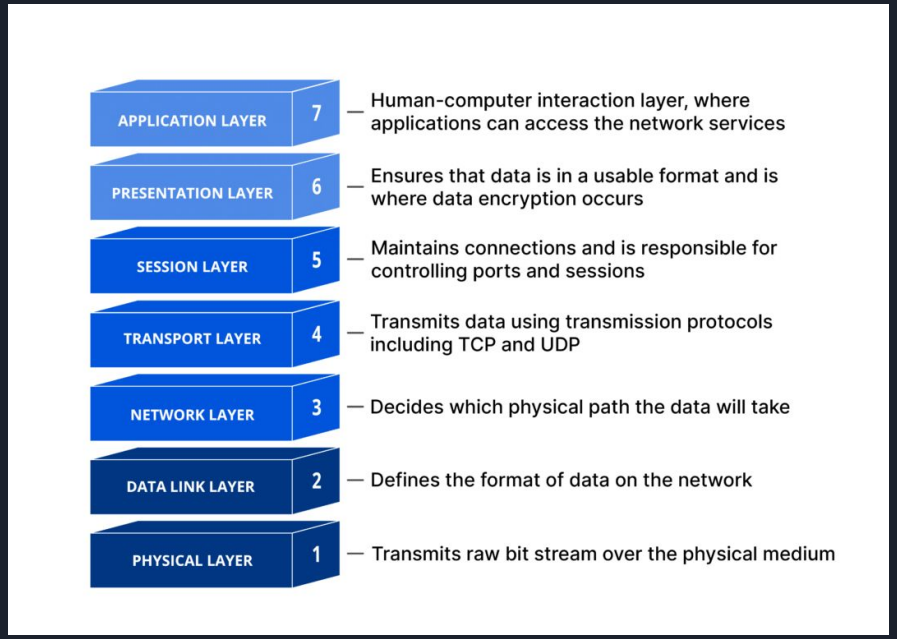
- Input validation
  - Cleansing the inputs allowed within an entry box
  - Restrict certain characters from being allowed to be entered
- Output encoding
  - Prevents special characters entered to being converted to executable code
  - Translate special characters that could be used to cause harm into safe equivalents
    - Ex: “<” into “&lt”
- Regular update up service
  - New vulnerabilities and methodology are discovered frequently and it is the host’s job to ensure they stay up to date on the latest discoveries to roll out patches

DDoS attacks

- Unmanageable amount of traffic to a systems servers, network, or services.
- Goal is to crash the target's system such that they're unable to continue normal functions targeting availability in our CIA triad.
- Timeline:
  - Hacker infiltrates unsuspecting persons to use as his botnet through use of malware installed on their computers.
  - Hacker selects their target.
  - Hacker uses his botnet of computers to all access target's IP address simultaneously
  - Target is unable to service large army of bots causing systems to crash.

Types of DDOS attacks

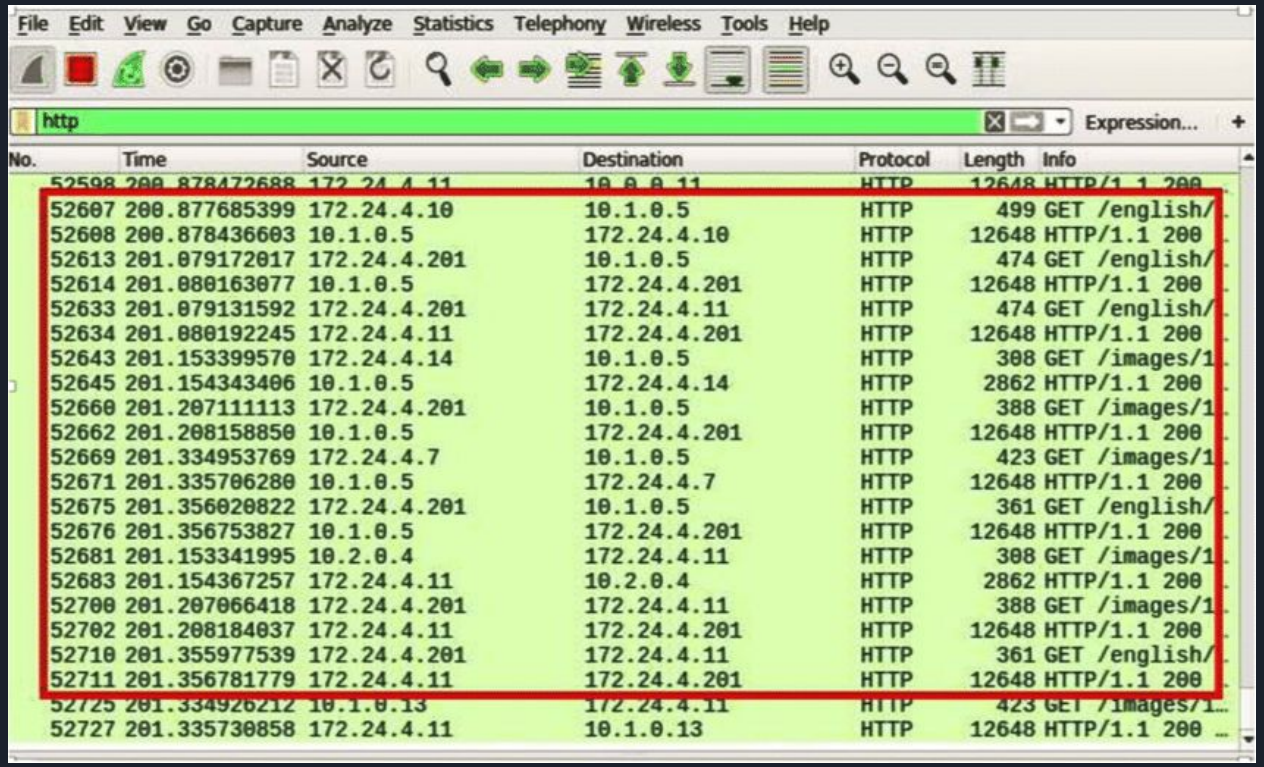
	Layer Targeted
Application Layers Attacks	7
Protocol Attacks	3 & 4
Volumetric attacks	3 & 4



Application Layer (Layer 7 attack)

- Goal: overload server's resources such that they crash to inability to keep up with requests.
- Target: HTTP aspect of a server.
- Examples of an attack:
  - Having a botnet admit multiple http requests simultaneously.
  - Having a botnet continuously refresh a webpage simultaneously.
- Cause server to crash due to insufficient computational power in comparison to number of RPS.

Example:


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
http
Expression...
No. Time Source Destination Protocol Length Info
52607 200.877685399 172.24.4.10 10.1.0.5 HTTP 499 GET /english/
52608 200.878436603 10.1.0.5 172.24.4.10 HTTP 12648 HTTP/1.1 200
52613 201.079172017 172.24.4.201 10.1.0.5 HTTP 474 GET /english/
52614 201.080163077 10.1.0.5 172.24.4.201 HTTP 12648 HTTP/1.1 200
52633 201.079131592 172.24.4.201 172.24.4.11 HTTP 474 GET /english/
52634 201.080192245 172.24.4.11 172.24.4.201 HTTP 12648 HTTP/1.1 200
52643 201.153399570 172.24.4.14 10.1.0.5 HTTP 308 GET /images/1
52645 201.154343406 10.1.0.5 172.24.4.14 HTTP 2862 HTTP/1.1 200
52660 201.207111113 172.24.4.201 10.1.0.5 HTTP 388 GET /images/1
52662 201.208158850 10.1.0.5 172.24.4.201 HTTP 12648 HTTP/1.1 200
52669 201.334953769 172.24.4.7 10.1.0.5 HTTP 423 GET /images/1
52671 201.335706280 10.1.0.5 172.24.4.7 HTTP 12648 HTTP/1.1 200
52675 201.356020822 172.24.4.201 10.1.0.5 HTTP 361 GET /english/
52676 201.356753827 10.1.0.5 172.24.4.201 HTTP 12648 HTTP/1.1 200
52681 201.153341995 10.2.0.4 172.24.4.11 HTTP 308 GET /images/1
52683 201.154367257 172.24.4.11 10.2.0.4 HTTP 2862 HTTP/1.1 200
52700 201.207066418 172.24.4.201 172.24.4.11 HTTP 388 GET /images/1
52702 201.208184037 172.24.4.11 172.24.4.201 HTTP 12648 HTTP/1.1 200
52710 201.355977539 172.24.4.201 172.24.4.11 HTTP 361 GET /english/
52711 201.356781779 172.24.4.11 172.24.4.201 HTTP 12648 HTTP/1.1 200
52727 201.335730858 172.24.4.11 10.1.0.13 HTTP 12648 HTTP/1.1 200

Protocol Attacks (Layers 3 and 4)

- Goal : Overload a server with too many packets
- Target: The TCP handshake
- Example of an attack:
  - Attacker sends TCP SYN packets with IP addresses that are spoofed
  - Server sends a response and waits for the ACK packet to be sent back from attacker
  - ACK packet never sent back causing the server to pile up on requests leading to server crashing
- The way protocol attacks overload is similar to application layer attacks but rely on a server's PPS.

Example:

Filter: c == 172.28.98.129) && (tcp.flags.ack == 1) | Expression... Clear Apply Save

No.	Time	Source	Destination	Protocol	Flags
1	2015-02-15 18:05:01.818908	172.28.98.129	172.31.252.197	TCP	0x0010
2	2015-02-15 18:05:01.819081	172.28.98.129	172.31.252.197	TCP	0x0010
3	2015-02-15 18:05:01.819222	172.28.98.129	172.31.252.197	TCP	0x0010
4	2015-02-15 18:05:01.819361	172.28.98.129	172.31.252.197	TCP	0x0010
5	2015-02-15 18:05:01.819498	172.28.98.129	172.31.252.197	TCP	0x0010
6	2015-02-15 18:05:01.819689	172.28.98.129	172.31.252.197	TCP	0x0010
13	2015-02-15 18:05:01.822284	172.28.98.129	172.31.252.197	TCP	0x0010
14	2015-02-15 18:05:01.822318	172.28.98.129	172.31.252.197	TCP	0x0010
15	2015-02-15 18:05:01.822448	172.28.98.129	172.31.252.197	TCP	0x0010
16	2015-02-15 18:05:01.822648	172.28.98.129	172.31.252.197	TCP	0x0010
17	2015-02-15 18:05:01.822789	172.28.98.129	172.31.252.197	TCP	0x0010
18	2015-02-15 18:05:01.822925	172.28.98.129	172.31.252.197	TCP	0x0010
19	2015-02-15 18:05:01.823060	172.28.98.129	172.31.252.197	TCP	0x0010
20	2015-02-15 18:05:01.823196	172.28.98.129	172.31.252.197	TCP	0x0010
21	2015-02-15 18:05:01.823331	172.28.98.129	172.31.252.197	TCP	0x0010
22	2015-02-15 18:05:01.823466	172.28.98.129	172.31.252.197	TCP	0x0010
23	2015-02-15 18:05:01.823600	172.28.98.129	172.31.252.197	TCP	0x0010
24	2015-02-15 18:05:01.823753	172.28.98.129	172.31.252.197	TCP	0x0010
25	2015-02-15 18:05:01.823892	172.28.98.129	172.31.252.197	TCP	0x0010
26	2015-02-15 18:05:01.824027	172.28.98.129	172.31.252.197	TCP	0x0010
27	2015-02-15 18:05:01.824161	172.28.98.129	172.31.252.197	TCP	0x0010

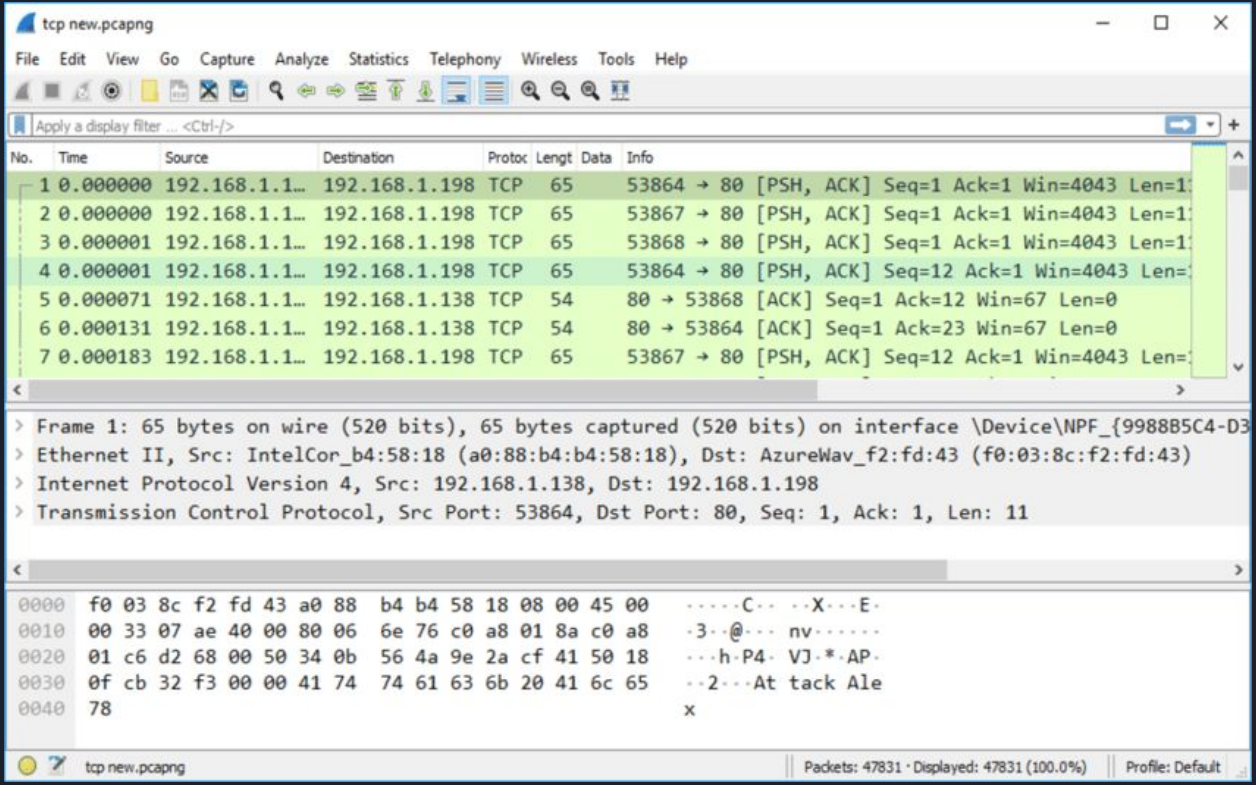
```
▽ Flags: 0x010 (ACK)
    000. .... .... = Reserved: Not set
    ...0 .... .... = Nonce: Not set
    .... 0... .... = Congestion Window Reduced (CWR): Not set
    .... .0.. .... = ECN-Echo: Not set
    .... ..0. .... = Urgent: Not set
    .... ...1 .... = Acknowledgment: Set
    .... .... 0... = Push: Not set
    .... .... .0.. = Reset: Not set
    .... .... ..0. = Syn: Not set
    .... .... ...0 = Fin: Not set
  Window size value: 512
  [Calculated window size: 512]
  [Window size scaling factor: -1 (unknown)]
```

```
0000  00 00 5e 00 01 67 04 01  3f 77 da 01 08 00 45 00   ..^..g.. ?w....E.
0010  00 28 0a 73 00 00 40 06  b8 da ac 1c 62 81 ac 1f   .(.s..@. ....b...
0020  fc c5 07 b0 00 50 41 59  37 72 4c fe 57 c6 50 10   .....PAY 7rL.W.P.
0030  02 00 d0 c1 00 00                                  ......
```

Volumetric Attacks (Layer 3 and 4)

- Goal: Overwhelm the server with traffic and requests
- Target: A server's bandwidth
- Example of an attack:
  - Attacker gains access to target's spoofed ip address
  - Attacker sends small DNS requests to a DNS server using spoofed ip.
  - DNS server responds to the target with larger DNS response than attacker sent.
  - The victim's servers become unable to keep up with traffic received causing them to shut down.
- Unlike the previous two attacks volumetric attacks are measured in BPS or GBPS

Example:


tcp new.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
No. Time Source Destination Protoc Lengt Data Info
1 0.000000 192.168.1.1… 192.168.1.198 TCP 65 53864 → 80 [PSH, ACK] Seq=1 Ack=1 Win=4043 Len=1
2 0.000000 192.168.1.1… 192.168.1.198 TCP 65 53867 → 80 [PSH, ACK] Seq=1 Ack=1 Win=4043 Len=1
3 0.000001 192.168.1.1… 192.168.1.198 TCP 65 53868 → 80 [PSH, ACK] Seq=1 Ack=1 Win=4043 Len=1
4 0.000001 192.168.1.1… 192.168.1.198 TCP 65 53864 → 80 [PSH, ACK] Seq=12 Ack=1 Win=4043 Len=
5 0.000071 192.168.1.1… 192.168.1.138 TCP 54 80 → 53868 [ACK] Seq=1 Ack=12 Win=67 Len=0
6 0.000131 192.168.1.1… 192.168.1.138 TCP 54 80 → 53864 [ACK] Seq=1 Ack=23 Win=67 Len=0
7 0.000183 192.168.1.1… 192.168.1.198 TCP 65 53867 → 80 [PSH, ACK] Seq=12 Ack=1 Win=4043 Len=
Frame 1: 65 bytes on wire (520 bits), 65 bytes captured (520 bits) on interface \Device\NPF_{9988B5C4-D3
Ethernet II, Src: IntelCor_b4:58:18 (a0:88:b4:b4:58:18), Dst: AzureWav_f2:fd:43 (f0:03:8c:f2:fd:43)
Internet Protocol Version 4, Src: 192.168.1.138, Dst: 192.168.1.198
Transmission Control Protocol, Src Port: 53864, Dst Port: 80, Seq: 1, Ack: 1, Len: 11
0000 f0 03 8c f2 fd 43 a0 88 b4 b4 58 18 08 00 45 00 ·····C·· ··X···E·
0010 00 33 07 ae 40 00 80 06 6e 76 c0 a8 01 8a c0 a8 ·3··@··· nv·····
0020 01 c6 d2 68 00 50 34 0b 56 4a 9e 2a cf 41 50 18 ···h·P4· VJ·*·AP·
0030 0f cb 32 f3 00 00 41 74 74 61 63 6b 20 41 6c 65 ··2···At tack Ale
0040 78 x
tcp new.pcapng
Packets: 47831 · Displayed: 47831 (100.0%)
Profile: Default

Identifying a DDoS attack

- Large rise in traffic associated with a single location or IP address
- Website loading speed significantly toggled
- 503: Service Unavailable
- Rise in memory usage or CPU usage
- Consistent traffic source address requesting same data even after the request timed out

Real World Examples

- The Google Attack, 2020
- The AWS Attack, 2020
- The GitHub Attack, 2018

The Google Attack, 2020

- Largest bandwidth attack known
- Traced back to three Chinese internet service providers.
- Lasted six months
- Used 180,000 infected servers achieving an attack measuring at one point to be 2.5Tbps

The AWS Attack, 2020

- Target was an AWS customer of which Amazon never stated who.
- Lasted for three days peaking at 2.3 Tbps
- Whilst they were unable to achieve their goal of shutting down AWS this showed how using AWS hosting may have some vulnerabilities damaging their brand and revenue.

The GitHub Attack, 2018

- Lasted only twenty minutes peaking at 1.35 Tbps
- Traced back to thousands of different autonomous systems spread throughout more than ten thousand unique endpoints.
- Important due to the impressive use of a technique known as a Memcached DDos attack
    - What this did was it amplified the attacker's request size by 51,200 times the original size.


DEMO
LOADING...