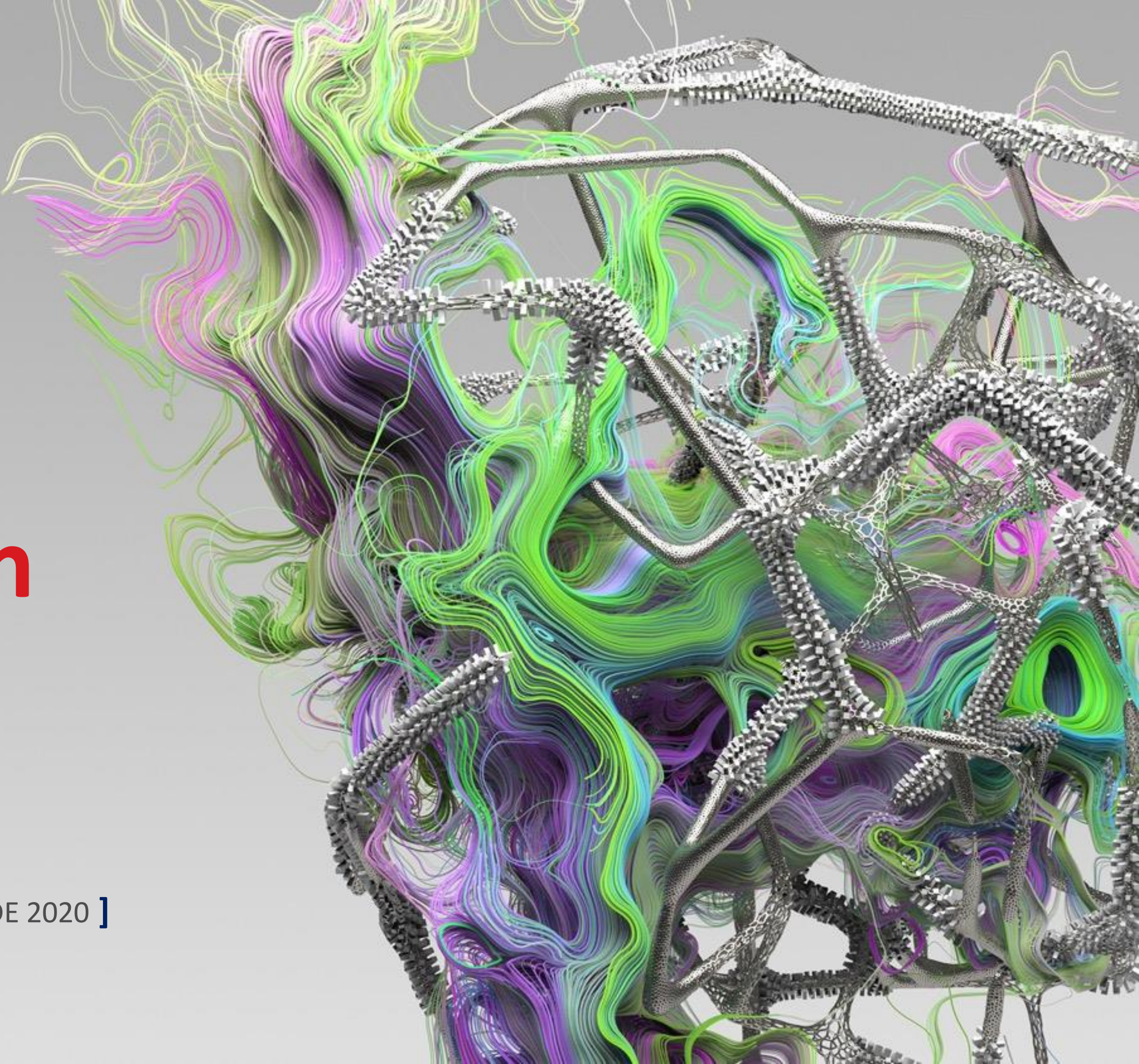
TREND MICRO

A Journey into Malware HTTP Communication Channels Spectacles

[Mohamad Mokbel | @MFMokbel | vOPCDE 2020]

July 15, 2020 - Toronto, Canada



Biography

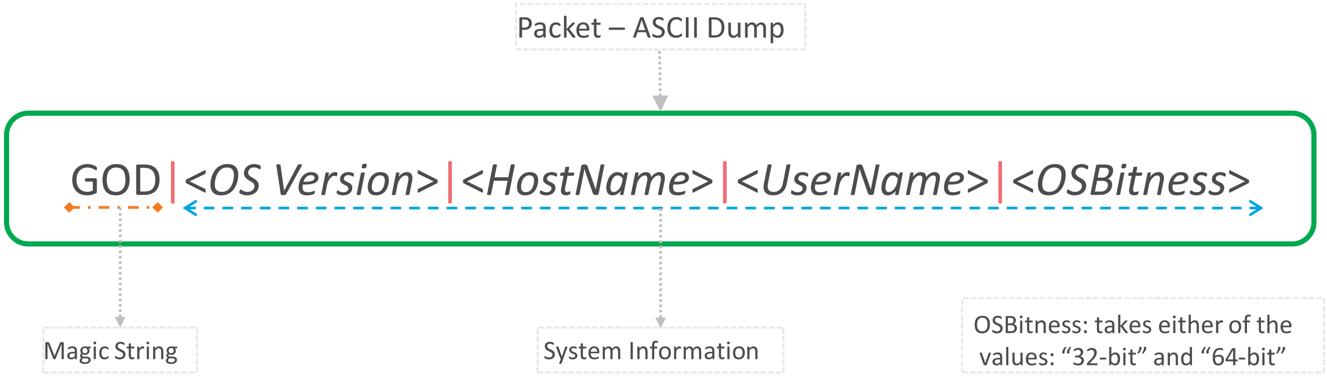
- Senior Security Researcher at Trend Micro
  - Member of the Digital Vaccine (DV) Lab
- Interests:
  - RE, Malware Research,
  - IDS/IPS,
  - C++, Compiler & Software Performance Analysis,
  - Exotic Communication Protocols

intrəˈdəkSHən

- Malware C&C communication protocols
- TCP/UDP Transport Layers
- Standard Proto
  - IRC, SMTP, FTP, HTTP(S), SMB, DNS, ICMP, SSL/TLS

intrəˈdəkSHən

- Custom Proto
  - Text-based. For example, data separated with a pipe character or any other delimiter:



Packet – ASCII Dump

GOD|*<OS Version>*|*<HostName>*|*<UserName>*|*<OSBitness>*

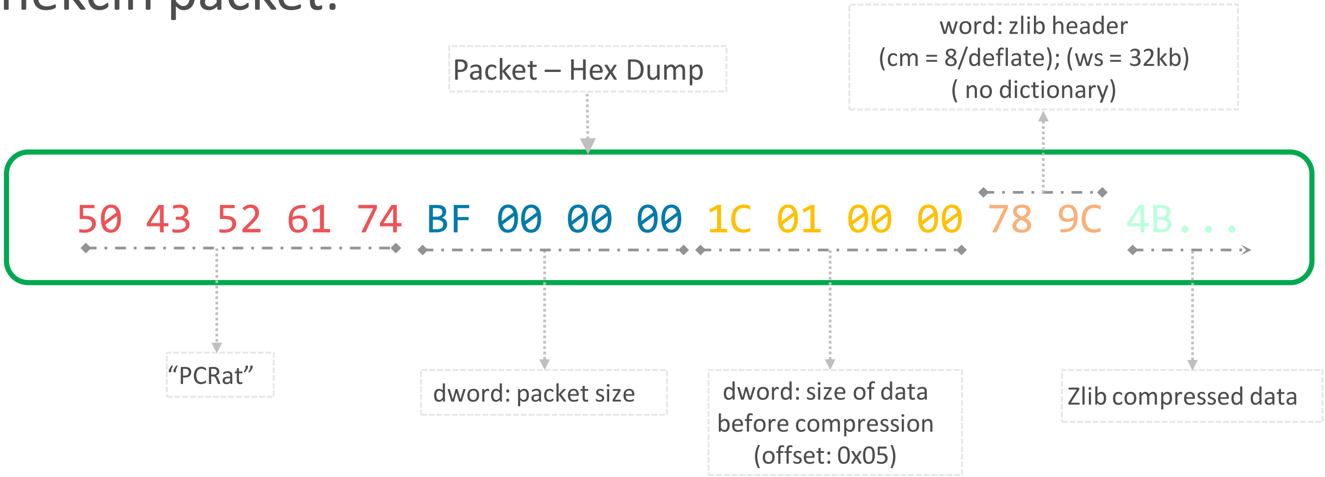
Magic String

System Information

OSBitness: takes either of the values: “32-bit” and “64-bit”

intrəˈdəkSHən

- Custom Proto
  - Binary-based. This is best illustrated with a variant of the PCRat chekcin packet:



Packet – Hex Dump

```
50 43 52 61 74 BF 00 00 00 1C 01 00 00 78 9C 4B...
```

word: zlib header
(cm = 8/deflate); (ws = 32kb)
(no dictionary)

“PCRat”

dword: packet size

dword: size of data
before compression
(offset: 0x05)

Zlib compressed data

intrəˈdəkSHən

- IRC vs HTTP
  - Client-server networking model
- A Protocol is never defined by its port number
  - It is the semantics and architectural design and implementation that defines it
- WinHTTP or WinINet: HTTP RFC standard compliant!
- Windows Sockets Winsock : not HTTP
- Malware has full control over the client and server setup and communications

intrəˈdəkSHən

- HTTP becoming the defacto protocol for C&C communications
  - Then, peculiarities in the construction of the request and response headers started to emerge
  - Examples:
    - Specific headers in a GET request that only make sense in a POST request
    - Using wrong Content-Length value that doesn’t match the actual payload’s size
- Leverage those mistakes to our advantage for writing IDS/IPS heuristic based filters
- Will go through a journey of documenting semantic- and syntax-level type errors, not only typographical

The mysterious case of PKEY (also known as Adelinoq)

- Wrong Content-Length value that doesn’t match the actual payload’s size
- Payload size is 271 bytes
- The malware uses the Winsock library ws2_32.dll

```
POST /link.php HTTP/1.1
Host: abcdefghij[.]com
Accept: text/html, */*
Content-Type: application/x-www-form-urlencoded
Content-Length: 4
Connection: close

// payload is shown in hexadecimal
000000A0  50 4b 45 59 00 00 00 00  00 00 00 00 00 00 00 00 PKEY....
```

Continue...

```
length_of("uri_file_name") +
length_of("post_req_format_pattern") +
length_of("payload_str") + length_of("C&C_server_addr") +
0x104 = packet size
```

For example,

```
length_of("/link.php") + 0x90(144) + length_of("PKEY") +
length_of("abcdefghij[.]com") + 0x104(260)
= 0x09 + 0x90(144) + 0x04 + 0x0E(14) + 0x104(260)
= 0x1AF(431) -> packet size
```

The mysterious case of ‘reverse gear’

- Direction of HTTP Request and Response is inverted

```
HTTP/1.1 200 OK
Server: sffe
Content-Length:53
Connection:Keep-Alive

// payload is shown in hexadecimal
0000004B  9f 98 85 84 92 88 96 84  84 92 85 83 f6 00 00 00
0000005B  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0000006B  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0000007B  00 00 00 00 00
```

Continue

```
GET /index.html HTTP/1.1
Accept:*/*
Host:127.0.0.1
Content-Length:30
Connection:Keep-Alive

// payload is shown in hexadecimal (A payload associated with a GET request!)
00000062  96 84 84 92 85 83 88 98  9c f6 00 00 00 00 00 00
00000072  00 00 00 00 00 00 00 00  00 00 00 00 00 00
```

The mysterious case of ProtonBot GET request

- Content-Type header in a GET request

```
GET /page.php?id=1A28798B-9001-51CF-710A-89AF207D10F2&clip=get HTTP/1.1
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Proton Browser
Host: example.com
Cache-Control: no-cache
```

The mysterious case of Crosswalk

- Content-Length header value is too large
- For example, the malware sent:

```
GET http://10.10.10.1/QUERY/en-us/msdn/ HTTP/1.1
dCy: RjFDRDJGSkcta2N0YTJOMFlUSk9NRmxVUw==1
Connection: Keep-Alive
Host: 10.10.10.1
Content-Length: 0
```

- And the server responded with:

```
HTTP/1.1 200 OK
Content-Length: 524288000
Connection: keep-alive

// payload is omitted for readability purpose
// payload is of binary type and actual size is 256 bytes
```

The mysterious case of Socnet

- Illegal characters in header name

```
GET /ab/setup.php?act=tw_get HTTP/1.1
Host: example.com
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US) AppleWebKit/532.5
(KHTML, like Gecko) Chrome/4.1.249.1064 Safari/532.5
Accept:
application/xml,application/xhtml+xml,text/html;q=0.9,text/plain;q=0.8,image/png
,*/*;q=0.5
(null)Connection: close
```

The mysterious case of Cobra and Aytoke

- Whitespace characters at the end and beginning of the headers

```
GET http://example.com/index01.html HTTP/1.1SP
Accept-Language: en-usSP
Accept: */*SP
User-Agent: MozillaSP
Proxy-Connection: Keep-AliveSP
Host: example.comSP
Cache-Control: no-store, no-cacheSP
Pragma: no-cacheSP
Cookie: <exfiltrated data (encoded)>SP
```

```
GET / HTTP/1.1
Host: checkup[.]dyndns[.]org
SPConnection: close

|00|
```

The mysterious case of Kloka

- Whitespace characters at the end of the header name

```
POST /test/add.php HTTP/1.1
HostSP: example.com
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64; rv:35.0) Gecko/20100101 Firefox/35.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept(-Encoding)SP: gzip, deflate
Content-Type: multipart/form-data; boundary=---------------------------84732825325733
Content-Length: 1234
```

Accepting or Rejecting a Request

Accepting or rejecting a request usually follows the robustness principle as clarified in the **RFC2145**, original RFC791 (section 3.2),

“In general, an implementation must be conservative in its sending behavior, and liberal in its receiving behavior. That is, it must be careful to send well-formed datagrams, but must accept any datagram that it can interpret (e.g., not object to technical errors where the meaning is still clear).”.

Of course, this is implementation and interpretation dependent, whereby different HTTP servers need not behave the same way.

Accept header shenanigans

- The many cases of malware Accept header blunders
- The Accept request header is meant to signal to the server what content type(s) the client is able to understand.
- Syntax:

```
<mime_type>/<sub_type>, <mime_type>/* or */*
```

- Check for the absence of the “/” character in the Accept header-field value, or check it against a whitelist.

2769 Pkt Cptrs		
18 malware families: Ø xml	*.* domain name	nonsense value

Accept Header (case in point)

```
POST /login.php HTTP/1.1
Accept: text/html,application/xhtml+xm.plication/xml;q=0.9,*/*;q=0.8
User-Agent: Mozilla/5.0 (Windows NT 6.1; rv:60.0) Gecko/20100101 Firefox/60.0
Host: 10.10.10.1:8080
Content-Length: 32
Cache-Control: no-cache

344431B6136D10C2BEC65491025A1EB7
```

The correct value

xml,application

Cookie header oddities and other peculiarities

- Used for tracking, session management or customization
- Requested by the server via the Set-Cookie header to store whatever Cookie on the client side

```
Cookie: name=value; namex=valuex; namey=valuey
```

- Check for the absence of the separator ‘=’ in the Cookie name-value pair(s)

2769 Pkt Cptrs	
13 malware families: 9 Ø 1 hs	3 data exfiltration

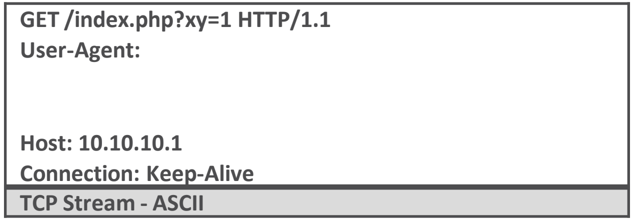
- Checking for the pair separator ‘;’ such that it is not followed by a space, reveals the malware family Noobot (not in the results of the first check), exfiltrating data via the Cookie header

Cookie… Sefnit

```
GET favicon.ico HTTP/1.1
Host: example.com
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg
Accept-Language: en-us
UA-CPU: x86
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; .NET CLR 3.0.0.1)
Cookie: 5b1ac9eff0694a3dbd7add2cd8ecad580cec7db8.
Content-Length: 10038
Max-Forwards: 107719
|00|
```

Silencing Silence (the mysterious case of Silence/TrueBot)

- Multiple new lines between headers

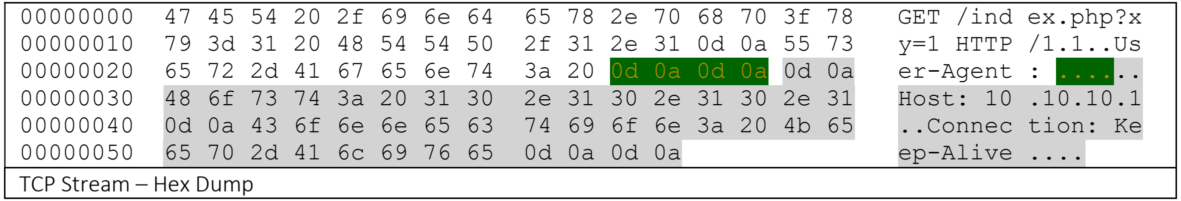


```
GET /index.php?xy=1 HTTP/1.1
User-Agent:



Host: 10.10.10.1
Connection: Keep-Alive
```

TCP Stream - ASCII



```
00000000  47 45 54 20 2f 69 6e 64  65 78 2e 70 68 70 3f 78   GET /ind ex.php?x
00000010  79 3d 31 20 48 54 54 50  2f 31 2e 31 0d 0a 55 73   y=1 HTTP /1.1..Us
00000020  65 72 2d 41 67 65 6e 74  3a 20 0d 0a 0d 0a 0d 0a   er-Agent : ......
00000030  48 6f 73 74 3a 20 31 30  2e 31 30 2e 31 30 2e 31   Host: 10 .10.10.1
00000040  0d 0a 43 6f 6e 6e 65 63  74 69 6f 6e 3a 20 4b 65   ..Connec tion: Ke
00000050  65 70 2d 41 6c 69 76 65  0d 0a 0d 0a               ep-Alive ....
```

TCP Stream – Hex Dump

Silencing Silence… continue

- Multiple new lines between headers

```
004084D4      push    0                      ; dwFlags
004084D6      push    0                      ; pszProxyBypassW
004084D8      push    0                      ; pszProxyW
004084DA      push    0                      ; dwAccessType
004084DC      push    offset pszAgentW       ; "\r\n\r\n"
004084E1      call    eax ; WinHttpOpen
```

Other Cases and Discussions

- 2769 pcaps | http.user_agent contains “+” -> 15 malware families
  - Exception: Bestafera family uses the Googolebot (Desktop) user-agent value:
    - “Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html)”
- 2769 pcaps | http.user_agent contains “&” -> 1 (Alureon malware family)
  - Ex., User-Agent: id=5247819786&smtp=ok&ver=102
  - DanBot (secureworks)
- 2769 pcaps | http.user_agent contains “=” -> 8 malware families
  - Data exfiltration
  - Blatant mistake: (User-Agent: User-Agent=Mozilla/5.0 (Windows; U; Windows NT 5.1; zh-CN; rv:1.9.0.1) Gecko/2008070208 Firefox/3.0.1)
- 2769 pcaps | http.user_agent contains “\” -> 7 malware families

Other Cases and Discussions… continue…

- Scan HTTP headers for special or not commonly used characters
  - Will undoubtedly reveals strange traffic
  - Scanning a given header field-value for other than predefined values, might also uncover suspicious traffic

- 2769 pcaps | http.content_disposition !(contains "inline" or "attachment" or "form-data" or "name" or "filename“) -> 1 (Nalodew malware families)

```
POST /cm.php?do=3&cf=1|681957093953-1712835101-2-3116a03e| HTTP/1.1
Accept: */*
Content-Disposition: ###01504290527-0-2(v30000)
User-Agent: 30000!0*:---:<computer_name>-<user_name>:---:LAN Connection:---:www:---:NEW
Host: example.com
Content-Length: 3555
Connection: Keep-Alive
Cache-Control: no-cache

// payload is omitted for readability purpose
```

The one mistake we’re all looking for…

- Ultralocker malware family

```
POST / HTTP/1.1
User-Agent: agent
Referrer: http://www.yahoo.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 51
Accept: */*
Host: 10.10.10.1
Connection: Keep-Alive

pcname=<computer_name>&hwid=<id_value>&version=Locker
```

- Only 8 out of 2769 malware families have the Referer header field in their requests

Check for case sensitivity (48)

- Per specification, the header-field name is case insensitive,
  - however, legitimate traffic tend to follow proper casing on the headers, and
  - malware tend to deviate from such norm

Header field-name (lowercase)	#	
pragma	6	
cookie	1	
user-agent	2	one of the families used the library libsfml to generate network traffic, which has the header, all lower case, with the hardcoded value "libsfml-network/2.x".
connection	3	
cache-control	9	
host	2	
content-length	6	
content-type	6	
accept-encoding	3	
accept	0	
content-disposition	10	one family was found sending control commands via the content-disposition request and response headers.
proxy-connection	0	

Check for case sensitivity

- Case sensitivity of the HTTP request’s method: get (2), post (0)
- Bluesummer malware family (http)
- Other anomalies: “hs-uk”
  - zero-spacing between the separator ‘:’ and the header value

```
GET /index.html http/1.1
Accept:*/*
Accept-Language: hs-uk
Accept-Encoding:gzip ,deflate
Proxy-Connection: Keep-Alive
Pragma:no-cache

// this GET request has a payload
// payload is omitted for readability purpose
```

Let’s revisit the case of empty header(s)

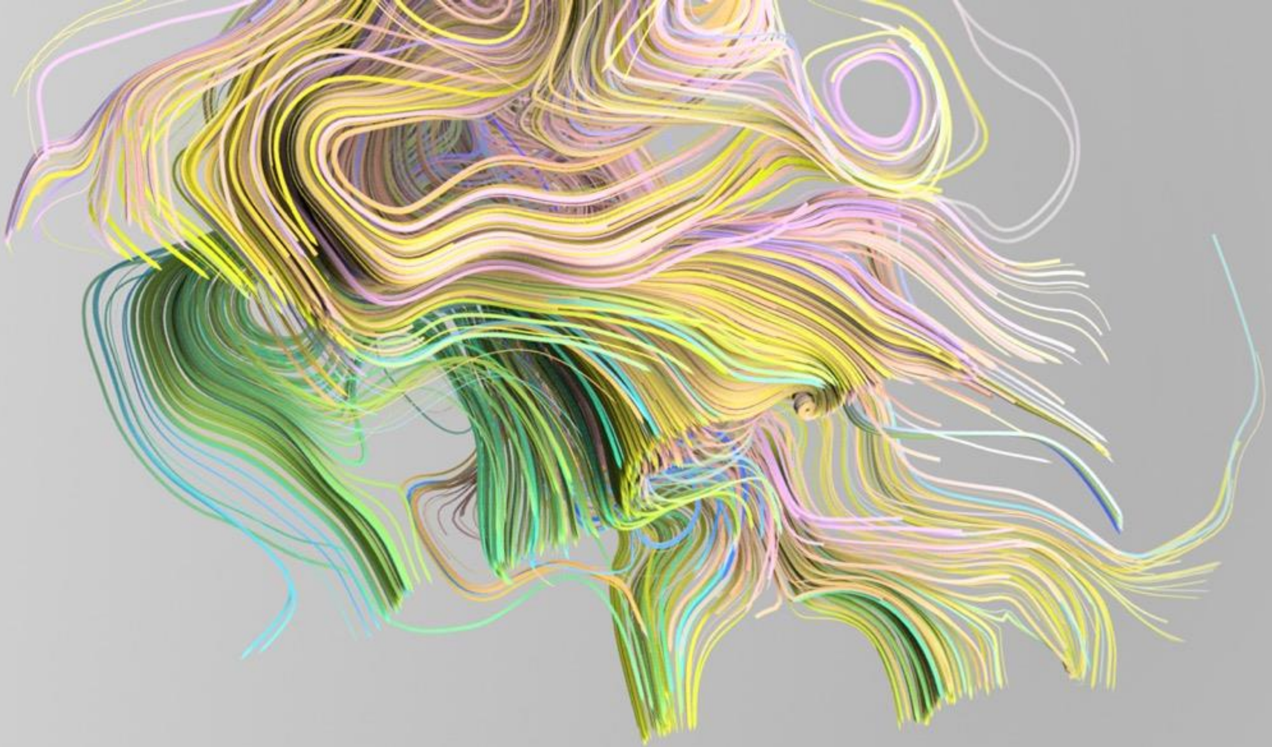
- Meet Beerish
- Used in a limited targeted attacks
- Delivered via the exploitation of CVE-2019-1367

```
GET /ixx/u3/scrobi.32 HTTP/1.1
Accept: */*
Accept-Language: en
Accept-Encoding:
User-Agent: IE7
Host:www.example.com
```

Other Considerations

- Both Apache and Nginx HTTP servers accept HTTP version numbers other than 1.0, 1.1 and 2.0?!
- You have to be port agnostic when hunting for those anomalies on the network
- Implementation is still a subjective interpretation
- Case in point, HTTP Request Smuggling attacks illustrate what could go wrong when the RFC specification is not honored

HTTP Desync Attacks: Request Smuggling Reborn



kənˈklo͞oZHən

Thank You

Q & A