Karlsruhe Institute of Technology

Resilient Networking
Module 5: Denial of Service

Thorsten Strufe – *This module prepared in cooperation with Günter Schäfer, Mathias Fischer, and the members of the Chair.*

Winter Term 2020 – KIT/TUD

Competence Center for Applied Security Technology

Denial of Service

- Classification
- DoS examples
  - Exploiting IP fragmentation and assembly
  - Abusing ICMP: Smurf attack
  - TCP SYN-Flood attack
  - DDoS
  - Botnets
  - DRDoS
- Countermeasures against DoS
  - Crypto Puzzles
  - Stateless Protocols
  - Avoid IP address spoofing / identifying malicious nodes
  - Filtering attack traffic
  - Industry solutions to DDoS mitigation

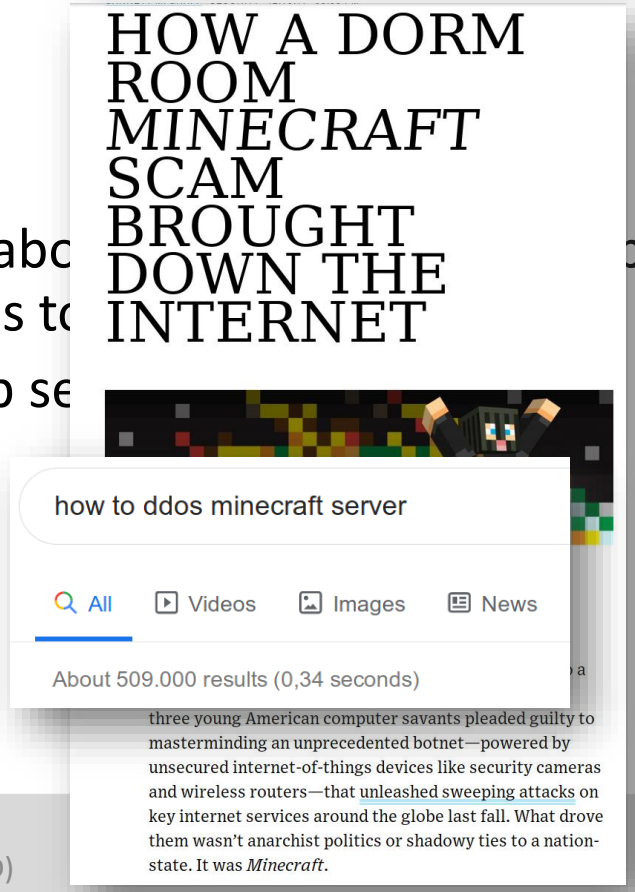
The Threat...

(source: Julie Sigwart - "Geeks")

Introduction

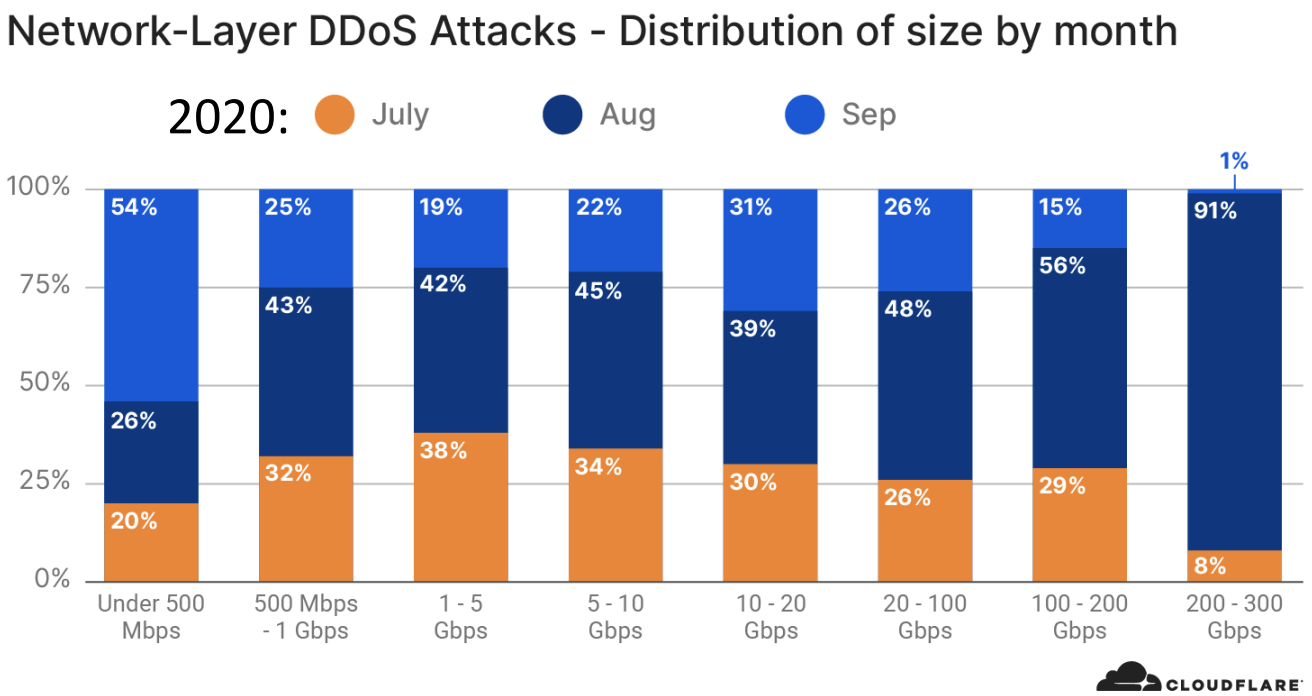
- What is Denial of Service?
  - Denial of Service (DoS) attacks aim at ***denying*** or ***degrading*** legitimate users' ***access to a service*** or network resource, or at bringing down the servers offering such services

- Motivations for launching DoS attacks:
  - Hacking (just for fun, by "script kiddies", ...)
  - Gaining information leap ($\rightarrow$ 1997 attack on bureau of labo ossibly launched as unemployment information has implications to
  - Discrediting an organization operating a system (i.e. web se
  - Revenge (personal, against a company, ...)
  - Political reasons ("information warfare")
  - Financial advantage (mirai and minecraft, 2016)
  - ...



How serious is the DoS problem? (1)

- Qualitative answer:
  - Very, as our modern information society depends increasingly on availability of information and communications services
  - Even worse, as attacking tools are available for download



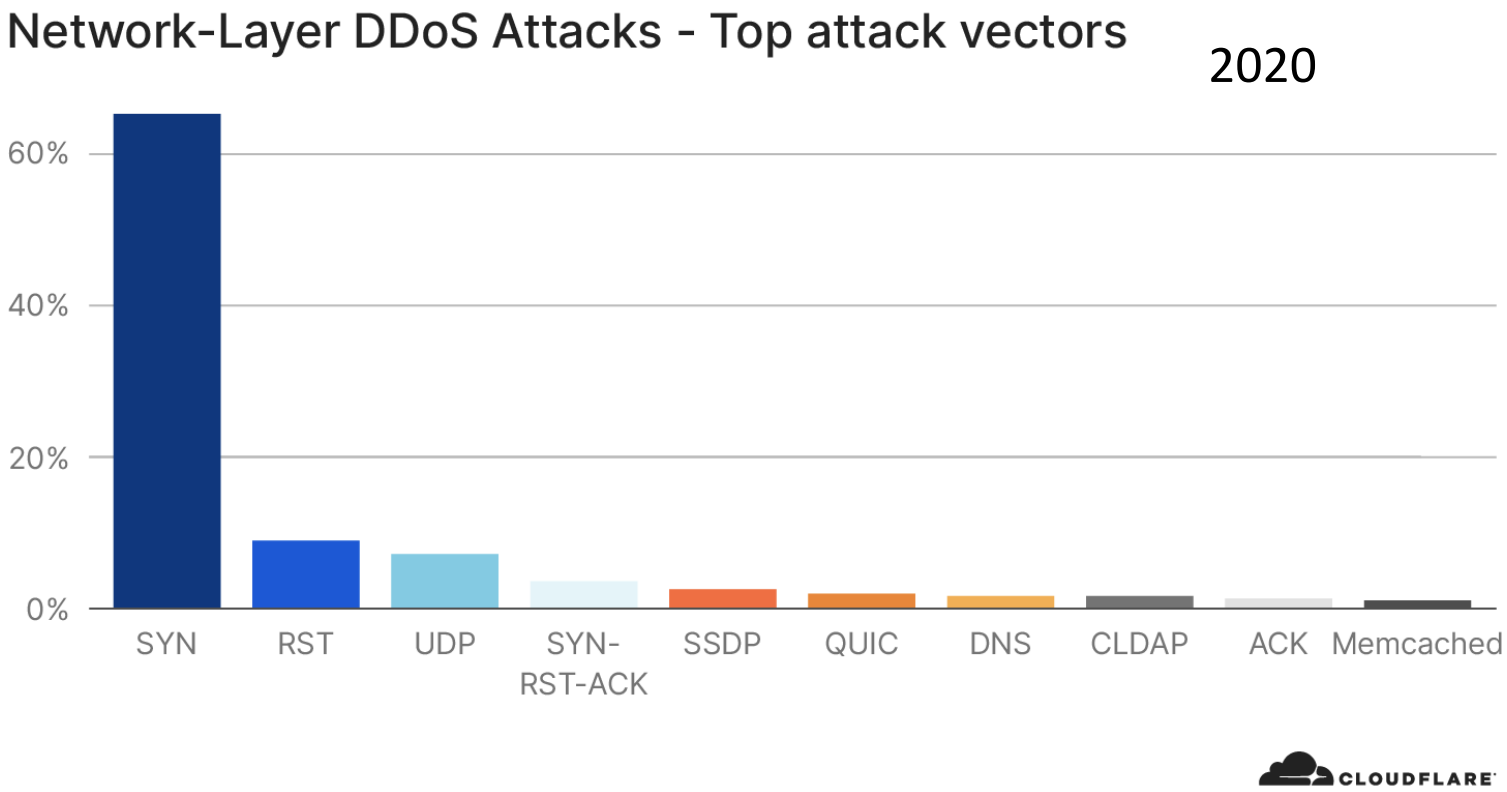
- Largest seen DoS attack so far: 2.3 Tbps (on Amazon AWS in 2020)

https://blog.cloudflare.com/network-layer-ddos-attack-trends-for-q3-2020/

How serious is the DoS problem? (2)

- Various attack vectors used

DDoS blackmailing is a lucrative business model!



https://blog.cloudflare.com/network-layer-ddos-attack-trends-for-q3-2020/

Denial of Service Attack Classes

Classification depending on different aspects:

- *Attack effect*
  - Resource destruction
  - Resource depletion

- *Origin of malicious traffic*
  - Single source with single / multiple (forged) source addresses
  - Multiple sources (Distributed DoS)

- *Attack target*
  - Victim
  - Infrastructure

Attack Effect in Denial of Service

- *Affected resource*
  - Network connectivity (uplink, transit link)
  - Computation
  - Memory

- *Resource **destruction***:
  - Hacking into systems
  - Making use of implementation weaknesses like buffer overflows
  - Deviation from proper protocol execution
  - Your common TU Dresden Excavator

- *Resource **depletion*** by causing:
  - Storage of (useless) state information
  - High traffic load (requires high overall bandwidth from attacker)
  - Expensive computations (“expensive cryptography”!)
  - Resource reservations that are never used (e.g. bandwidth)

Karlsruhe Institute of Technology

KASTEL

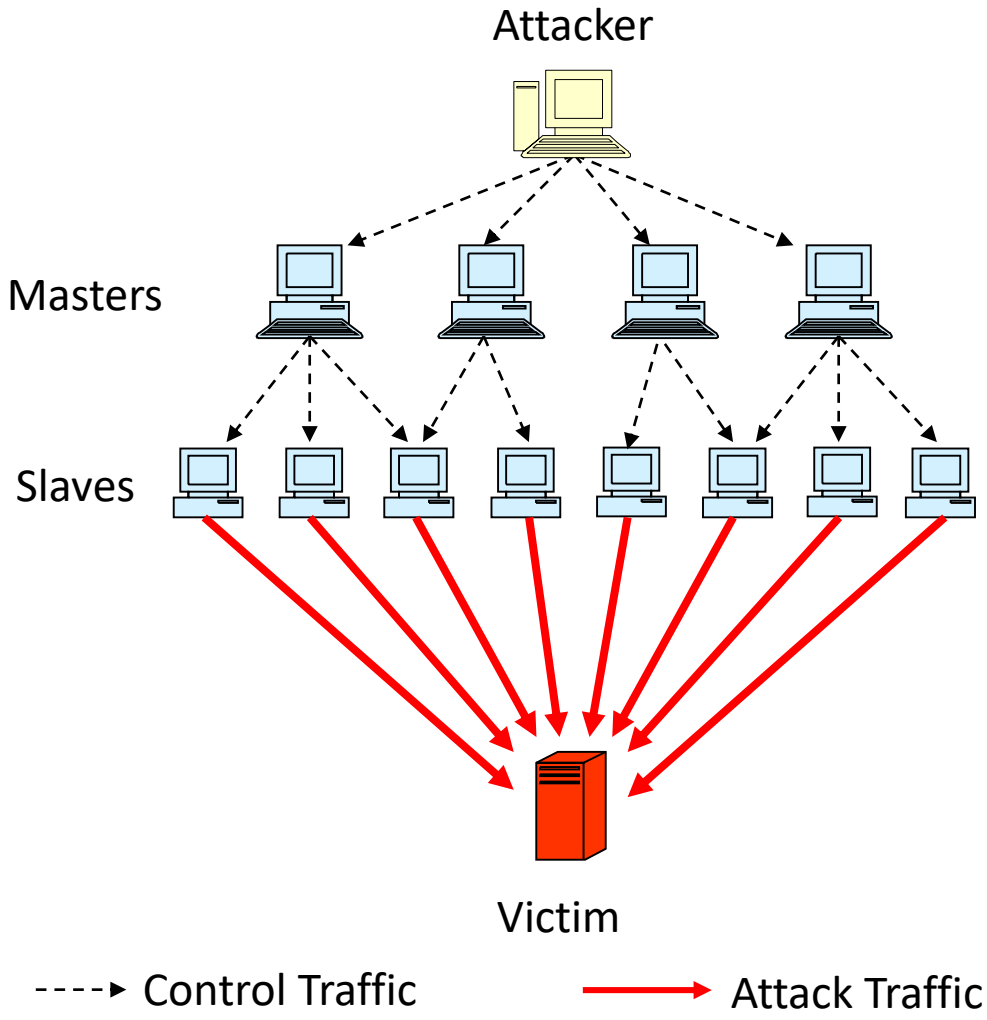
So how is it done?

Attacking Techniques

- ***Reflector*** attacks: Generate traffic indirection
- Request service in the name of the victim (e.g. spoofed IP – *which protocols*?)
- Hides attack source, allows for external amplification

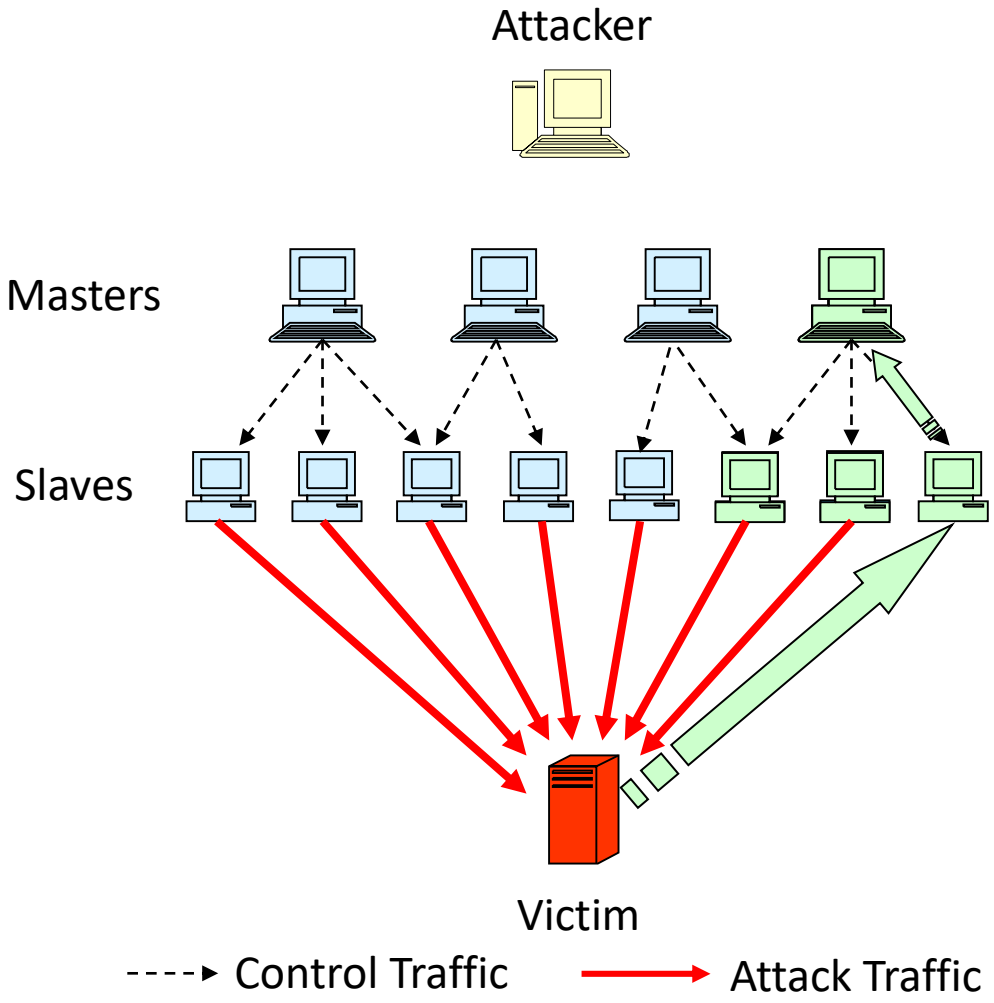
- ***Amplification*** attacks: Leverage asymmetry in protocols
- Send lightweight requests (low cost) that generate heavyweight responses or heavy load on the service (crypto)
- Increases damage

DoS Tools: Botnets 101



- The attacker classifies the compromised systems in:
  - Master systems
  - Slave systems
- Master systems:
  - Receive command data from attacker
  - Control the slaves
- Slave systems:
  - Launch the proper attack against the victim
  - During the attack there is no traffic from the attacker

Botnet Strategies: Partitioning



- Each master system only knows some slave systems
- Therefore, the network can handle partial failure, caused by detection of some slaves or masters

Karlsruhe Institute of Technology

KASTEL

Resource Destruction

Resource Destruction – Examples (1)

- *Resource Destruction:*
- Physically/Logically destroy a resource that is vital for targeted service

- *Hacking:*
  - Exploiting weaknesses that are caused by careless operation of a system
  - Examples: default accounts and passwords not disabled, badly chosen passwords, social engineering (incl. malware attachments), etc.

- *Making use of implementation weaknesses*
  - Buffer Overflows, Format-String-Attacks, ...

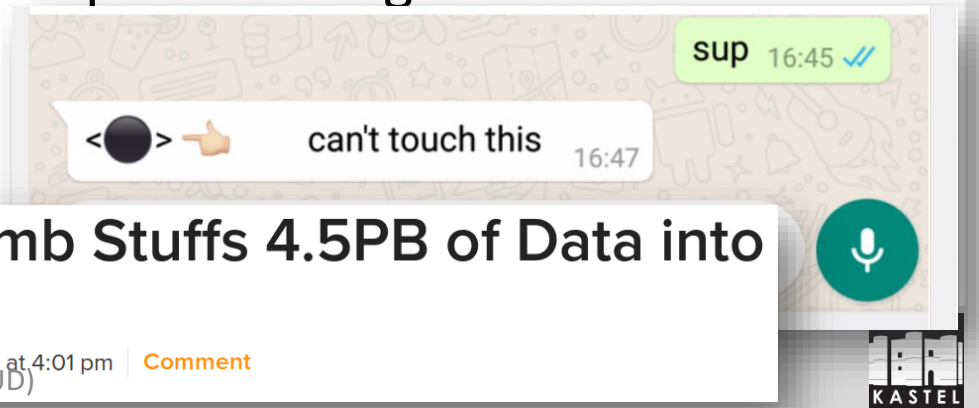
- *Deviation from proper protocol execution:*
  - Example: exploit IP's fragmentation & reassembly

Resource Destruction – Examples (2)

- Original Teardrop attack: exploit IP's fragmentation & reassembly
  - Send IP fragments to broadcast address 192.168.133.0
  - BSD-based OS used to respond to broadcast messages, messages can be fragmented
  - Response requires ***reassembly,*** first
  - If an attacker sends a ***lot of fragments without*** ever sending a ***first / last fragment***, the buffer of the reassembling system gets ***overloaded***
  - (Routers use BSD-based TCP/IP stacks -> attack on network infrastructure)

  - Sending a series of fragmented IP datagram pairs with overlapping offset to target
  - Windows 95: crashed when trying to reassemble one pair of datagrams

More recently: ⚐0 ;-)



Defending Against Resource Destruction DoS

Karlsruhe Institute of Technology

Defenses against disabling services:

- Hacking:
  - Good system administration
  - Firewalls, logging & intrusion detection systems
- Implementation weakness:
  - Code reviews, stress testing, etc. (in theory: verification and microkernels)
- ***Protocol deviation:***
  - Fault tolerant protocol design
  - Attack-aware protocol deployment (fail2ban, rate limiting, etc)
  - “DoS-aware protocol design”:
    - Be aware of possible DoS attacks when e.g. reassembling packets
    - Do not perform expensive operations, reserve memory, etc., before authentication

Karlsruhe Institute of Technology

KASTEL

Resource Depletion

Background: Internet Control Message Protocol

- Internet Control Message Protocol (ICMP) has been specified for communication of error conditions in the Internet
- ICMP PDUs are transported as IP packet payload and identified by value “1” in the protocol field of the IP header

- Two main reasons make ICMP particular interesting for attackers:
  - It may be addressed to broadcast addresses
  - Routers respond to it

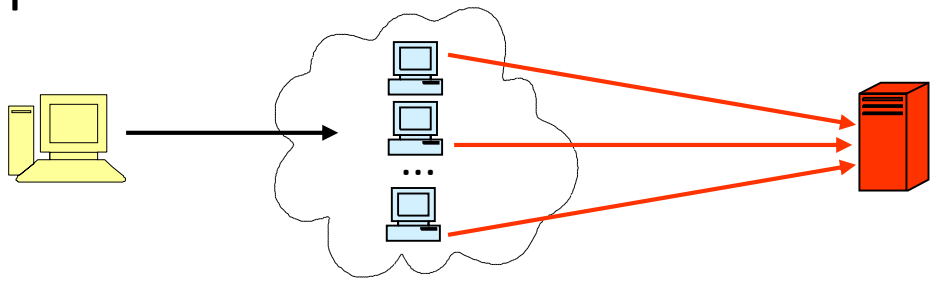
ICMP Functions

- ***Announce network errors:*** e.g. a host or entire portion of the network being unreachable, or a TCP or UDP packet directed at a port number with no receiver attached (destination unreachable)
- ***Announce network congestion:*** routers generate ICMP source quench messages, when they need to buffer too many packets
- ***Assist troubleshooting:*** ICMP supports an Echo function, which just sends an ICMP echo packet on a round trip between two hosts
- ***Announce timeouts:*** if an IP packet's TTL field drops to zero, the router discarding the packet may generate an ICMP packet (time exceeded)
- ***Announce routing detours:*** if a router detects that it is not on the route between source and destination, it may generate an ICMP redirect packet

The mother of DoS: Smurf – ICMP Bandwidth Depletion

- Two reasons make ICMP particular interesting for attackers:
  - It may be addressed to broadcast addresses
  - Routers respond to it

- The ***Smurf attack*** - ICMP echo request to broadcast:
  - Routers (sometimes) allow ICMP echo requests to broadcast addresses…
  - An attacker sends an ICMP echo request to a *broadcast address* with the *source address* forged to refer to the victim
  - All devices in the addressed network respond to the packet
  - The victim is flooded with replies to the echo request
  - With this technique, the network being abused as an (unaware) attack amplifier is also called a *reflector network*:



More recent examples…

Jonker, Mattijs, et al. "Millions of targets under attack: a macroscopic characterization of the DoS ecosystem." *Proceedings of the 2017 Internet Measurement Conference*. ACM, 2017.

Rossow, Christian. "Amplification Hell: Revisiting Network Protocols for DDoS Abuse." *NDSS*. 2014.

"Identifying the scan and attack infrastructures behind amplification DDoS attacks." *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2016.

Schuchard, Max, et al. "Losing control of the internet: using the data plane to attack the control plane." *Proceedings of the 17th ACM conference on Computer and communications security*. ACM, 2010.

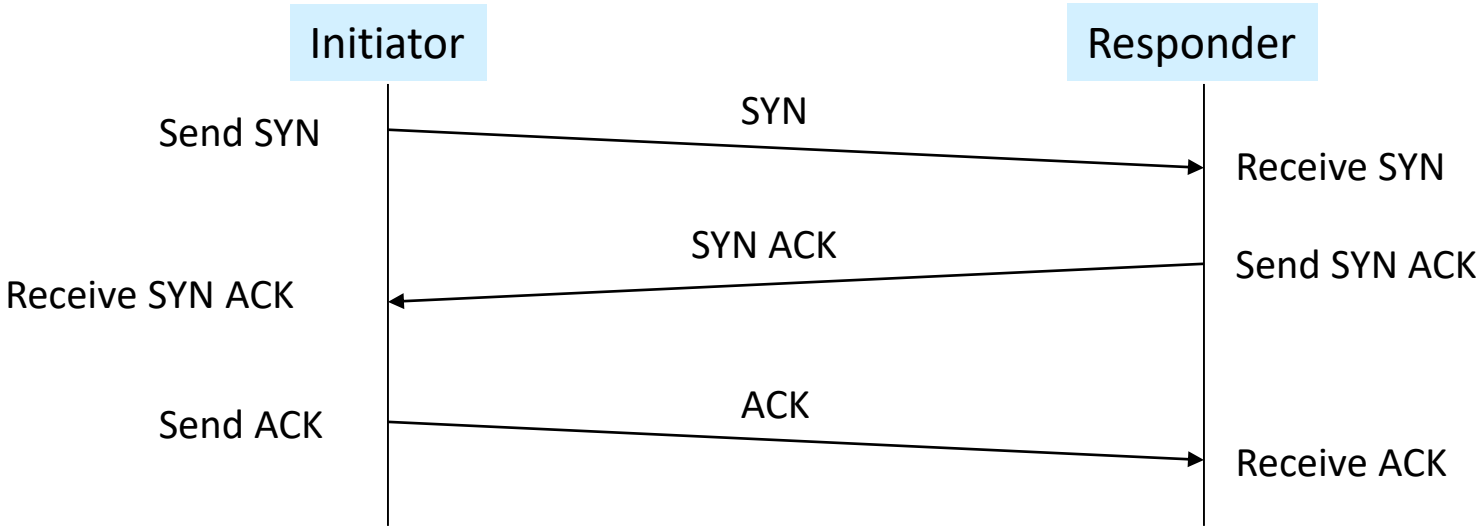
Smith, Jared M., and Max Schuchard. "Routing around congestion: Defeating DDoS attacks and adverse network conditions via reactive BGP routing." *2018 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2018.

Global Distributed Denial-Of-Service (DDoS) Protection Market 2019 – …nie Networks, ARBOR NETWORKS, Imperva …uard

The global "Distributed Denial-Of-Service (DDoS) Protection" market report …ted Denial-Of-Service (DDoS) … also assesses the Distributed Denial- … of topography, technology, and consumers. The report also covers the volume … of the market during the projected … ted Denial-Of-Service (DDoS) … sentation of the Distributed Denial- … he global and regional level. **The key** … OR NETWORKS, Imperva Incapsula

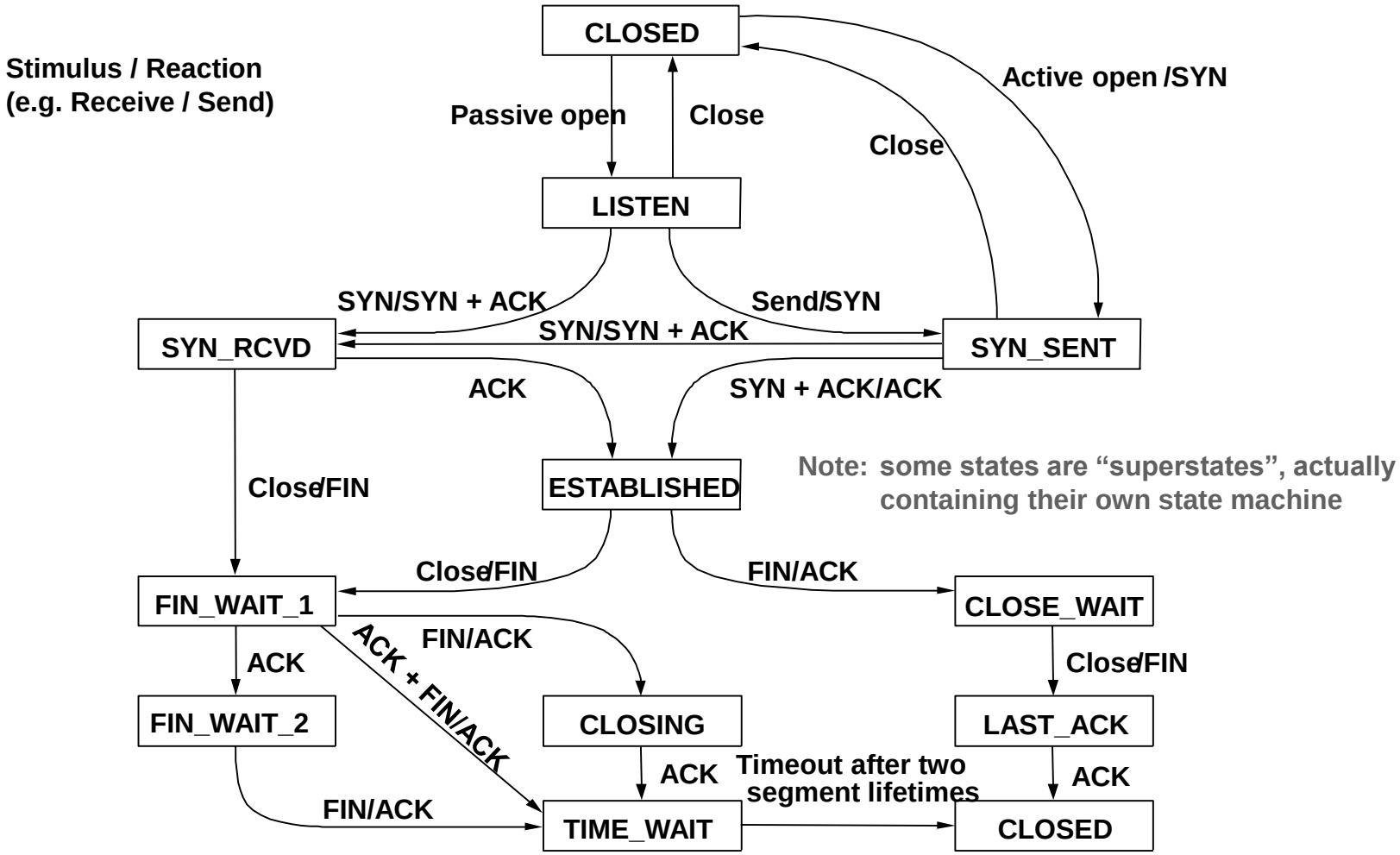
Depleting Memory: TCP's Three-Way-Handshake

- The *Transmission Control Protocol (TCP):*
  - provides a connection-oriented, reliable transport service
  - uses IP for transport of its PDUs
- TCP connection establishment is realized with handshake:



- After handshake, data can be exchanged in both directions
- Both peers may initiate termination of the connection (two-way-handshake)

TCP Connection Management: State Diagram



Background: Reaction According to Protocol

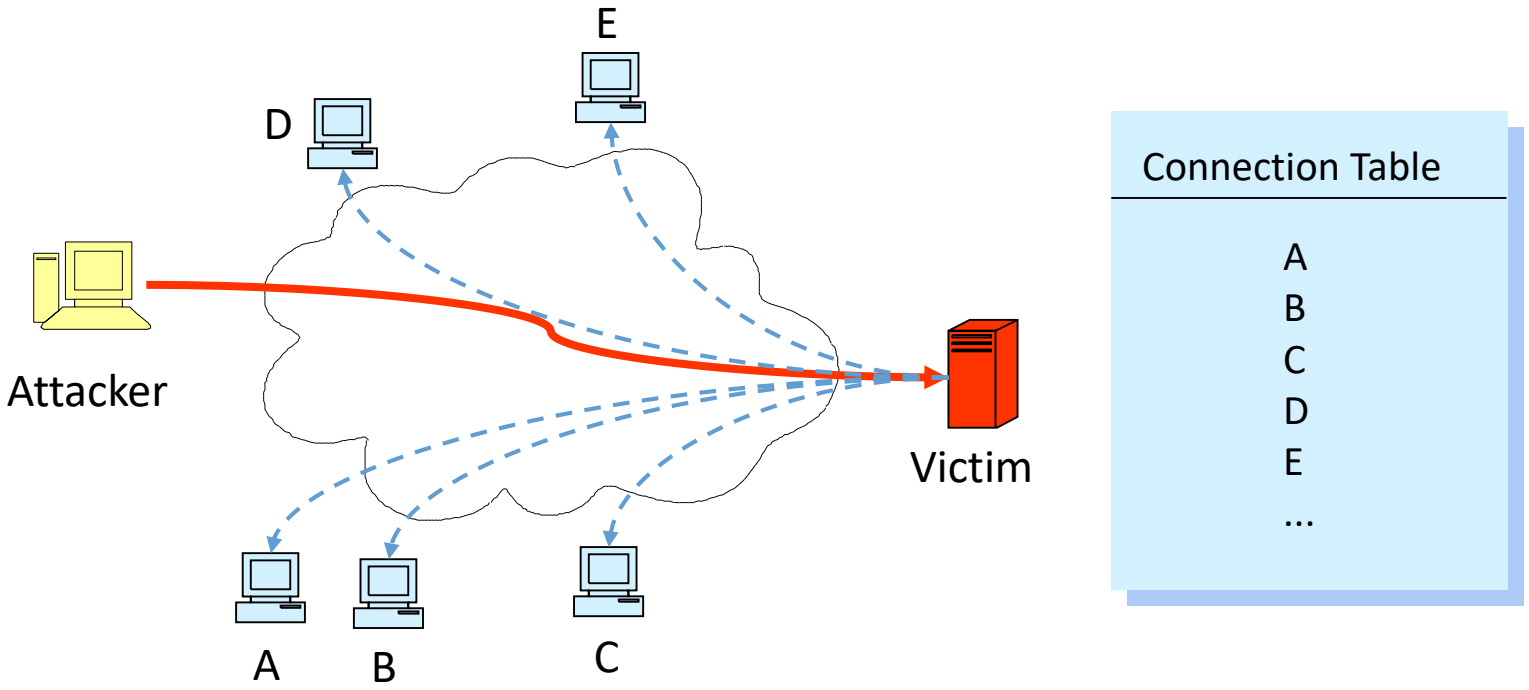
Karlsruhe Institute of Technology

Reply packets according to protocol specification if state not available

Packet Sent	Reaction of Receiver
TCP SYN (to open port)	TCP SYN ACK
TCP SYN (to closed port)	TCP RST (ACK)
TCP ACK	TCP RST (ACK)
TCP DATA	TCP RST (ACK)
TCP RST	no response
TCP NULL	TCP RST (ACK)
ICMP Echo Request	ICMP Echo Reply
ICMP TS Request	ICMP TS Reply
UDP Packet (to open port)	protocol dependent
UDP Packet (to closed port)	ICMP Port Unreachable
TCP SYN ACK (to closed port)	----

TCP SYN Flood: Memory Depletion

- Category *Storage of useless state information:*
  - Here: TCP-SYN flood attack



More recent Memory Depletion DoS Attacks

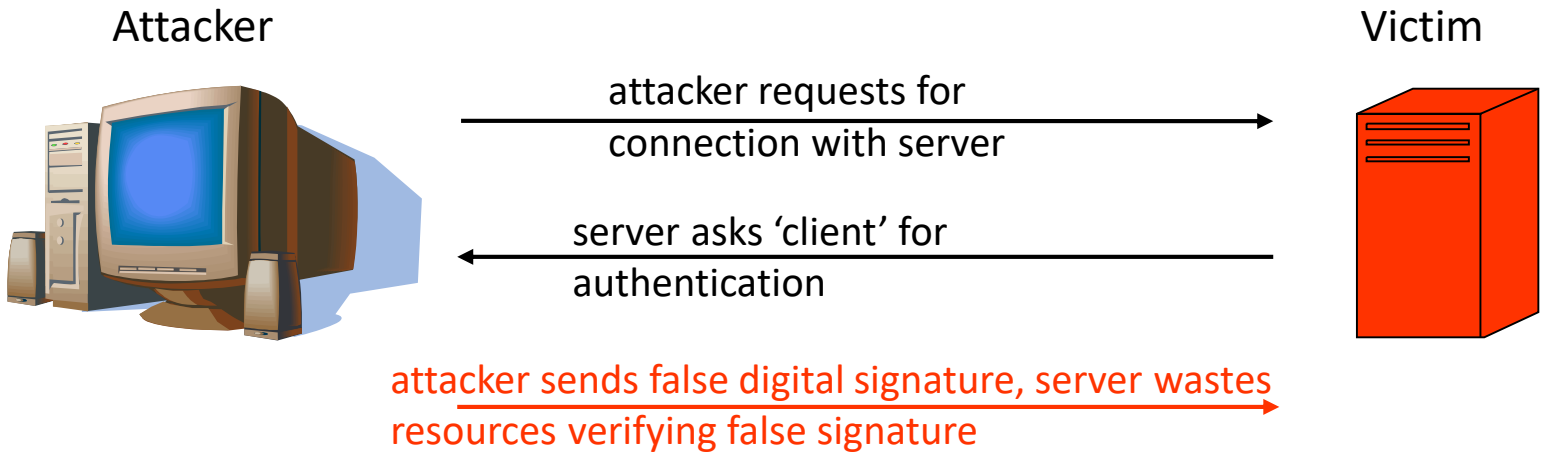
- Zip bombs (see above)
  - Exploit recursive/nested compression to create very large output
  - Recently also with overlapping files (non-recursive)

- „A billion laughs“
  - „XML bomb“
  - Exponential entity expansion attack on parsers

https://www.bamsoftware.com/hacks/zipbomb/

DDoS: CPU Exhaustion

- Category *CPU exhaustion by expensive computations:*
  - Here: attacking with bogus authentication attempts



  - The attacker usually either needs to receive or guess some values of the second message, that have to be included in the third message for the attack to be successful
  - Also, the attacker, must trick the victim repeatedly to perform the expensive computation in order to cause significant damage

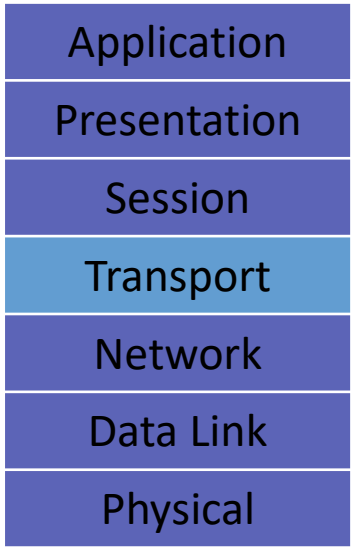
Background: Secure Socket Layer (SSL)

SSL was designed in the early 1990's to primarily protect HTTP sessions and it provides the following security services:

- *Peer entity authentication:*
  - ***Prior to any communications*** between a client and a server, an ***authentication protocol*** is run to authenticate the peer entities
  - Upon successful completion of the authentication dialogue an *SSL session* is established between the peer entities
- *User data confidentiality:*
  - If negotiated upon session establishment, user ***data is encrypted***
  - Different encryption algorithms can be negotiated: RC4, 3DES, AES, ...
- *User data integrity:*
  - ***HMAC*** based on a cryptographic hash function is appended to user data
  - The MAC is computed with a negotiated secret in prefix-suffix mode
  - Either MD5 or SHA can be negotiated for MAC computation

Background: Transport Layer Security

- Transport layer provides end-to-end communication between application processes
- Main tasks
  - Isolation of higher protocol layers
  - Transparent transmission of user data
  - Global addressing of application processes
  - Overall goal: provisioning of an efficient and reliable service
- *Transport layer security protocols* aim on enhancing service of the transport layer by assuring additional security properties



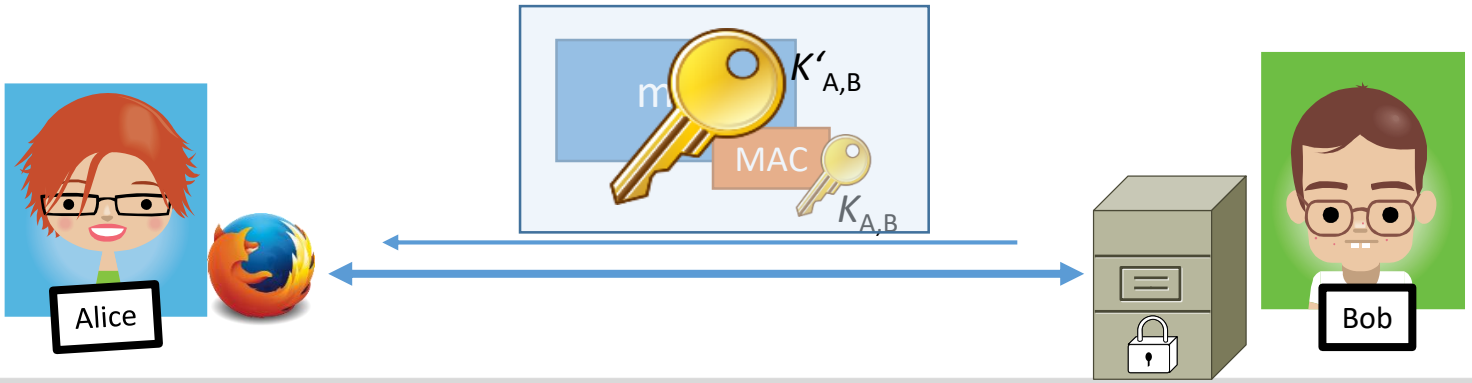
OSI-Model by layer

- Security protocols at transport layer: SSL, TLS, DTLS, SSH

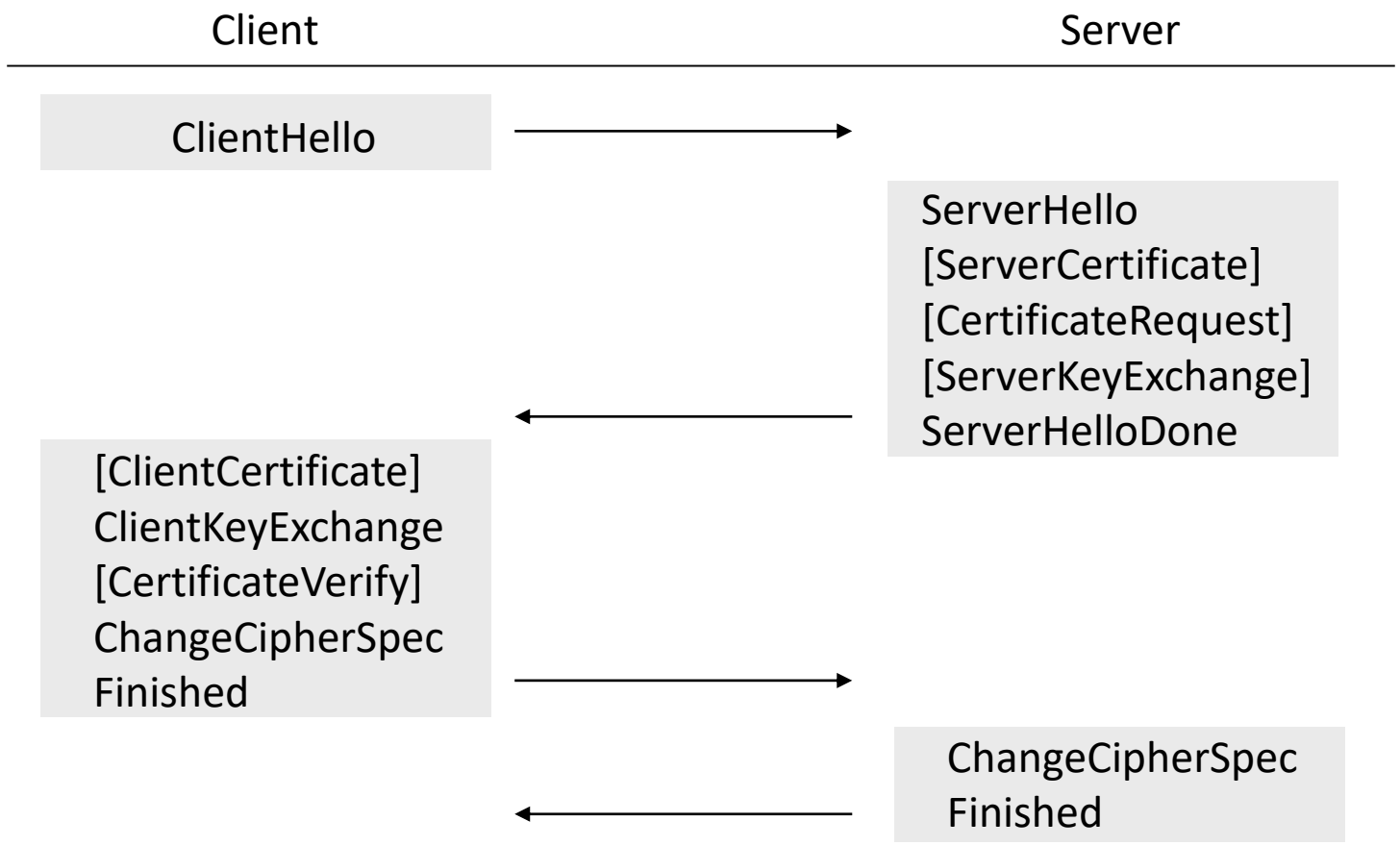
- History
  - SSL was designed in the early 1990's to primarily protect HTTP sessions
  - In 1996 the IETF decided to specify a generic *Transport Layer Security (TLS)* protocol that is based on SSL

SSL/TLS Security Services

- *Peer entity authentication:*
  - Prior to any communications between client and server, authentication protocol is run to authenticate the peer entities
  - Upon successful completion of authentication dialogue *SSL session* is established
- *User data integrity:*
  - A MAC based on a cryptographic hash function is appended to user data
  - The MAC is computed with a negotiated secret in prefix-suffix mode
  - Either MD5 or SHA can be negotiated for MAC computation
- *User data confidentiality:*
  - If negotiated upon session establishment, user data is encrypted
  - Different encryption algorithms can be negotiated: RC4, DES, 3DES, IDEA

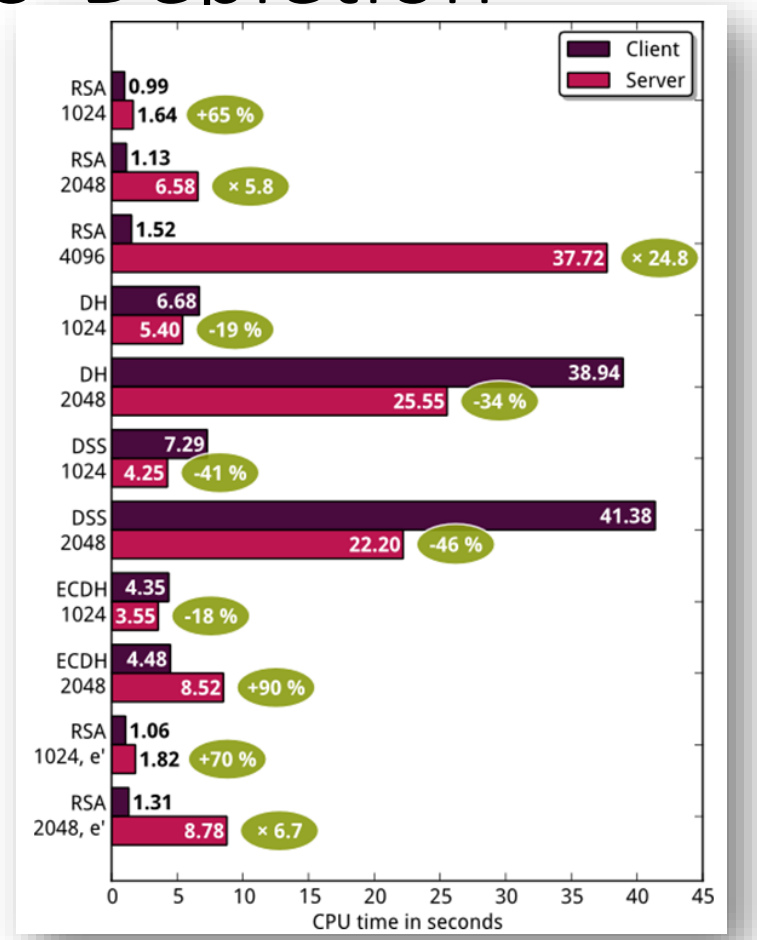


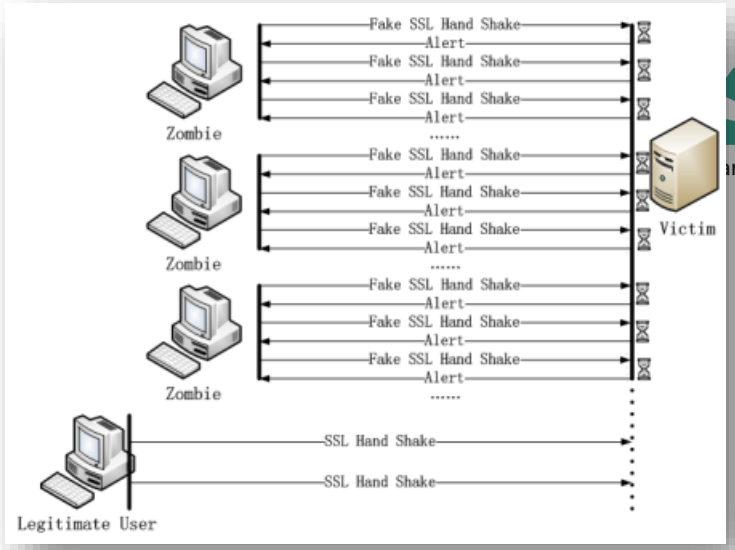
SSL Authentication: Full Handshake

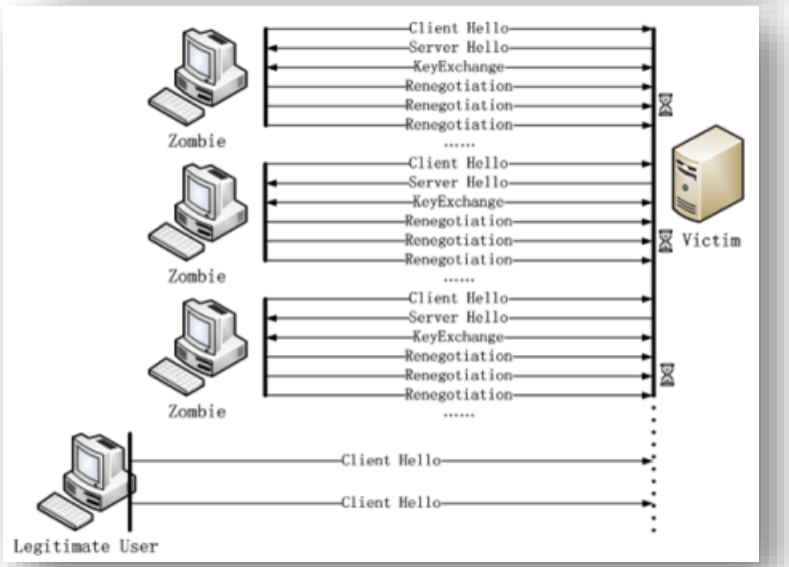


[...] denotes optional messages

SSL CPU-Depletion







THC-SSL-DOS

More recent CPU Exhaustion Attacks…

Nexus Intelligence Insights: CVE-2018-1109-Braces Regular expression Denial of Service (ReDoS) attack

by Elisa Velarde on June 28, 2019

Summ
once
Expre

What r
and filt
slighte

ReDoS
the po
consur
occur

Name

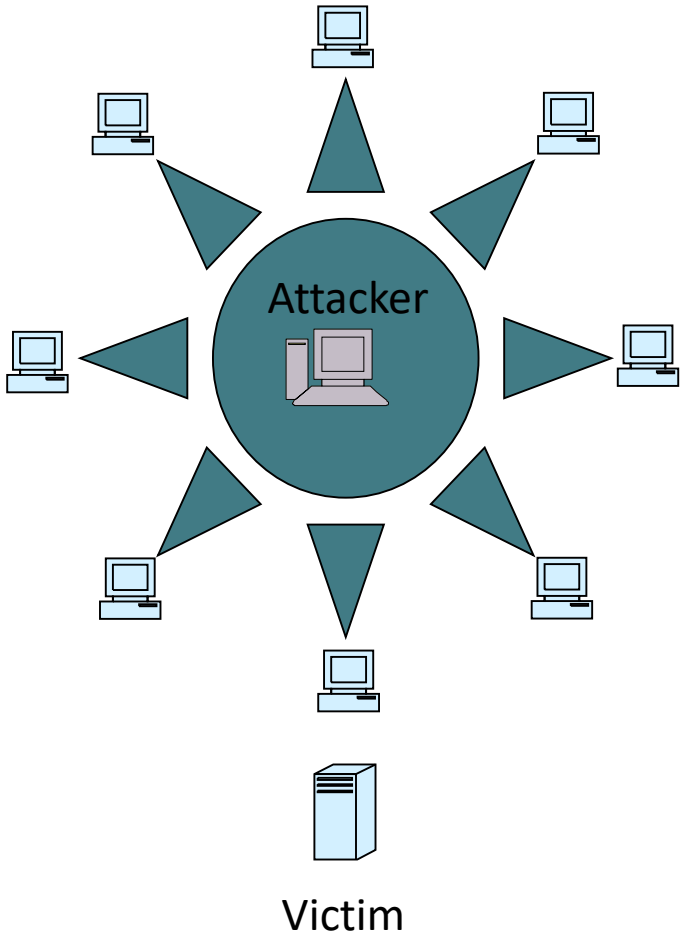
Cisco Security

Cisco Security Advisories and Alerts

Advanced Search

ADVISORY/ALERT	IMPACT	CVE	LAST UPDATED	VERSION
Search Advisory/Alert Name	All	Search CVE	Most Recent	
Cisco ASA and FTD Software Cryptographic TLS and SSL Driver Denial of Service Vulnerability	High	CVE-2019-1873	2019 Jul 11	1.1
Cisco IOS XR Software BGP MPLS-Based EVPN Denial of Service Vulnerability	High	CVE-2019-1849	2019 Jul 10	1.1
Multiple Issues in Cisco Small Business 250/350/350X/550X Series Switches Firmware and Cisco FindIT Network Probe	Informational		2019 Jul 09	1.1
Cisco Unified Communications Manager Session Initiation Protocol Denial of Service Vulnerability	High	CVE-2019-1887	2019 Jul 08	1.1
Cisco IP Phone 7800 and 8800 Series Session Initiation Protocol Denial of Service Vulnerability	Medium	CVE-2019-1922	2019 Jul 08	1.1
Cisco Web Security Appliance HTTPS Certificate Denial of Service Vulnerability	High	CVE-2019-1886	2019 Jul 03	1.0
Cisco Small Business Series Switches Memory Corruption Vulnerability	High	CVE-2019-1892	2019 Jul 03	1.0
Cisco Small Business Series Switches HTTP Denial of Service Vulnerability	High	CVE-2019-1891	2019 Jul 03	1.0

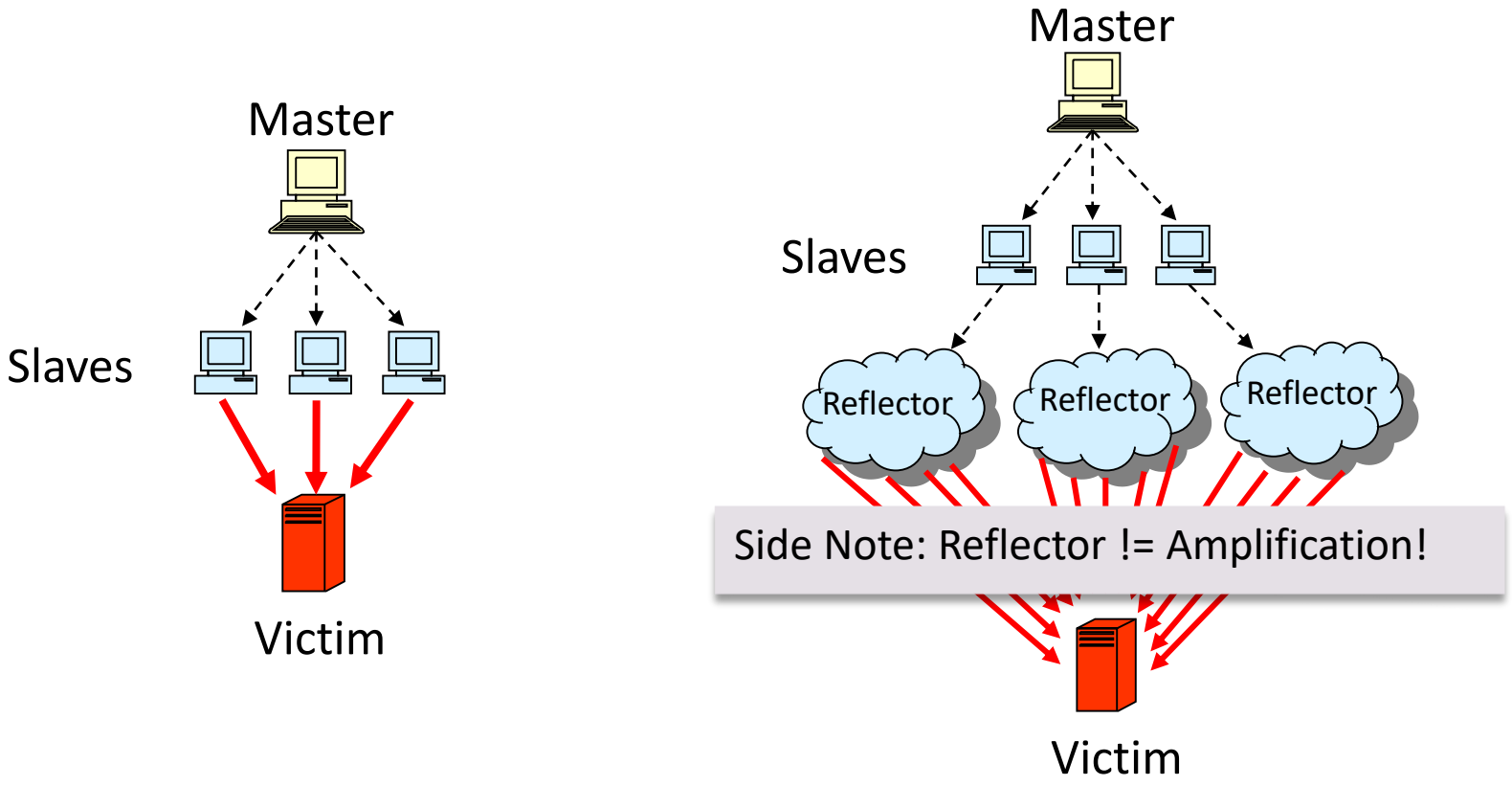
Examples: Resource Depletion with DDoS (1)



- Attacker intrudes multiple systems by exploiting known flaws
- Attacker installs DoS-software:
  - „Root Kits“ are used to hide the existence of this software
  - Very often DoS software makes system part of a Botnet
- DoS-software is used for:
  - Exchange of control commands
  - Launching an attack
  - Coordinating the attack

Examples: Resource Depletion with DDoS (4)

Different Attack Network Topologies



a) Master-Slave-Victim

b) Master-Slave-Reflector-Victim

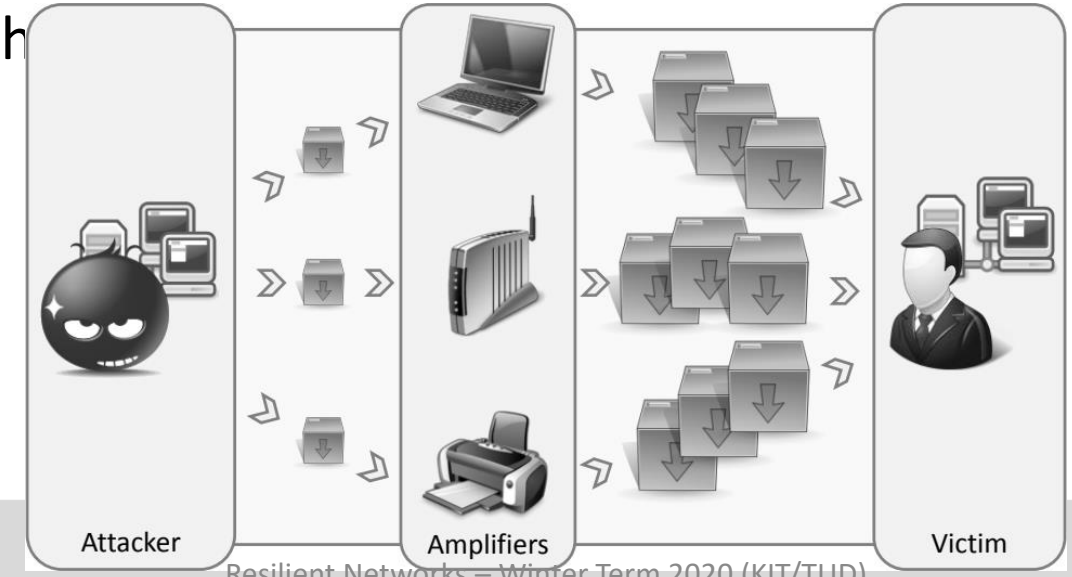
Distributed Reflective Denial-of-Service (DR-DoS)

DRDoS - Amplification Attacks (1)

[Ros14]

- Use available public services on the Internet, e.g., open DNS resolvers
- Distributed Reflective Denial-of-Service (DR-DoS)

- Attack:
  1. Attacker sends few spoofed small requests in the name of the victim
  2. The reflectors reply accordingly to the protocol
  3. Amplification of th nd size



DRDoS - Amplification Attacks (2)

- Amplification Factors
  - Bandwidth amplification factor

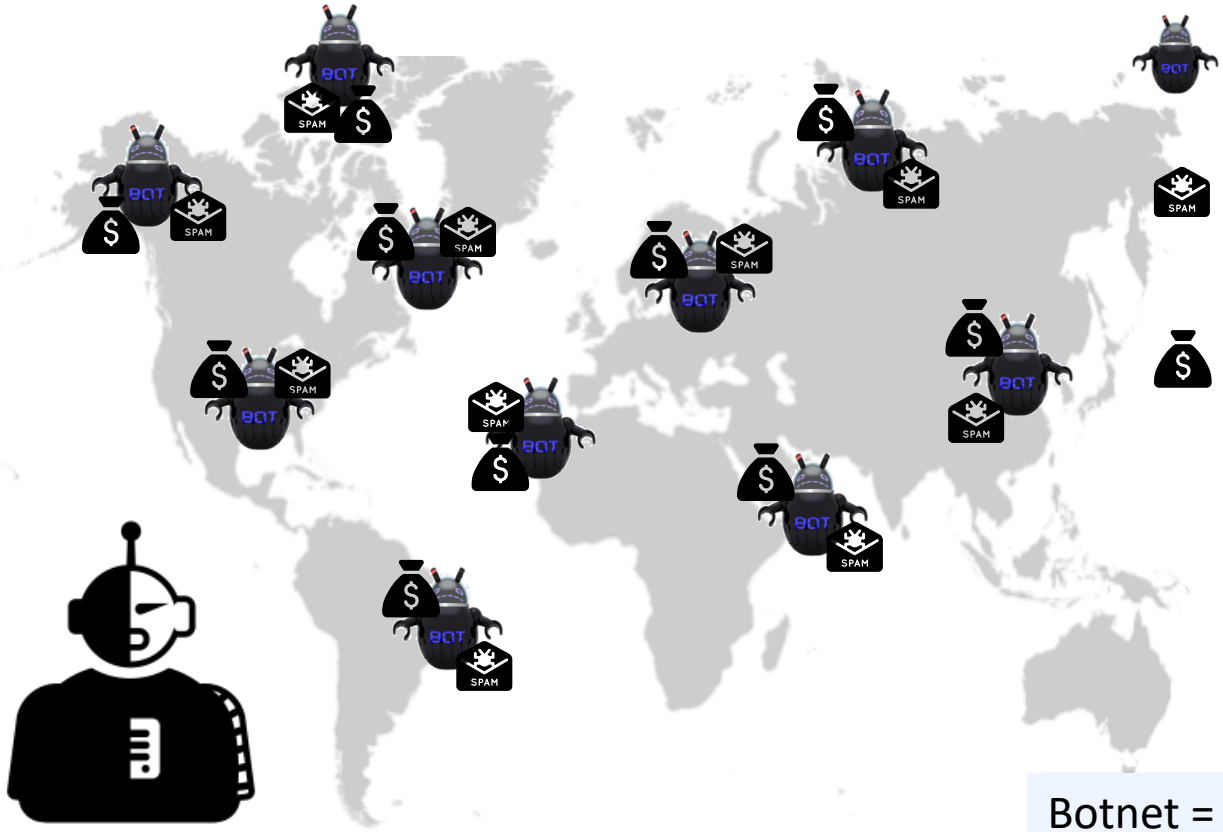
$$BAF = \frac{len(UDP\ payload)\ amplifiers\ to\ victim}{len(UDP\ payload)\ attacker\ to\ amplifier}$$

  - Packet amplification factor

$$PAF = \frac{number\ of\ packets\ amplifier\ to\ victim}{number\ of\ packets\ attacker\ to\ amplifier}$$

Protocol	BAF *all*	BAF 50%	BAF 10%	PAF *all*	Scenario	Amplifiers
SNMP v2	6.3	8.6	11.3	1.00	*GetBulk* request	4,832,000
NTP	556.9	1083.2	4670.0	3.84	Request client statistics	1,451,000
DNS_{NS}	54.6	76.7	98.3	2.08	ANY lookup at author. NS	255,819
DNS_{OR}	28.7	41.2	64.1	1.32	ANY lookup at open resolv.	7,782,000
NetBios	3.8	4.5	4.9	1.00	Name resolution	2,108,000
SSDP	30.8	40.4	75.9	9.92	*SEARCH* request	3,704,000
CharGen	358.8	n/a	n/a	1.00	Character generation request	89,000
QOTD	140.3	n/a	n/a	1.00	Quote request	32,000
BitTorrent	3.8	5.3	10.3	1.58	File search	5,066,635
Kad	16.3	21.5	22.7	1.00	Peer list exchange	232,012
Quake 3	63.9	74.9	82.8	1.01	Server info exchange	1,059
Steam	5.5	6.9	14.7	1.12	Server info exchange	167,886
ZAv2	36.0	36.6	41.1	1.02	Peer list and cmd exchange	27,939
Sality	37.3	37.9	38.4	1.00	URL list exchange	12,714
Gameover	45.4	45.9	46.2	5.39	Peer and proxy exchange	2,023

Botnets



Conficker: 15 million infected computers (in 2009)

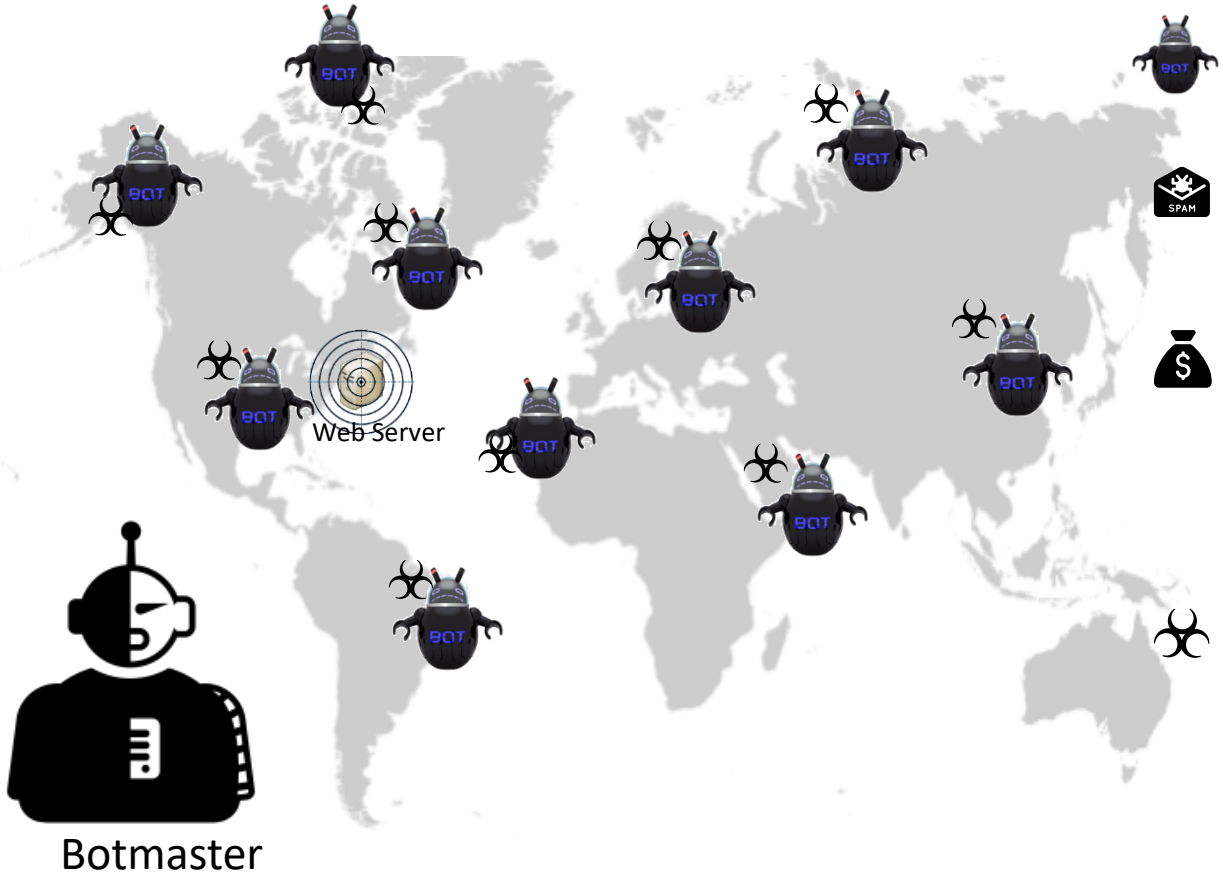
Grum: capable of 39.9 billion messages a day – up to 26% of world's email spam (in 2010)

ZeroAccess: $100k a day (in September 2012)

Dridex (Bugat v5): Banking Trojan caused $30.5 million lost (in October 2015)

Botnet =
Army of infected machines (Bots)
Controlled by one instance
Automatic update mechanism

Botnets



Conficker: 15 million infected computers (in 2009)

Grum: capable of 39.9 billion messages a day – up to 26% of world's email spam (in 2010)

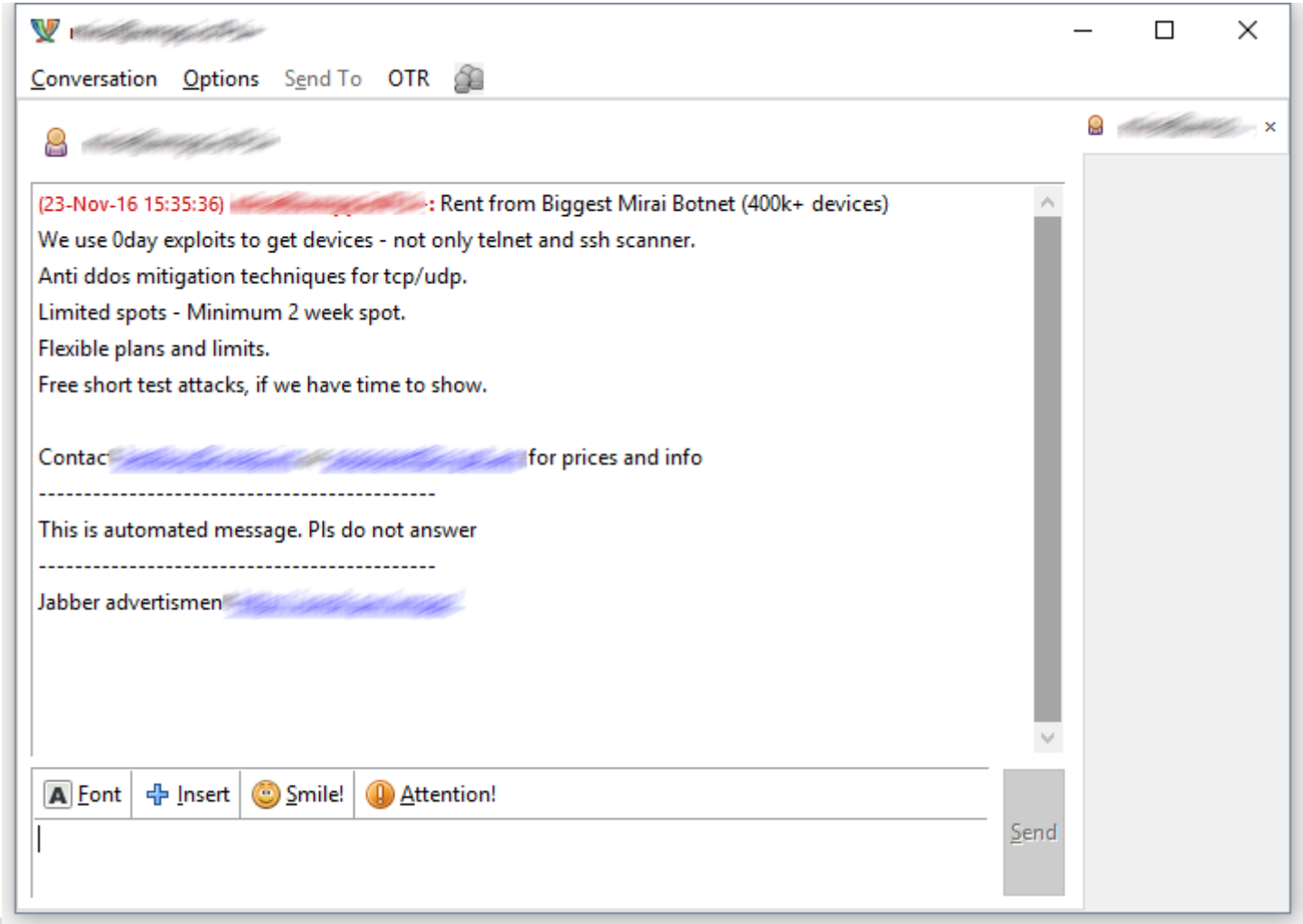
ZeroAccess: $100k a day (in September 2012)

Dridex (Bugat v5): Banking Trojan caused $30.5 million lost (in October 2015)

Becomes even worse:
Mirai: IoT botnet for DDoS

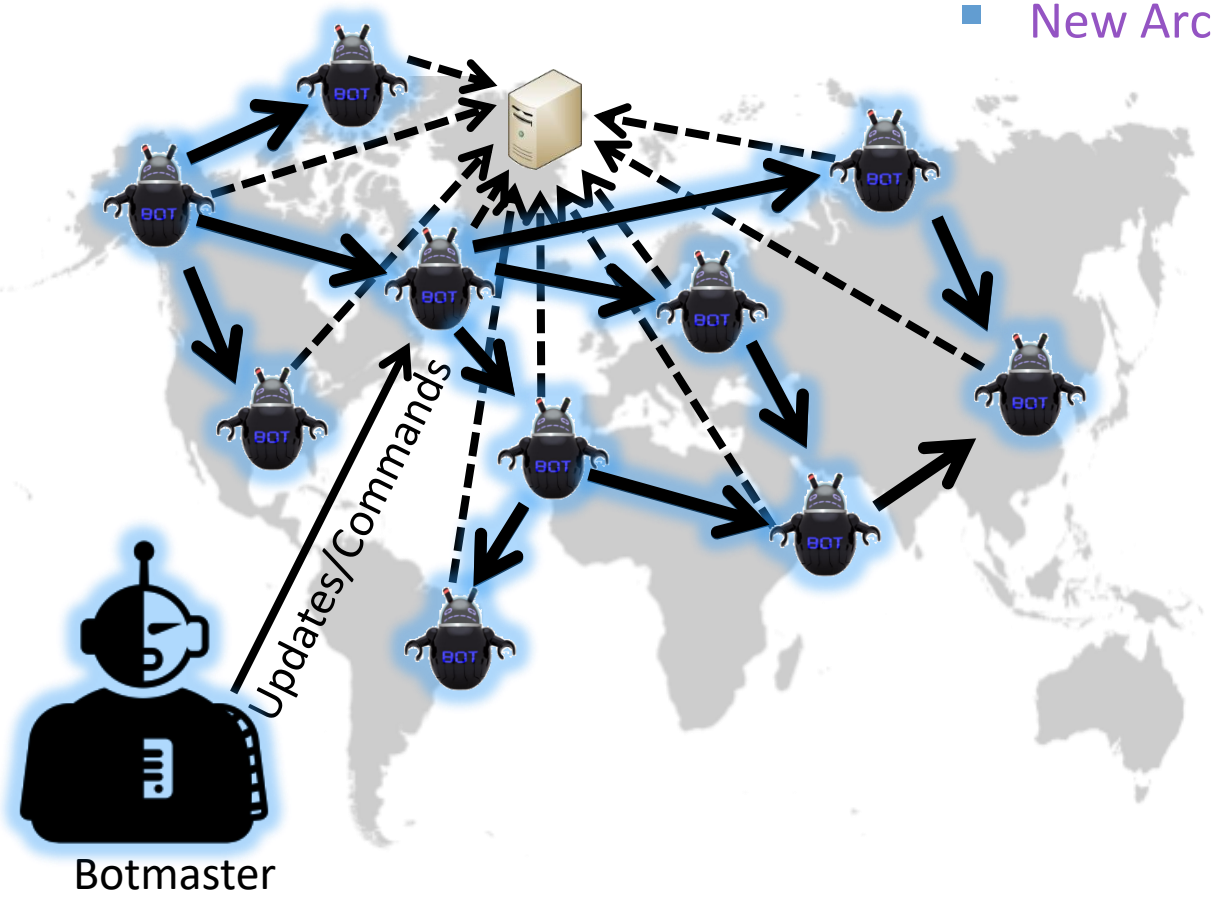
620 Gbps on KrebsOnSecurity
and
1.1Tbps on hoster OVH
(both in September 2016)

Mirai Botnet Advertisement



More recent: P2P-Botnets

- Traditionally centralized
- New Architecture: P2P Overlay



Defending Against Resource Depletion DoS

Karlsruhe Institute of Technology

- Defenses against resource depletion:
- Generally:
  - ***Rate Control*** (ensures availability of other functions on same system)
  - Authentication & Accounting

- Expensive computations: careful protocol design, verifying the initiator's "willingness" to spend resources himself (e.g. "client puzzles")

- Memory exhaustion: stateless protocol operation

Attack Sources and Spoofed Addresses

Karlsruhe Institute of Technology

- Concerning origin of malicious traffic:
- Defenses against single source attacks:
  - Disabling of address ranges (helps if addresses are valid)

- Defenses against forged source addresses:
  - Ingress Filtering at ISPs (if the world was an ideal one...)
  - “Verify” source of traffic (e.g. with exchange of “cookies”)
  - Tracing back the true source of packets with spoofed addresses

- Widely distributed DoS:
  - Offloading to Site Delivery Services/CDN

Memory Exhaustion: Stateless Protocols

Karlsruhe Institute of Technology

- Basic idea:
  - Avoid storing information at server, before DoS attack can be ruled out
  - So, as long as no assurance regarding the client has been reached all state is “stored” in the network (transferred back and forth)

Stateful Operation	Stateless Operation
1. C → S: Msg_1	1. C → S: Msg_1
2. S → C: Msg_2	2. S → C: Msg_2 , $State_{S1}$
S stores $State_{S1}$	
3. C → S: Msg_3	3. C → S: Msg_3 , $State_{S1}$
4. S → C: Msg_4	4. S → C: Msg_4 , $State_{S2}$
S stores $State_{S2}$	
...	...

- Drawback: requires higher bandwidth and more message processing

CPU Exhaustion: Client Puzzles/Proof of Work

Karlsruhe Institute of Technology

Observations and assumptions:

- DoS (also: spam) works because there's no postage paid (cost) when message is sent
- Amplification attacks require few resources at client and cause large load at victim

- *Proof of Work*: level the playing fields by making the clients prove that they invested resources
- One-way functions are cheap to evaluate, but "impossible" to invert

- Good (as any) approach to inversion is guessing, partial guessing may be possible:
  - Chances to guess x such that

$$P[H(x) = yyyyyyy0] = .5$$

what about $P[H(x) = yyyy000]$? ;-)

Simple Client Puzzles:

- Let server draw a pre-image at random
- Provide client with image and request it to provide the pre-image

KASTEL

Countering CPU Exhaustion with Client Puzzles (3)

Karlsruhe Institute of Technology

- Reusable client puzzles according to Aura et al:

1. Server periodically broadcasts random number N_S and difficulty level k
2. Every client C can then create a solution to a new instance of this puzzle by:
   - Generating a fresh random number N_C
   - Determining with brute force search (= trying all possible values) an X such that:

$$H(C, N_S, N_C, X) \overset{!}{=} \underbrace{00000}_{k} Y$$

- Summary:
  - Client puzzles provide an effective means to slow down potential DoS attackers significantly
  - At the same time, the length of messages is only increased minimally (about one byte for parameter k and up to eight bytes for the solution X)
  - This may protect servers at the early stage of a normal authentication where the computations are the most CPU intensive

Aura, Tuomas, Pekka Nikander, Jussipekka Leiwo, "DOS-resistant authentication with client puzzles." Workshop on security protocols. **2000**

KASTEL

Conclusion

- Increasing dependence of modern information society on availability of communication services

- While some DoS attacking techniques can be encountered with “standard” methods, some can not:
  - Hacking, exploiting implementation weaknesses, etc. may be encountered with firewalls, testing, monitoring etc.
  - Malicious protocol deviation & resource depletion is harder to defend against

- Designing DoS-resistant protocols emerges as a crucial task for network engineering:
  - Network protocol functions and architecture will have to be (re-)designed with the general risk of DoS in mind
  - Base techniques: stateless protocol design, cryptographic measures like authentication, cookies, client puzzles, etc.

Verifying the Source of a Request

- Problem: Spoofed addresses allow adversaries to hide

- Basic solution:
  - Before working on a new request, verify if the “initiator” can ***receive messages,*** sent to the claimed source of the request

  - Only a legitimate client or an attacker which can receive the “cookie”, can send the cookie back to the server
  - Of course, an attacker must not be able to guess the content of a cookie

- Discussion:
  - Advantage: allows to counter simple spoofing attacks
  - Drawback: requires one additional message roundtrip

But...

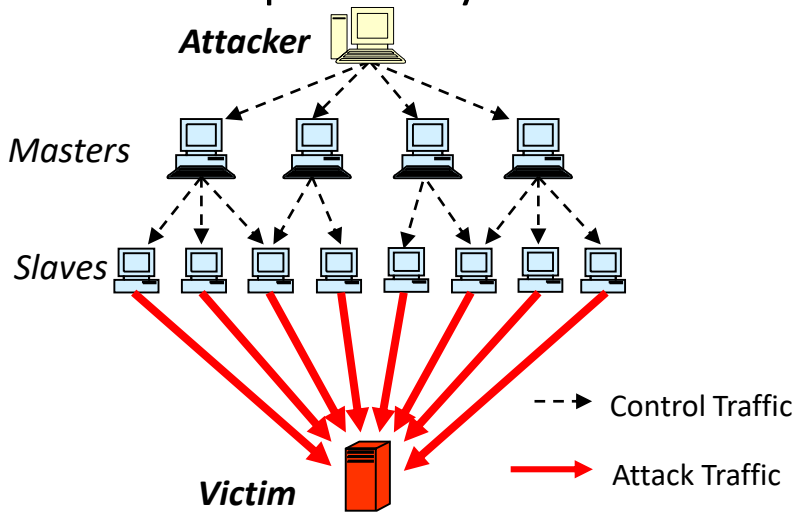
- Verifying the source of a request with a cookie exchange can ***avoid spending significant computation*** or ***memory*** resources on a bogus request

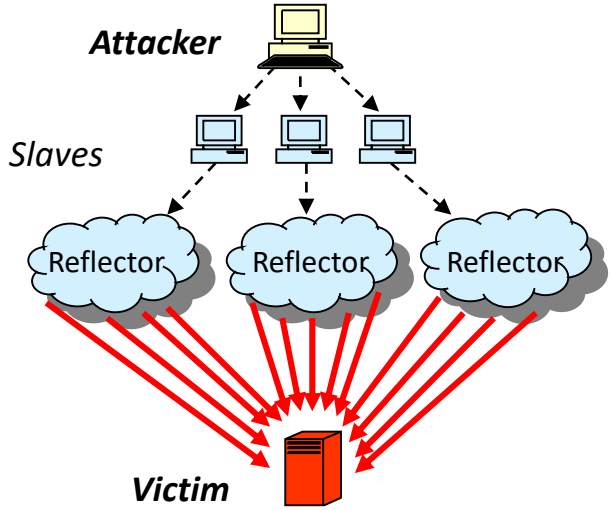
- What if the attacker is only interested in ***exhausting*** the access or packet processing ***bandwidth*** of a victim?
  - Obviously, sending cookies to all incoming packets even aggravates the situation!
  - Such an attack situation, however, is quite easy to detect: there are simply too many packets coming in

- Problems in such a case:
  - Which packets come from ***genuine sources*** and which are ***bogus ones***?
  - Even worse: source addresses given in the packets may be spoofed
  - Where do the spoofed packets come from?

IP-Address Spoofing

- Reprise: DoS-/ DDoS-Attacks
  - Direct Attacks (Master – network of slaves)
    - Problem of spoofed source addresses of attack packets sent by the slaves
  - Reflector Attacks (Master – (slaves –) reflecting nodes)
    - Problem of address-spoofing: set victims‘ IP-address as source
- Main problem is the possibility to lie about the source address…





Possible Solutions to DDoS-Attacks (1)

Karlsruhe Institute of Technology

- Solutions to ***Reflector Attacks***: secure available services
  - Prevent amplification: Balance effort of request and reply

    e.g.: Prohibit ICMP-Echo-Request to broadcast addresses
    - => Reflectors don't amplify attack magnitude

      (however: does this work with all protocols? DNS?)

  - Access-controlled services: provide service to authorized parties only

    e.g.: Prohibit recursive DNS queries for external users

Possible Solutions to DDoS-Attacks (2)

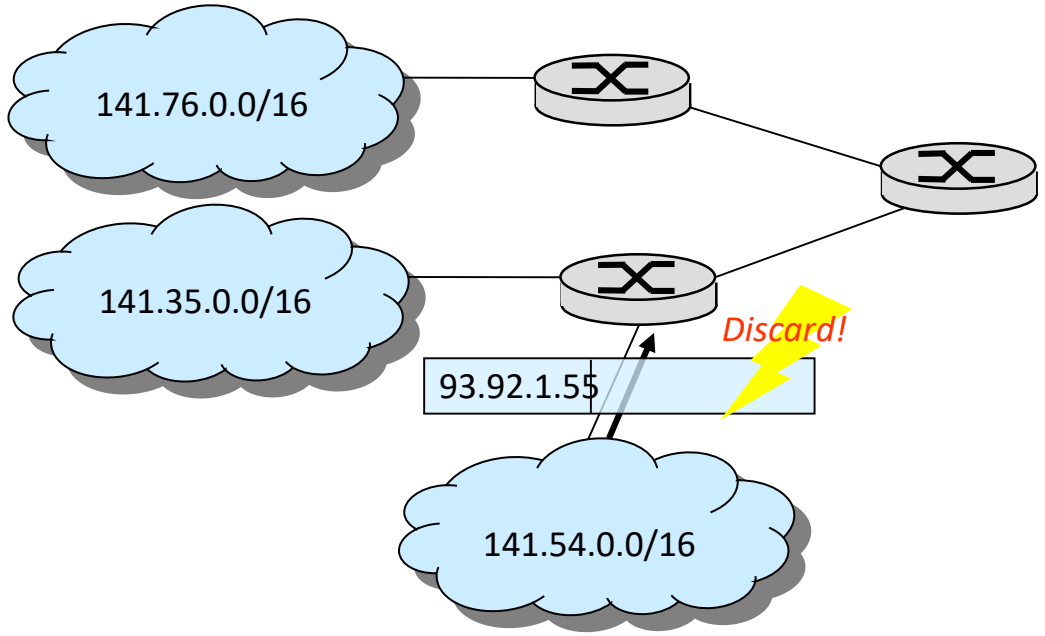
Karlsruhe Institute of Technology

- Possible Solutions to ***Direct Attacks***:
  - Avoid IP-Address spoofing
  - Live with spoofed addresses and restrain effect of attacks
    - Locate source of attack-packets
    - Filter traffic from attacking nodes
    - Inform admin/root of attacking networks/node

- But: IP is connectionless! Necessary to find means to trace back the traffic to the original source / attacking node!
- Identify: zombie, spoofed address, ingress router, routers on path…

Inhibiting Spoofed Addresses: Ingress Filtering (RFC 2267)

- Routers block arriving packets with illegitimate source addresses.



- IETF BCP 38 (May 2000)

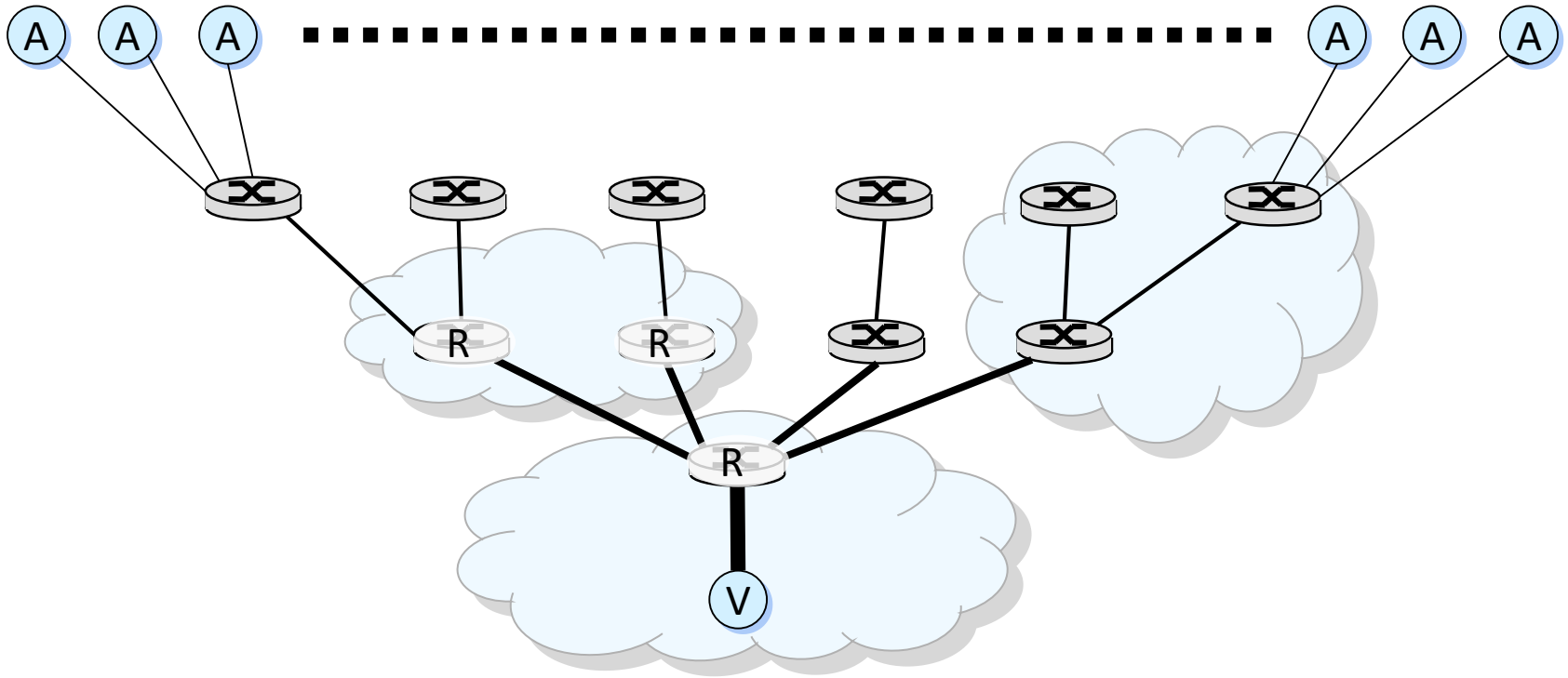
Ingress Filtering (2)

- Difficult in the backbone *(how to check if route is valid?)*
- Easily possible at access links → ISPs

- Problems occur:
  - Issues with Mobile-IP (theoretic) and load testing (local)
  - Large management overhead at router-level
  - Processing overhead at access routers
    - (e.g., big ISP running a large AS with numerous IP-Ranges and DHCP)
  - Universal deployment needed (cf. the situation today…)

- *ISPs don't really have an incentive in blocking any traffic*

Identify Malicious Nodes: DDoS Attack-Tree



- Rooted Tree with
  - Victim *(V)* (root of the tree)
  - Routers *(R)*
  - Attackers *(A_i)*

Questions with forged IP addresses:

- Where are malicious nodes?
- Which router (ISP) is on attack path?

Identifying Malicious Nodes: Assumptions

Karlsruhe Institute of Technology

- Packets are subject to ***reordering and loss***
- ***Resources*** at routers are ***limited***
- ***Routers*** are usually ***not compromised***
- Attackers may ***generate any packet***
- Attackers are ***aware of tracing***
- ***Multitude*** of ***attacking packets*** (usually many)
- ***Routes*** between A and V are ***stable*** (in the order of seconds)
- Multiple attackers can act in ***collusion***

KASTEL

Identify Malicious Nodes: Proposed Solutions

Karlsruhe Institute of Technology

Simple classification of solutions:

- Network Logging
  - Log information on processed packets and path
- Attack Path Traceback
  - Trace attack path through network
- Other / Related
  - Attack Mitigation/Avoidance

KASTEL

Requirements / Evaluation Metrics

Karlsruhe Institute of Technology

1. Involvement of ISP (required or not)
2. Amount of necessary packets to trace attack
3. Effect of partial deployment
4. Resource overhead
    - Processing overhead at routers
    - Memory requirements
    - Bandwidth overhead
5. Ease of Evasion
6. Protection
7. Scalability
8. Performance towards Distributed DoS
9. Performance towards packet transformations

KASTEL

Involvement of ISP

- ISPs don‘t really have an incentive in preventing „attack-traffic“:
- Paid by number of transmitted bytes
- Receive complaints about service failures (churn!)
- Which traffic is „malicious“ and which is not?
- „Malicious“ for whom?

- Incentives of ISPs:
- Infrastructure is expensive
- Management-/ down times are expensive
- Administrators are expensive

Amount of Packets Needed to Track Source

Karlsruhe Institute of Technology

- Different types of attacks:

- Bandwidth resource exhaustion
  - Continuous stream of packets for the time span of the attack
  - Packet flood to bring link / host down
  - One attacker / multiple attackers (multiple attack paths)

- Well targeted packets (resource destruction, e.g. Teardrop attack)

- Which attacker can be traced?

Effect of Partial Deployment

Karlsruhe Institute of Technology

- What if only a few ISPs deploy the mechanism (at first)?

- Still *some* benefit?
  - Attackers in the deploying ISPs traceable?
  - Ingress of attack packets traceable?
  - Cooperation of „islands“ possible – gain in knowledge if two ISPs deploy mechanism which are connected through a third transit domain?

Resource Overhead

- Resources in the network are scarce (memory, processing)!
- How much processing overhead is implied for the routers
  - Additional packet analysis
  - Additional functions
- How much information has to be stored at routers / in the network
  - Log of all processed packets?
- If mechanism needs communication:
  - In band / out of band?
  - How much extra bandwidth is needed to distribute information?

Ease of Evasion, Protection & Scalability

Karlsruhe Institute of Technology

- **Ease of Evasion**:
  - How easy is it for an attacker to evade the mechanism?
  - Can the attacker send special packets that mislead the mechanism?
    - To stay transparent
    - To mislead an investigator
    - Attack the mechanism itself

- **Protection**:
  - What if an attacker subverts one or many network elements on the path: Can the mechanism still produce meaningful results?

- **Scalability:**
  - Does the mechanism scale with growing network sizes?
  - How much extra configuration is needed (only at new, or at all devices?)
  - How much do the elements depend on each other?

Performance: DDoS and Packet Transformation

Karlsruhe Institute of Technology

- Ability to handle DDoS:
  - Can the mechanism produce meaningful results, if a victim is attacked on different paths?

- Ability to handle packet transformation:
  - Does the mechanism produce meaningful results (results at all) if the packets are transformed due to:
    - Network Address Translation (NAT)
    - Packet fragmentation
    - Packet duplication
    - Tunneling

Identifying Malicious Nodes: Proposed Solutions

Karlsruhe Institute of Technology

- Network Logging
  - Local network logging
  - Aggregated network logging
  - Source Path Identification („Hash-based IP-Traceback“)
- Attack Path Traceback
  - Input Debugging
  - Controlled Flooding
  - ICMP Traceback
  - Probabilistic Packet Marking („IP-Traceback“)
- Other / Related
  - Hop-Count Filtering
  - Aggregate Based Congestion Control (ACC)
  - Secure Overlay Services

KASTEL

Logging Approaches

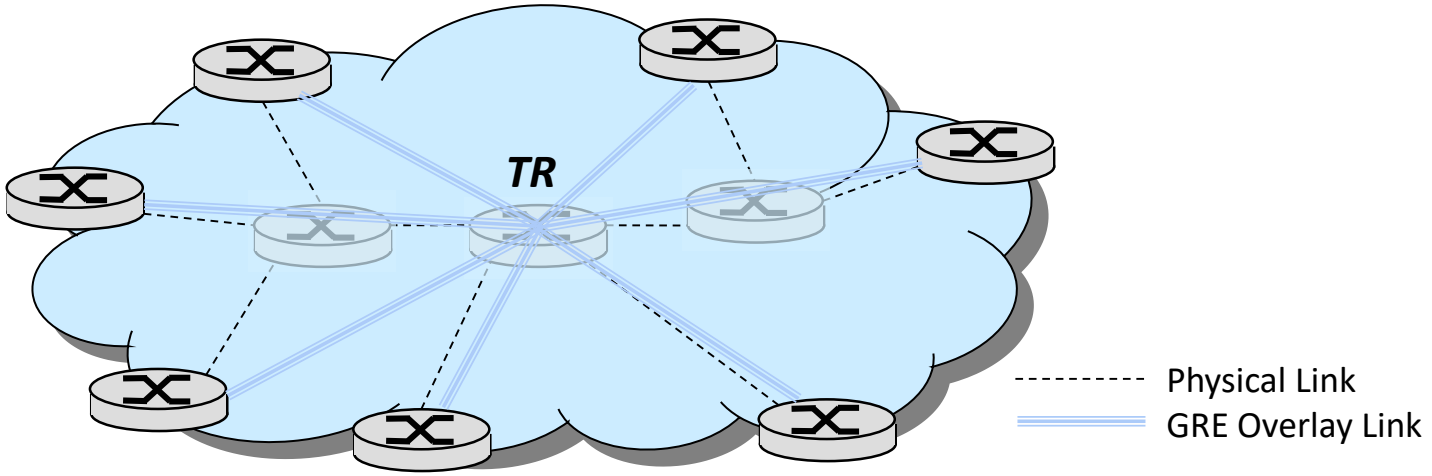
Karlsruhe Institute of Technology

- Log information on processed packets and path

- Network logging
  - Local network logging:
    - All routers log all traffic
    - Too much overhead!
    - **Does not scale**
  - Aggregated network logging
  - Source Path Identification („Hash-based IP-Traceback“)

Aggregated Network Logging

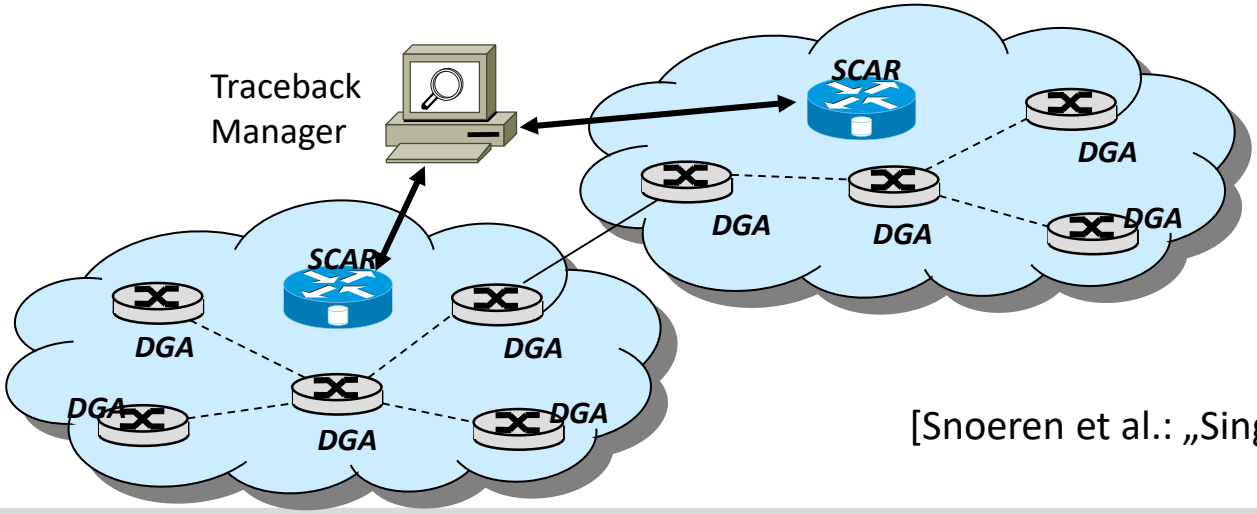
- Centralized approach:
  - Introduction of „Tracking Router“ (TR)
  - Forwarding all traffic through TR (*via GRE*)
  - TR logs all traversing traffic
  - Creates one single point of failure! Does not scale! *(Altough: SDN…)*



[Stone: „Centertrack: An IP Overlay Network for Tracking DoS Floods“]

Source Path Identification

- Source Path Identification Engine *(SPIE, aka Hash-based IP Traceback)*
- Storage of compressed data in specialized devices
  - DGA generate digests of data *(Data Generation Agent)*
  - SCAR for storage and retrieval *(SPIE Collection & Reduction Agents)*
  - STM for central management *(SPIE Traceback Manager)*



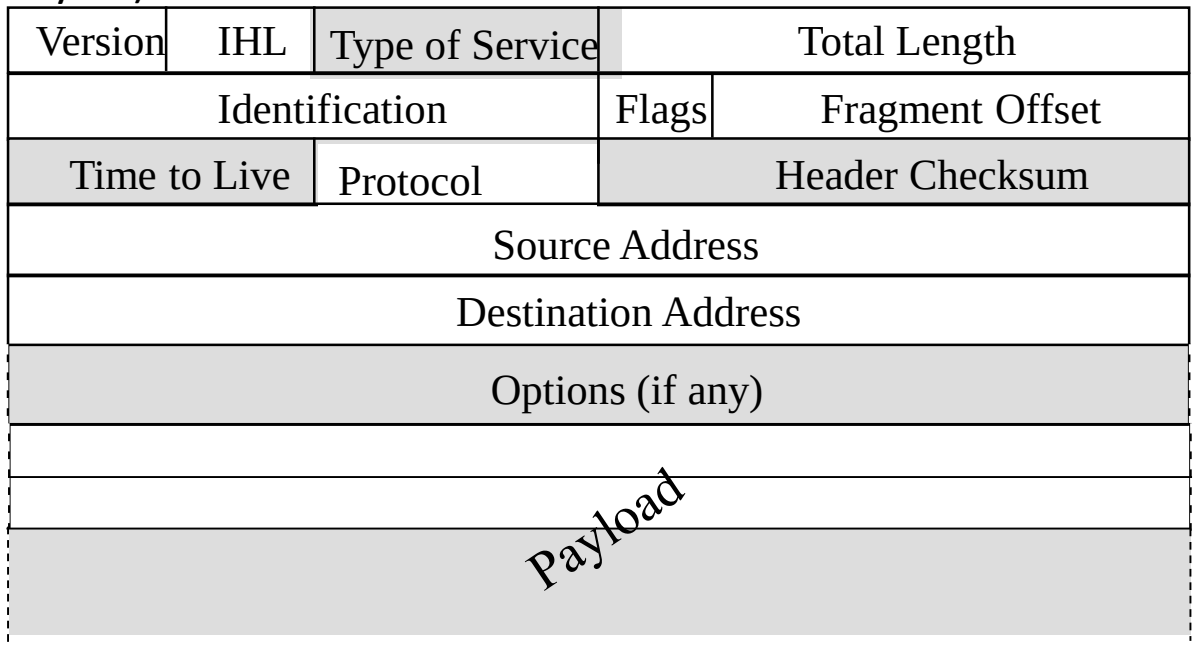
[Snoeren et al.: „Single-Packet IP-Traceback“]

Source Path Identification (2)

- *„Store all information on traversed packets?“*
- No! *What do we need to store?*
- Store digests of:
  - Constant fields in IP Header (16 bytes)
  - First 8 bytes of payload
- Still a lot, compress:

  Hashed in

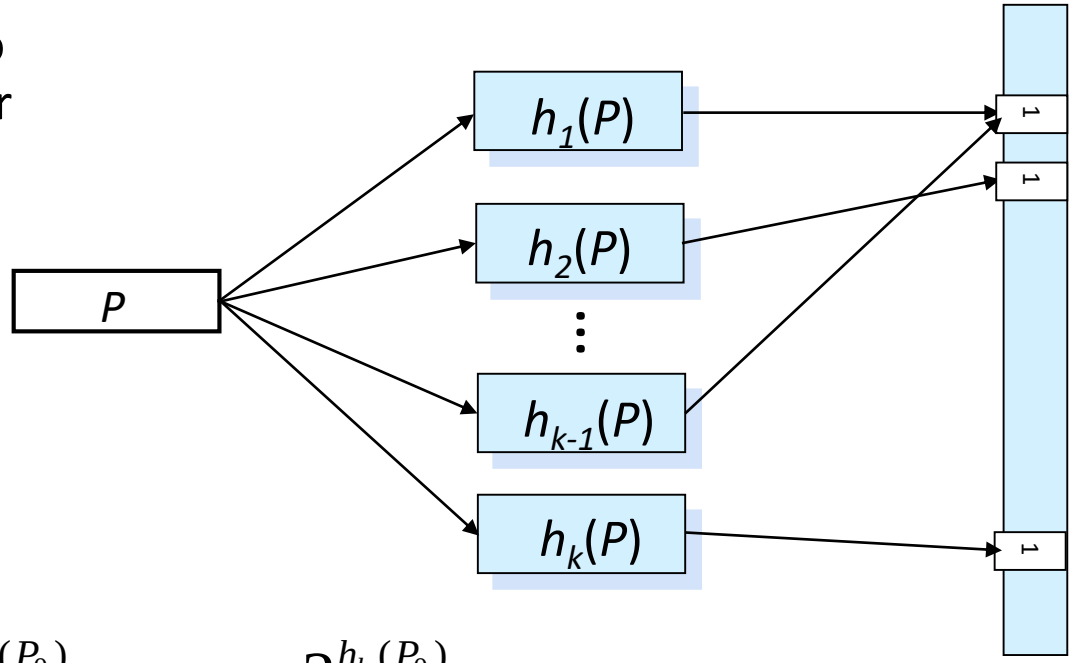
  Bloom Filters



Source Path Identification: *Bloom Filters* (1)

Karlsruhe Institute of Technology

- 24 bytes of each packet hashed with *k* hash functions h_i
- Hash values stored in filter:
  - To store $h_i(P)$, write a 1 into position $2^{h_i(P)}$ in bloom filter



$$BF(P_0) = 2^{h_1(P_0)} \text{ or } 2^{h_2(P_0)} \text{ or } \ldots \text{or } 2^{h_k(P_0)}$$

$$BF(P_n) = BF(P_{n-1}) \text{ or } 2^{h_1(P_n)} \text{ or } 2^{h_2(P_n)} \text{ or } \ldots \text{or } 2^{h_k(P_n)}$$

KASTEL

Source Path Identification: Bloom Filters (2)

Karlsruhe Institute of Technology

- During normal operation DGAs maintain bloom filters, if bloom filter more than 70% “full” (70% of the bits are set to “1”), sent to SCAR

- *Detection* if a specific packet was processed:
  - Hash packet with k hash functions h_i
  - If any of the corresponding bits in all stored bloom filters is 0: Packet has not been processed
  - All bits of a bloom filter are 1: Packet most probably traversed the DGA

- *Path retrieval*:
  - Victim contacts STM with pattern “P” of attack packet
  - STM distributes pattern “P” to SCARs
  - SCARs perform k hashes $h_1(P)$.. $h_k(P)$ to test which DGA forwarded matching packet

KASTEL

Traceback Approaches

- Trace attack path backwards through network

- Attack Path Traceback
  - Input Debugging
  - Controlled Flooding
  - ICMP Traceback
  - Probabilistic Packet Marking („IP-Traceback“)

Input Debugging

- During attack:
  - Trace attack-path „by hand“
  - Contact administrator / ISP
  - Admin matches ingress port for a given packet pattern of egress port
  - Repeat until source is found…

- Disadvantages:
  - Cumbersome (what if admin X is not available?)
  - Slow
  - Expensive (manual intervention)
  - Not scalable

…Yet the most applied method until today…

Controlled Flooding

Karlsruhe Institute of Technology

- During Single Source DoS-Attacks, traversed backbone links on the attack path are (heavily) loaded
- Traceback attack path by testing links:
  - Measure incoming attack traffic
  - From victim to approximate source:
    - Create load on suspect links in the backbone
    - Measure difference in incoming attack traffic: if less attack packets arrive, the link is on the attack path…
- Need possibility to create load on links to test with access on end-hosts around the backbone (chargen-service on multiple foreign end-hosts)
- ☹ DoS of the backbone in itself
- Testing high speed backbone links using end-hosts difficult (how many dsl-links do you need to saturate one CISCO-12000-Link (10Gbps)?

[Burch & Cheswick: „Tracing Anonymous Packets to Their Approximate Source“]

KASTEL

ICMP Traceback

- *Assumption*:
  - DoS attacks are composed of packet floods
  - Traceback on probabilistic sample of traffic possible

- *Approach*:
  - Routers give destination information about path of packets
  - For 1 in 20k IP packets routers send additional ICMP iTrace to destination

- Information in the iTrace-Packet:
  - TTL → 255 (number of hops between router and destination)
  - Timestamp
  - Address of router
  - Ingress (previous hop) and Egress ports (next hop on path)
  - Copy of payload of traced packet (for identification)

[Bellovin: „ICMP Traceback Messages“]

ICMP Traceback: Open Issues

Karlsruhe Institute of Technology

- Signaling out of band → additional traffic (even at low rate)
- Large amount of packets needed to reconstruct the full attack path
  (Tradeoff: Amount of ICMP packets vs. speed of path detection)
- Victim needs to analyze large amount of iTrace messages
- Firewalls (often) drop ICMP messages
- Evasion: Possibility to create fake iTrace messages (easy to evade)
  (Potential solution: set up a PKI and let each router sign iTrace messages…)

KASTEL

Probabilistic Packet Marking (aka „IP Traceback“, PPM)

Karlsruhe Institute of Technology

Approach similar to ICMP Traceback:

- Mark forwarded packets with a very low probability
- In-band signaling to avoid additional bandwidth needs (mark packets directly)

- Different marking methods possible
- Different signaling (encoding) methods possible

[Savage et al.: „Network Support for IP Traceback“]

PPM Marking: Node Append

- Similar to IP Record Route: append each node‘s address to IP packet
- → Complete attack path in every received packet

Marking Procedure at router R:

```
For each packet w, append R to w
```

Path Reconstruction Procedure at victim v:

```
for any packet w from attacker
extract path (R1,..,Rj) from the suffix of w
```

- Pros and Cons:
  - Converges quickly, easy to implement
  - High bandwidth overhead (especially for small packets)
  - Possible additional fragmentation of IP packets

PPM Marking: Node Sampling (1)

Karlsruhe Institute of Technology

- Similar to ICMP Traceback, but use ***additional IP header field***

Marking Procedure at router R:

```
For each packet w, with probability p write R into w.node
```

Path Reconstruction Procedure at victim v with additional node table NodeTbl (node, count):

```
For each packet w from attacker, z ← w.node
if z in NodeTbl
        increment z.count
else
        insert (z,1) in NodeTbl
sort NodeTbl by count
extract path (R1,..,Rj) from ordered fields in NodeTbl
```

- Routers close to victim have higher probability of marking: the higher the count in NodeTbl the closer the router

PPM Marking: Node Sampling (2)

Karlsruhe Institute of Technology

- Issues of node sampling:
- Additional IP header field needed

- Routers far away from victim contribute only few samples (marks are overwritten) and large number of packets needed to recover complete path

  (p=0.51, d=15: > 42k packets needed to completely reconstruct attack path)

- In DDoS with multiple attackers different paths can not easily be distinguished

PPM Marking: Edge Sampling, Marking

- Mark packets with:
- Backbone edge *e (u,w)* (start router u, end router w) and distance *d(u,v)*
- Victim v can deduct graph of edges e and reconstruct attack tree

```
Marking Procedure at router R:
    For each packet w, with probability p
        write R into w.start and 0 into w.distance
    else // probability 1-p
        if w.distance = 0 then
                write R into w.end
        increment w.distance
```

PPM Marking: Edge Sampling, Reconstruction

- In order to reconstruct the attack tree

Path Reconstruction Procedure at victim v with additional attack tree t:

```
for each packet w from attacker
   if w.distance = 0 then
      insert edge (w.start, v, 0) into t
   else
      insert edge (w.start, w.end, w.distance) into t
remove all edges (x,y,d) with d ≠ d(x,v) in t
extract path (R1,..,Rj) enumerating acyclic paths in t
```

PPM Encoding

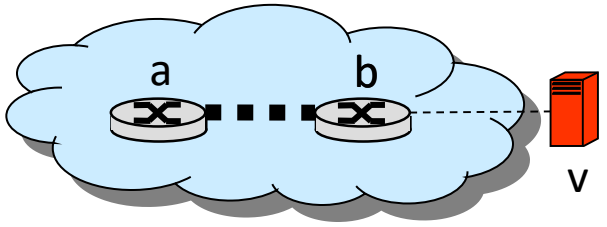
- With IP routers using IP addresses, marking of w.start, w.end, w.distance needs 32 + 32 + x bits.

- Solution: coding edge as IP(w.start) XOR IP(w.end)

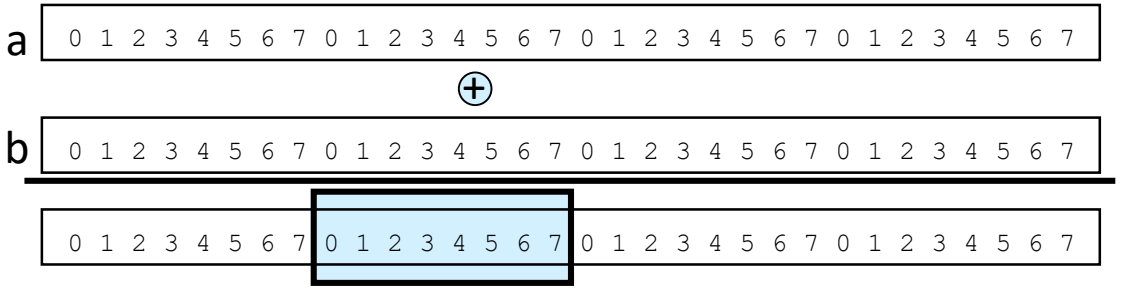
  (last hop known (w.distance = 0), others determined through XOR at victim)

  → 32 bit („edge-id“) + x bits (distance)

- Transmit only fragment of edge-ids with every packet and mark with higher probability *(together with hashed values of the router's edge IP address to distinguish edges → 64 bit per edge)*
  - Edge-ID fragment 8 bits, offset 3 bits, distance 5 bits → 16 bits

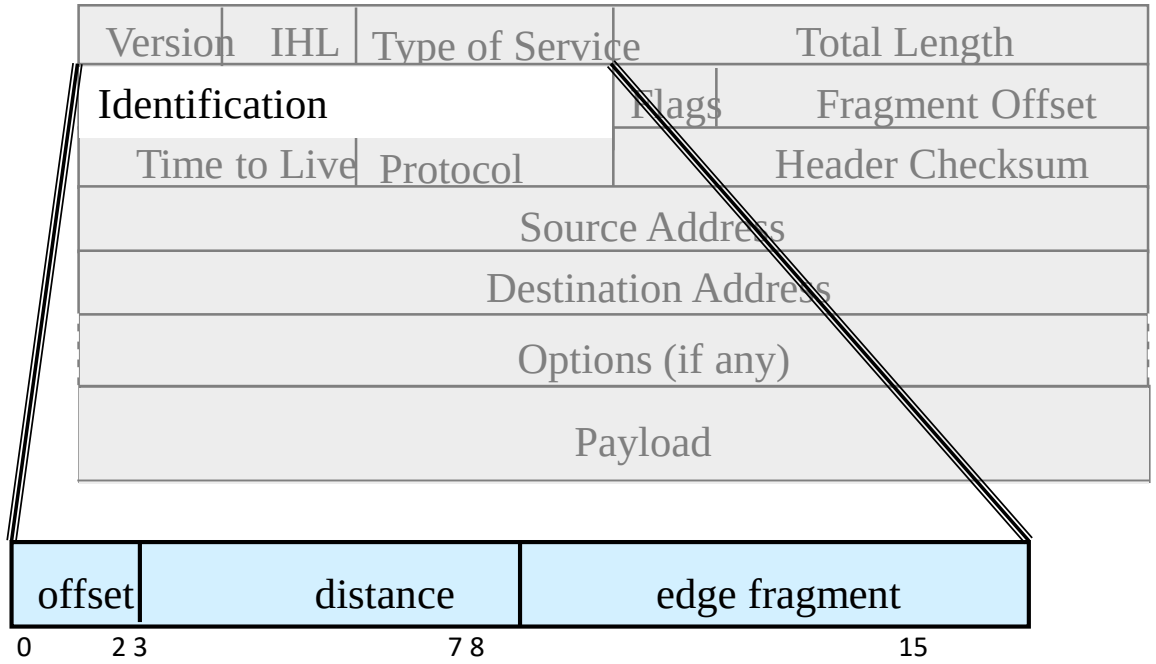


d(a,v) = 2



PPM Encoding: Encapsulation in IP header

- Using the „Identification“ field for in-band signaling (16 bit)



- *But the ID-Field is needed!?* In case of fragmentation:
  - Downstream marking: send ICMP Echo Reply („packet lost“)
  - Upstream marking: set „don‘t fragment“ flag

PPM Advantages and Disadvantages

Karlsruhe Institute of Technology

- ☺ Stable
- ☺ Meaningful results under partial deployment
- ☺ No bandwidth overhead
- ☺ Low processing overhead

- ☹ Works mainly for bandwidth exhaustion attacks
  - Many packets needed for reconstructing attack path
  - Fragmented packets can not be traced (e.g. Teardrop attack, however, Teardrop is not bandwidth exhaustion anyway)
- ☹ Victim under attack needs rather high amount of memory (many packets!) and processing time
- ☹ In order to avoid spoofing, authentication needed (PKI, signatures)

Related Techniques for Mitigation / Avoidance

Karlsruhe Institute of Technology

- Hop-Count Filtering
- Aggregate Based Congestion Control (ACC)
- Secure Overlay Services

KASTEL

Aggregate Based Congestion Control

Karlsruhe Institute of Technology

- Is it possible, to restrain attack traffic in the backbone?
  - Traffic is very diverse in the backbone, in general
  - However, attack traffic forms an aggregate of similar traffic

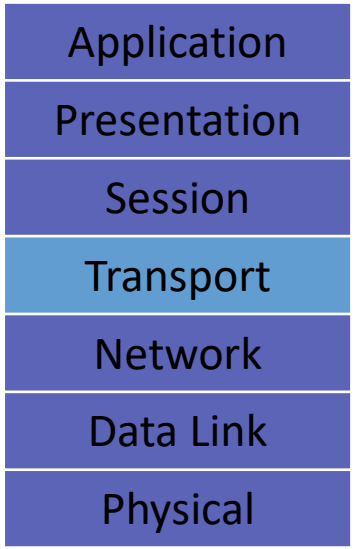
  (Identified by analyzing the dropped traffic:
  select the destination addresses with more than twice the mean number of drops and
  cluster these destination addresses to 24bit prefixes)

- ACC/pushback is a reactive approach:
  - If router/link is congested, can an aggregate be identified?
  - If there is an aggregate, limit the rate of aggregate traffic
  - If the aggregate persists, perform „pushback“: inform upstream routers to limit rate of the aggregate

[Mahajan, Bellovin & Floyd: „Controlling High Bandwidth Aggregates in the Network “]

Background: Transport Layer Security

- Transport layer provides end-to-end communication between application processes
- Main tasks
  - Isolation of higher protocol layers
  - Transparent transmission of user data
  - Global addressing of application processes
  - Overall goal: provisioning of an efficient and reliable service
- *Transport layer security protocols* aim on enhancing service of the transport layer by assuring additional security properties



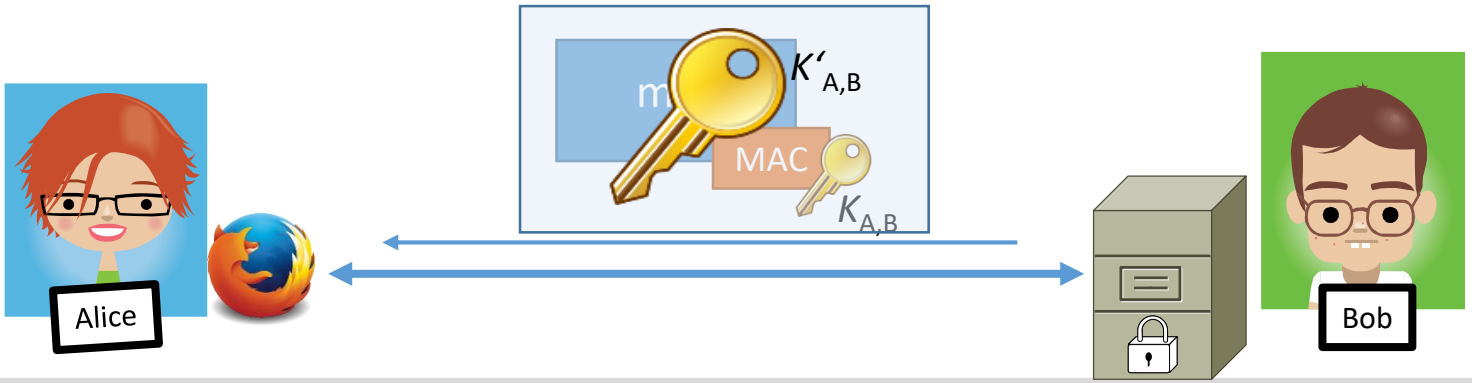
OSI-Model by layer

- Security protocols at transport layer: SSL, TLS, DTLS, SSH

- History
  - SSL was designed in the early 1990's to primarily protect HTTP sessions
  - In 1996 the IETF decided to specify a generic *Transport Layer Security (TLS)* protocol that is based on SSL

SSL/TLS Security Services

- *Peer entity authentication:*
  - Prior to any communications between client and server, authentication protocol is run to authenticate the peer entities
  - Upon successful completion of authentication dialogue *SSL session* is established
- *User data integrity:*
  - A MAC based on a cryptographic hash function is appended to user data
  - The MAC is computed with a negotiated secret in prefix-suffix mode
  - Either MD5 or SHA can be negotiated for MAC computation
- *User data confidentiality:*
  - If negotiated upon session establishment, user data is encrypted
  - Different encryption algorithms can be negotiated: RC4, DES, 3DES, IDEA

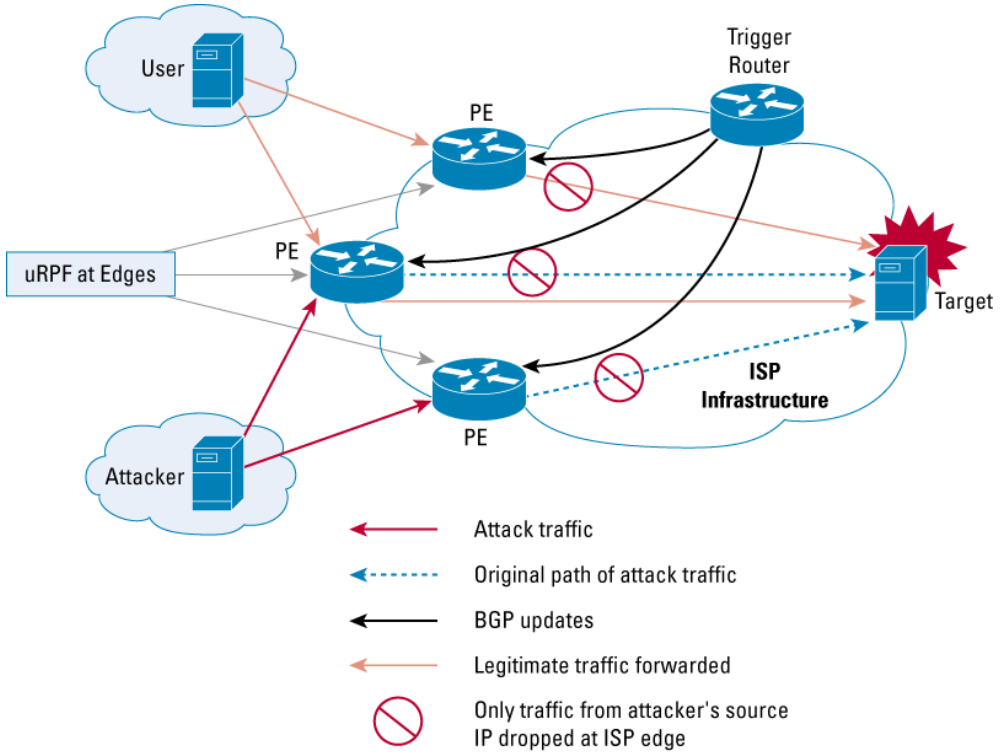


Remote-Triggered Black Hole Filtering (2) - S/RTBH

[CI05]

Source-Based Remotely Triggered Black Hole Filtering (S/RTBH)

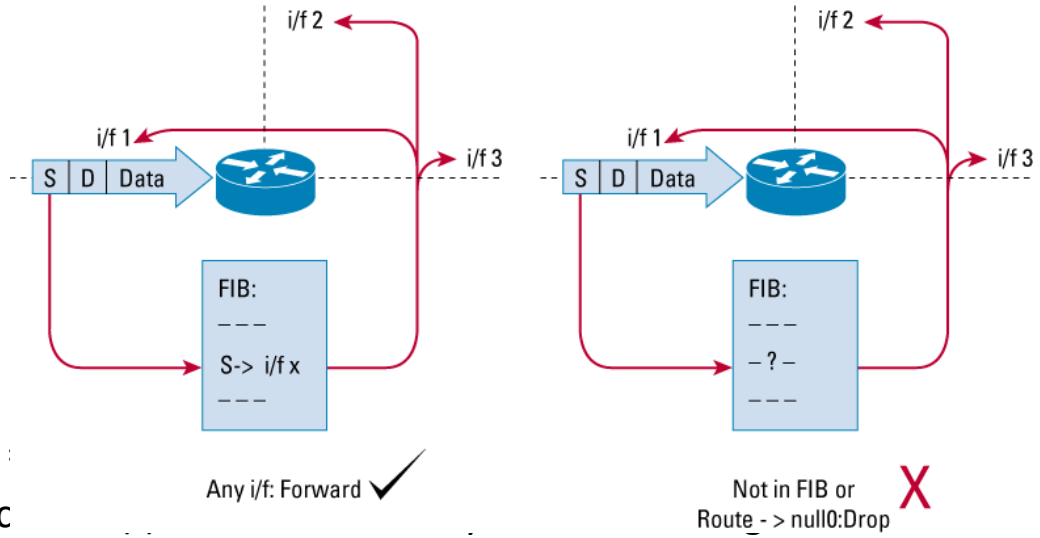
- Goal: Block all incoming traffic from a particular address (space)
  - Before traffic enters the target network, at BGP router level
  - Configure BGP-speaking routers to discard respective traffic that is not coming from the “expected” interface
  - Trigger router speaks iBGP (interior BGP) with border routers
  - Routers use Unicast Reverse Path Forwarding (uRPF)



Remote-Triggered Black Hole Filtering (3) - S/RTBH

[CI05]

- Leveraging Unicast Reverse Path Forwarding (uRPF) (RFC 5635)
  - Routers perform a route lookup of the source address upon packet reception
  - Loose Mode:
    - Requires: egress interface for route lookup exists in Forwarding Information Base (FIB) at all [or, != */dev/null*]
    - iBGP updates to explicitly invalidate routes to suspicious source addresses by setting their next hop to /dev/null (or null0)
  - Strict Mode:
    - Requires: ingress interface
    - (+) Might filter spoofed pac



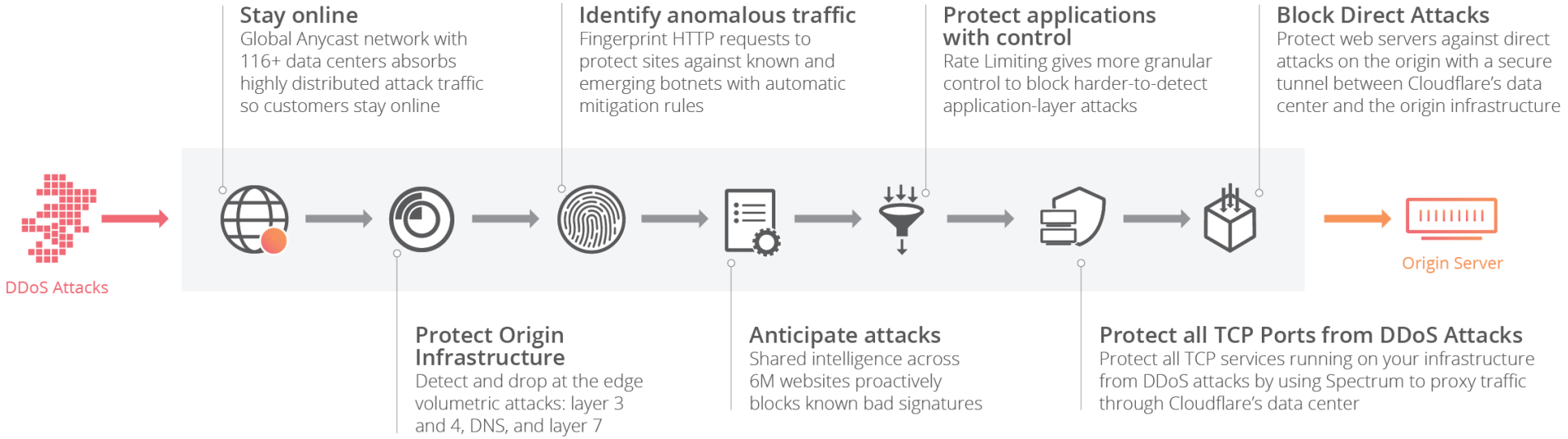
Recapitulation: Source Identification of IP Traffic

Karlsruhe Institute of Technology

- Problem: nodes may lie about their IP address
- Spoofing enables attackers to perform DoS/DDoS attacks
- If the source of an attack can be identified, attack traffic can be restrained
- Different approaches to identify attacker / routers / ISP on attack path:
  - Logging in the network
    - „Aggregated network logging“
    - Source Path Isolation („Hash-based IP Traceback“)
  - Traceback of packet flow
    - Controlled Flooding
    - ICMP Traceback
    - Probabilistic Packet Marking („IP Traceback“)
  - Other Means (Mitigation/Avoidance of attacks)

DDoS Mitigation in the Wild

- Business model: being a DDoS (/security) shield.
- Companies like Cloudflare or Imperva Incapsula
  - Content Delivery Networks
  - Operation of IDSs/IPSs and Firewalls



Source: https://www.cloudflare.com/

Some Upcoming Challenges

- The introduction of Internet protocols in classical and mobile telecommunication networks also introduces the Internet's DoS vulnerabilities to these networks

- Programmable end-devices (e.g., smartphones) may constitute a large base of possible slave nodes for DDoS attacks on mobile networks

- Software defined radio implementation may allow new attacking techniques:
  - Hacked smart phones answer to arbitrary paging requests
  - Unfair / malicious MAC protocol behavior
  - ...

- The ongoing integration of communications and automation may enable completely new DoS threats

Conclusion

Karlsruhe Institute of Technology

- Increasing dependence of modern information society on availability of communication services
- While some DoS attacking techniques can be encountered with “standard” methods, some can not:
  - Hacking, exploiting implementation weaknesses, etc. may be encountered with firewalls, testing, monitoring etc.
  - Malicious protocol deviation & resource depletion is harder to defend against
- Designing DoS-resistant protocols emerges as a crucial task for network engineering:
  - Network protocol functions and architecture will have to be (re-)designed with the general risk of DoS in mind
  - Base techniques: stateless protocol design, cryptographic measures like authentication, cookies, client puzzles, etc.

References (1)

[CSI00] Computer Security Institute and Federal Bureau of Investigation. *2000 CSI/FBI Computer Crime and Security Survey.* Computer Security Institute Publication, March 2000.

[Akamai16] Akamai. (2016). akamai's [state of the internet] Q1 2016 report, 77. https://doi.org/10.1017/CBO9781107415324.004

[Dar00] T. Darmohray, R. Oliver. *Hot Spares For DoS Attacks.* ;login:, 25(7), July 2000.

[JuBr99] A. Juels und J. Brainard. *Client Puzzles: A Cryptographic Countermeasure Against Connection Depletion Attacks.* In Proceedings of the 1999 Network and Distributed System Security Symposium (NDSS'99), Internet Society, March 1999.

[Mea00] C. Meadows. *A Cost-Based Framework for the Analysis of Denial of Service in Networks.* 2000.

[MVS01] D. Moore, G. M. Voelker, S. Savage. *Inferring Internet Denial-of-Service Activity.* University of California, San Diago, USA, 2001.

[NN01] S. Northcutt, J. Novak. *Network Intrusion Detection - An Analyst's Handbook.* second edition, New Riders, 2001.

[TL00] P. Nikander, T. Aura, J. Leiwo. *Towards Network Denial of Service Resistant Protocols.* In Proceedings of the 15th International Information Security Conference (IFIP/SEC 2000) Beijing, China, 2000.

[BA03] A. Belenky, N. Ansari:"On IP Traceback", in IEEE Communications Magazine, July 2003

[BC00] Burch & Cheswick: „Tracing Anonymous Packets to Their Approximate Source", Proceedings of the 14th USENIX conference on System administration, 2000

[Bel01] Bellovin: „ICMP Traceback Messages", Internet-Draft draft-ietf-itrace-01.txt, 2001

References (2)

[JWS03] Jing & Wang & Shin: „Hop-Count Filtering: An Effective Defense Against Spoofed DDoS Traffic“, Proceedings of the 10th ACM conference on Computer and communications security, 2003

[KMR02] Keromyits & Misra & Rubenstein: „SOS: Secure Overlay Services“, Proceedings of ACM SIGCOMM, 2002

[MBF01] Mahajan & Bellovin & Floyd: „Controlling High Bandwidth Aggregates in the Network“, Technical report, 2001

[RSG98] Reed, Syverson & Goldschlag: „Anonymous Connections and Onion Routing“, IEEE Journal on Selected Areas in Communications, 1998

[Sav01] Savage et al.: „Network Support for IP Traceback“, IEEE/ACM Transactions on Networking (TON), 2001

[Sto00] Stone: „Centertrack: An IP Overlay Network for Tracking DoS Floods“, Proceedings of 9th USENIX Security Symposium, 2000.

[Sno02] Snoeren et al.: „Single-Packet IP-Traceback“, IEEE/ACM Transactions on Networking (TON), 2002

[Ros14] Rossow, Christian. "Amplification Hell: Revisiting Network Protocols for DDoS Abuse." NDSS. 2014.

[JiWa+] Cheng Jing, Haining Wang, Kang G. Shin: „Hop-Count Filtering: An Effective Defense Against Spoofed DDoS Traffic“, CCS, 2003

c Cisco “Remotely triggered black hole filtering- destination based and source based” , Whitepaper, https://www.cisco.com/c/dam/en_us/about/security/intelligence/blackhole.pdf

Source: https://www.cisco.com/c/dam/en_us/about/security/intelligence/blackhole.pdf