

Denial of Service attacks

Chapter 21.4: DDoS attacks



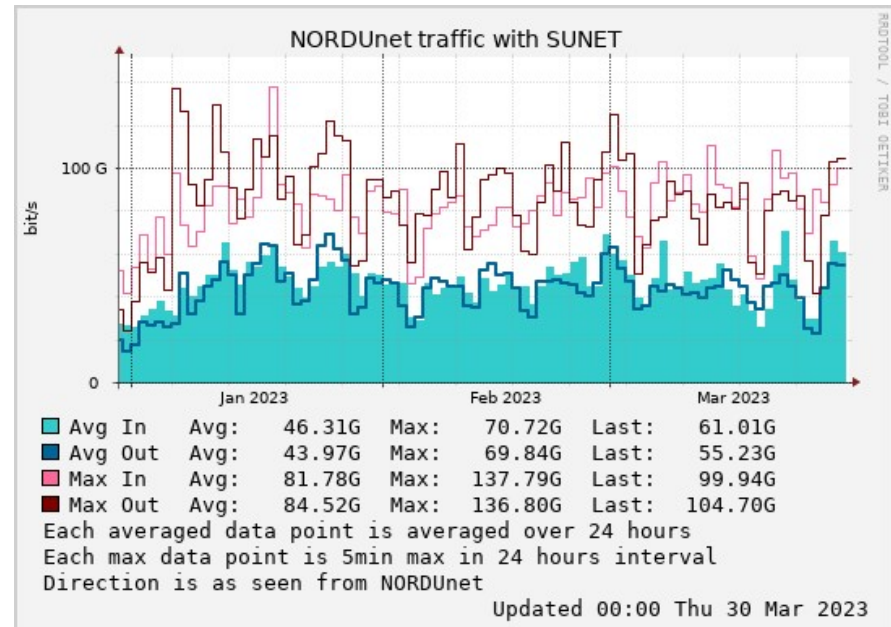
The reading material for TCP also contains DoS attacks, read it!

SUNET – Swedish University Network



SUNET: >8,000 km fiber, 100 Gbps redundant links

Last three months, average and peak traffic



<https://stats.nordu.net/stat-q/r-all?q=all&name=SUNET>

Different DoS attacks

DoS attack: when an attacker attempts to prevent legitimate users from accessing a computer resource, normally by overwhelming it with malicious traffic

1. Bandwidth exhaustion

- Victim and its network drowns in data
- Example: Reflection, flooding and magnification attacks (Smurf and Fraggle)

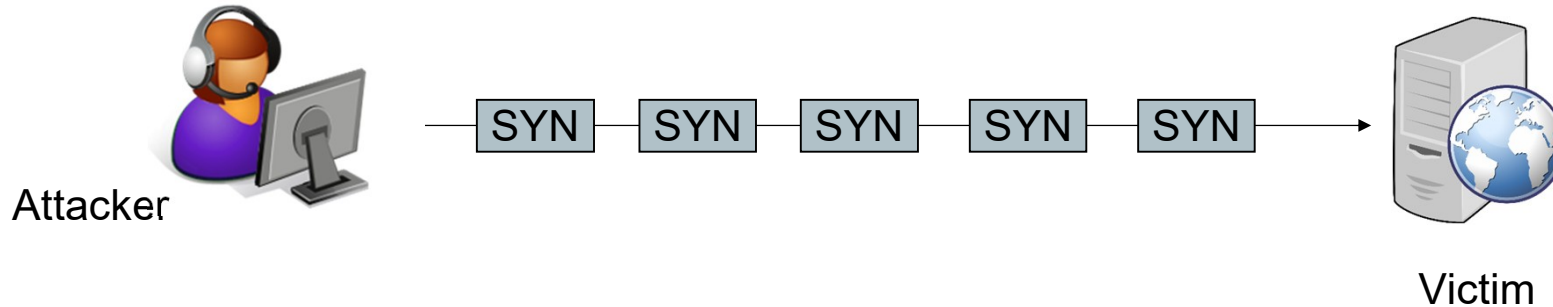
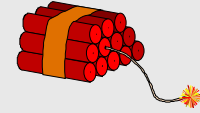
2. Resource exhaustion

- Example: SYN Flood DoS Attack – victim's internal resources are exhausted
- Example: Make victim calculate new crypto-keys at high rate, for example a web server
- Possible solution: Make it more expensive for the attacker than for the victim to connect

3. Trigger implementation bugs


- Idea: send unusual input to applications that developers did not test
- Make internal tables overflow, create unusual situations for software, send unexpected input, ...
- Examples: Ping-of-Death, Teardrop, etc.
- If one packet crashes a system – when it is back, a new packet makes it crash again
- Even firewalls can crash

TCP: SYN flood attack

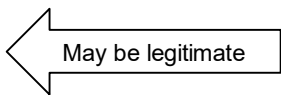


- Attacker Sends Flood of SYN segments
- Victim allocates resources, sends SYN/ACK and waits for ACK
- Many systems have a table (with fixed size) for each port , a “backlog”
 - Linux and Solaris: 1024 (configurable)
 - Windows has dynamic backlogs (20 – 20,000)
- When victim’s connection table is full: can not respond to new connections
 - Retransmits SYN/ACK after 3, 6, 12, 24, 48 seconds and then timeout after 96 seconds → it takes 189 seconds until connection dropped
 - Just send connection requests faster than victim times out to keep table full

SYN attacks are easy to find – if we look



```
# netstat -n
tcp 0 0 10.100.0.200:21 237.177.154.8:25882 SYN_RECV
tcp 0 0 10.100.0.200:21 236.15.133.204:2577 SYN_RECV
tcp 0 0 10.100.0.200:21 127.160.6.129:51748 SYN_RECV
tcp 0 0 10.100.0.200:21 230.220.13.25:47393 SYN_RECV
tcp 0 0 10.100.0.200:21 227.200.204.182:60427 SYN_RECV
tcp 0 0 10.100.0.200:21 232.115.18.38:278 SYN_RECV
tcp 0 0 10.100.0.200:21 229.116.95.96:5122 SYN_RECV
tcp 0 0 10.100.0.200:21 236.219.139.207:49162 SYN_RECV
tcp 0 0 10.100.0.200:21 238.100.72.228:37899 SYN_RECV
...
```



OS protection against SYN flooding

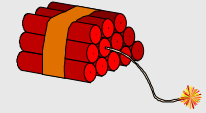
- Start **freeing pending connections** if needed
 - Decrease timeout values (max 10 seconds ok)
 - Decrease number of retransmits of SYN/ACK (0 or 1 ok)
 - Increase backlog table size (from 1,000 to 65,000 ?)
 - Drop older entries – round robin
 - Random drop – increases chance that attacker's own packets are dropped
 - Limited effect if attacker has good bandwidth
- Allocate “**micro records**” for each connection request
 - Typically 16 bytes: just record source and dest. IP addr, ports, ISN, options
 - 100,000 pending connections requires less than 2 MByte
 - If internal data structure is dynamic, queue may never fill [Windows]
- Send necessary state information to client instead of storing it
 - **SYN cookies** [Linux, Unix]

SYN cookies [RFC 4987]

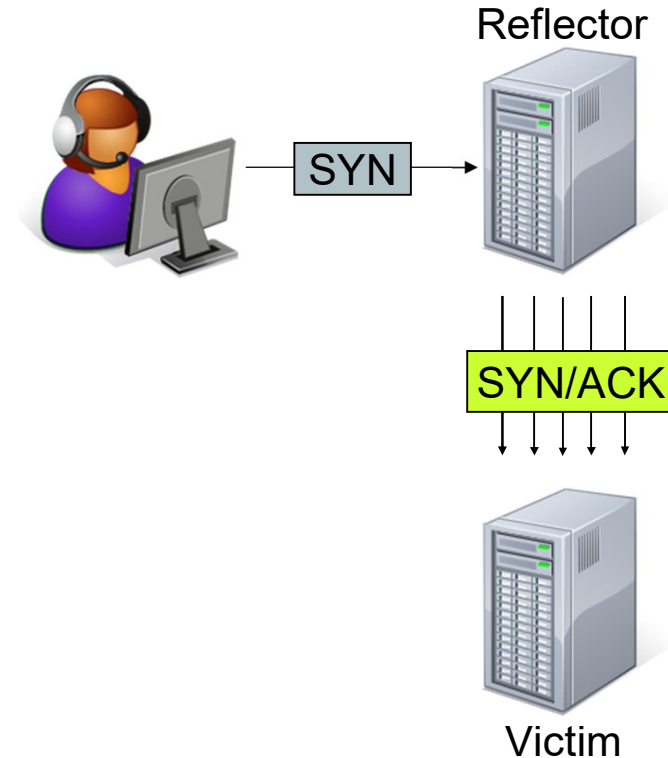
- Some systems (Solaris, Linux, FreeBSD, MacOS, ...) can be configured to use a different SYN handling mechanism when under attack
- **ISN = hash (timestamp, src_IP, src_port, dest_IP, dest_port, client ISN, secret)**
 - First five bits in ISN is “time MOD 32” to satisfy TCP RFC
 - A timestamp is part of the hash (minute resolution)
 - Exact implementation may differ somewhat between systems
- No data saved internally when SYN is received
 - The initial sequence number (ISN) in reply contains all state information needed
 - When ACK is received, hash is checked and data structures are allocated
- Complications:
 - Cannot send TCP options in initial SYN packet (e.g. window scaling and SACK)
 - Probably still the best alternative!
- A stateless firewall will have problems with this
 - SYNs may be blocked, but ACKs where attackers guess cookies will be forwarded
 - Solution: only enable SYN cookies for ports that are flooded with SYNs
- **A stateful firewall can also be vulnerable to SYN flooding**
 - They also need to keep state
 - Good firewalls can protect systems against SYN floods like this
 - They may answer on behalf of the real server but must then translate all ISNs (act as proxy)



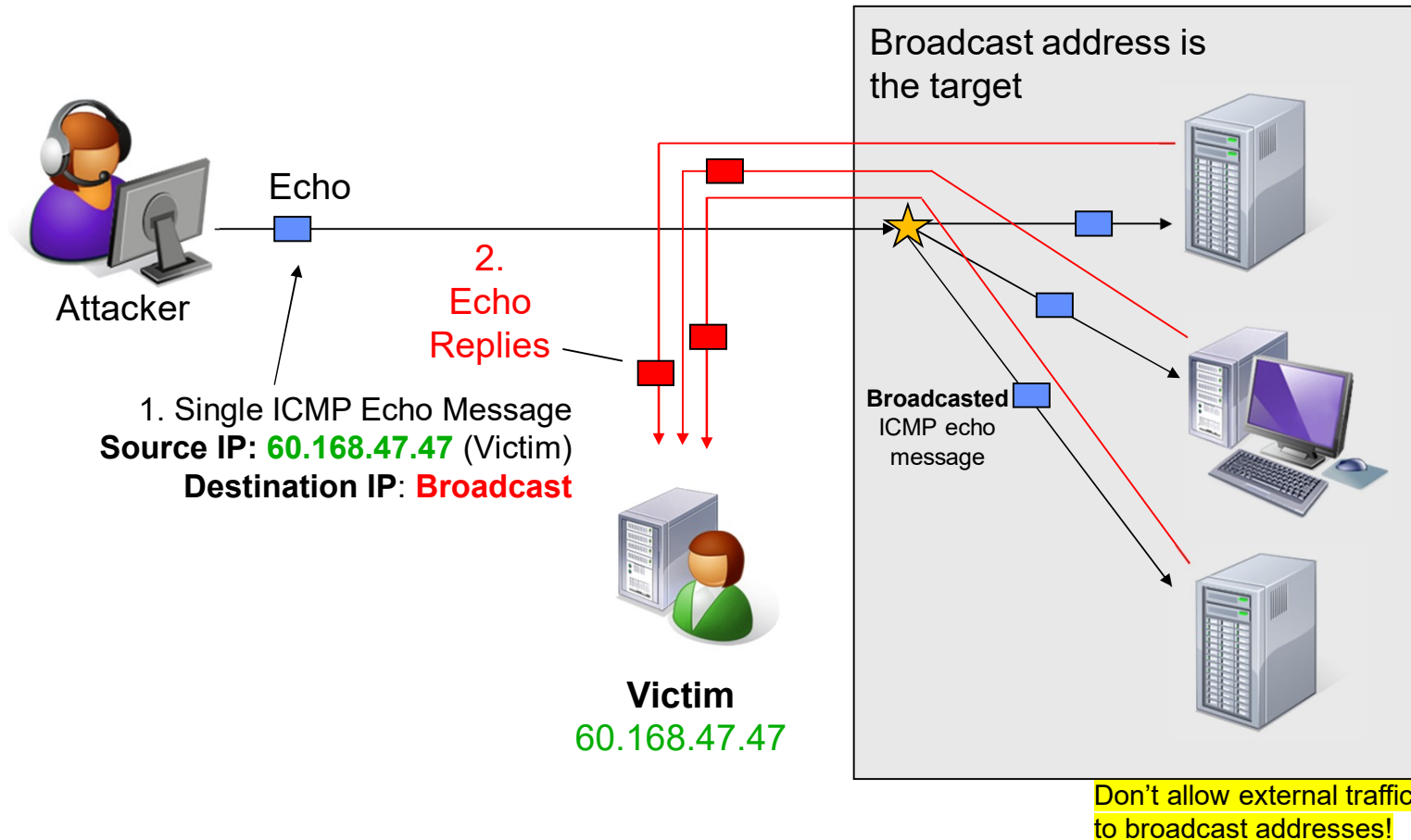
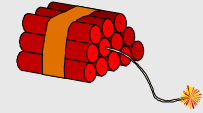
TCP magnification attacks



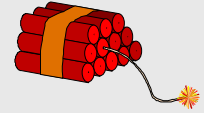
- Send SYN requests to reflectors (systems) with victim as the source address
- SYN/ACK will be sent – to victim
 - Reflector will retransmit SYN/ACKs typically 5 times if it gets no reply from victim
- Real attacker's address is hidden
 - Victim must go to the reflector system to find the real source – and source address may be faked
- Timeout and number of retransmissions can be configured in most systems (can even be zero)
- Even routers and important systems can be used to flood victims with traffic
 - The SYN/ACK replies may come from legitimate systems
 - Therefore not possible to block traffic from these systems in firewalls – file server, DNS server, ...



The Smurf Magnification Attack



Magnification attacks (Smurf & Fraggle)



- **Smurf** uses ping (ICMP echo) packets to generate a flood of ICMP echo replies
 - Uses a broadcast address as destination with the victim as the source
 - Potentially all hosts on the subnet will talk to it (hundreds...)
- Lots of low bandwidth hosts can fill high bandwidth networks
- Other services (e.g. UDP-based) can also be used
 - When UDP is used is the attack called **Fraggle**
 - Example of UDP services (ports to send to): echo, chargen, NTP, ...
 - Two systems may even start talking to each other (“ping-pong” effect)
 - DNS reflector flood: DNS reply to victim can be much larger than request
- Countermeasures:
 - Stop spoofed addresses before they reach the inside (hard?)
 - Don’t allow broadcast as destination for outside traffic (ingress filtering)
 - Hosts should avoid responding to broadcast packets – exceptions necessary



DNS Magnification Attacks

- Popular recent years
- Send DNS request with **faked victim addresses**
 - Some DNS servers can reply with lots of info (whole zone)
 - Badly configured, buggy implementations, etc. may also be used
- Hard to filter – where/who should filter it?

This is a general problem – Internet services responding to UDP requests are often vulnerable to IP address spoofing (DNS, memcached, ...)

Can firewalls filter attacks?



- **Can filter:**

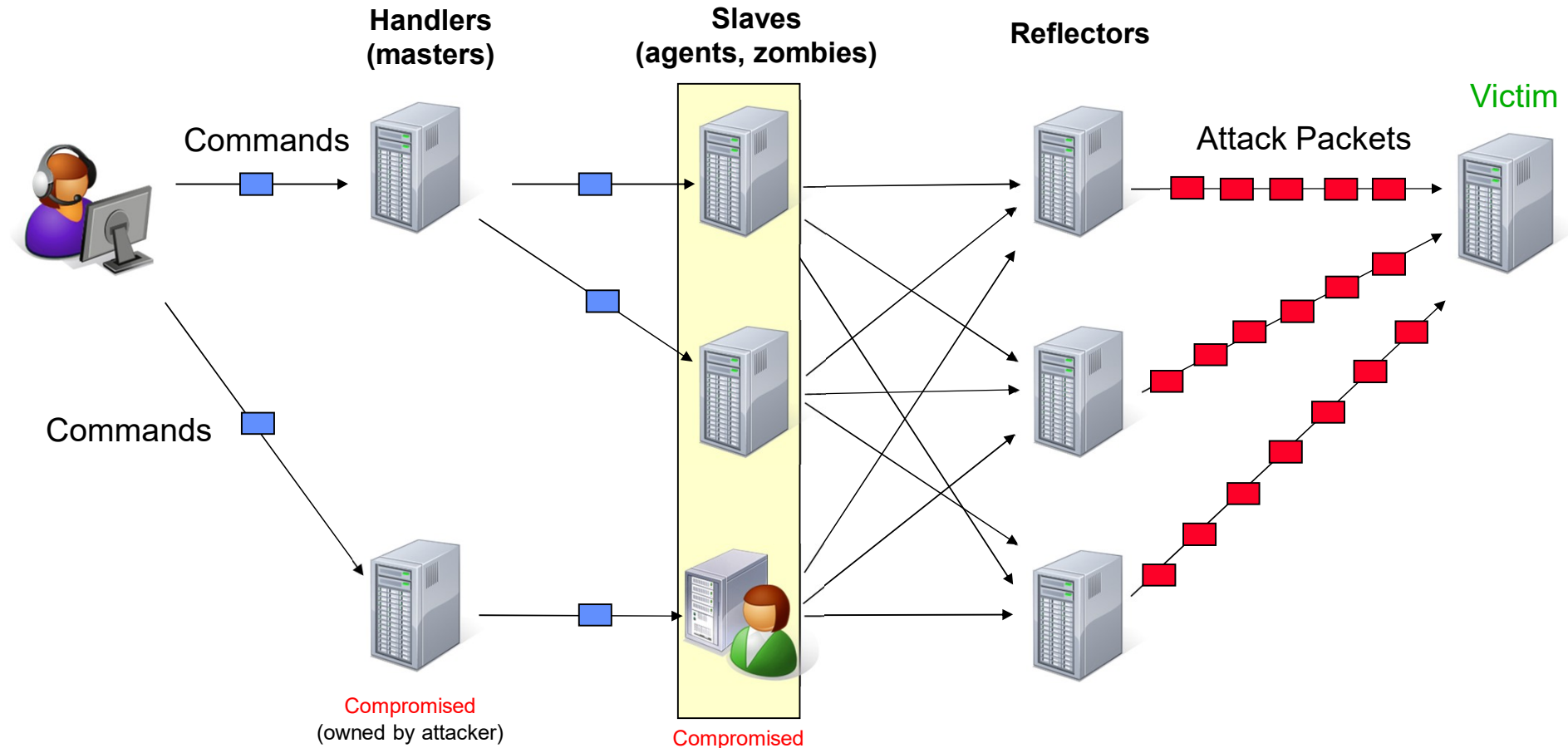
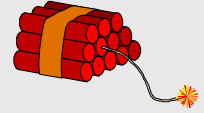
- Malformed and obviously illegal packets
- Traffic not needed by target: UDP traffic against a web server, traffic to closed ports, etc.

- **Can not filter:**

- Well formed packets that request legitimate/critical services
- Very hard to distinguish attack packets from legitimate service requests, such as http flooding of a web server
- HTTP traffic waiting for more data to come – but that never comes, i.e. consuming resources

DDoS – Distributed DoS attacks

DDoS Attacks through Handlers and Zombies



Early DDoS attacks

- **Trin00** [first well-known – 1999]
 - Made University of Minnesota without network access for almost 3 days
 - Communicates with **TCP/27665 with handlers** and **UDP/27444 with agents (zombies)**
 - **Zombies flood victim with 1000-byte UDP packets** to random UDP ports
- **TFN, Tribe flood network** [2000]
 - **Communication via ICMP Echo and Echo Reply messages**
 - They normally do not contain data, therefore ignored by logging software
 - **Spawns ICMP flood, SYN flood, UDP flood, and Smurf-style attacks**
 - Attacks against CNN.com, eBay, Yahoo.com, Amazon.com, Dell.com, eTrade
 - **Damages > \$1,000,000,000**
 - **FBI found a 15 year old script kiddie from Montreal**
 - 8 months in jail, wrote a book about it 2008

Botnets

- The zombies are often called “bots” from (ro)bots, remotely controlled zombies
- Bots are sold on the Internet – pay for number of hosts and time used
 - 2005: A bot network with 1,500,000 computers was found in the Netherlands
 - 2008: Canadian police have arrested 17 people suspected of running a botnet of a million computers in 100 different countries
 - 2011: The TDL Botnet infected more than 4,500,000 machines, 100,000 new IP addresses per day
 - IoT opens new doors...
- The attacker does not need to know where all bots are located
 - They will automatically connect (call home)
- IRC channels and p2p networks popular to instruct zombies →
 - They use legitimate communication ports (may be allowed through firewalls)
 - Firewalls have problems knowing that communication is not legitimate

ana maria wrote:

```
E1OWJEFF7189829F9C8064C38FA28E9DC388C3C0211230  
8389D37EBD67DE1E4AA088ECDC63F648D8350D4850E179  
488EC270DA73D30022DF589A23EA6AF64257995088D90C  
D768447C44CD05F86218529E1131AC07FA343ED1EC26EA  
22217E89E027F9181EE767ED6D8362E73818AD198867A52
```

Source: RSA FraudAction Research Labs

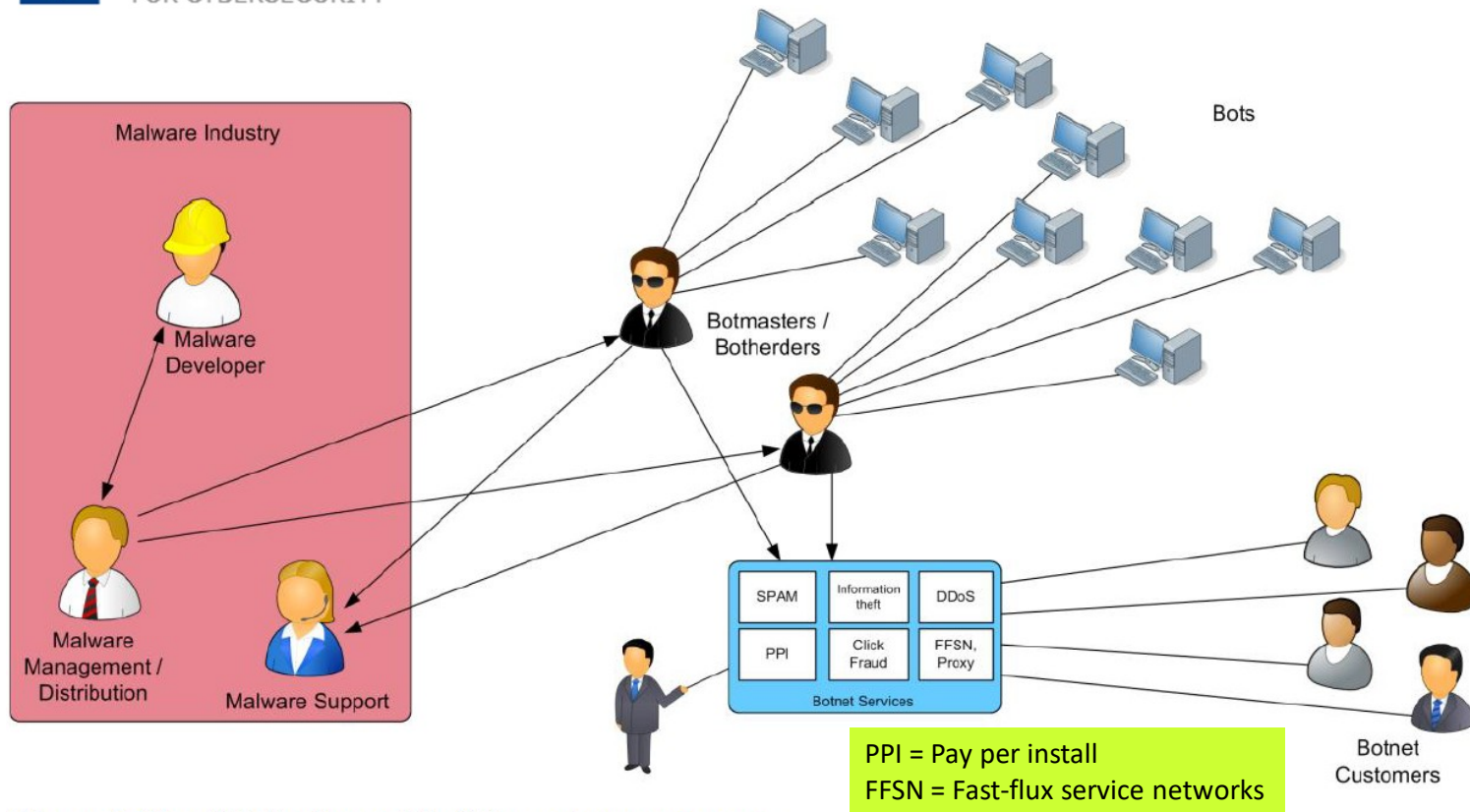
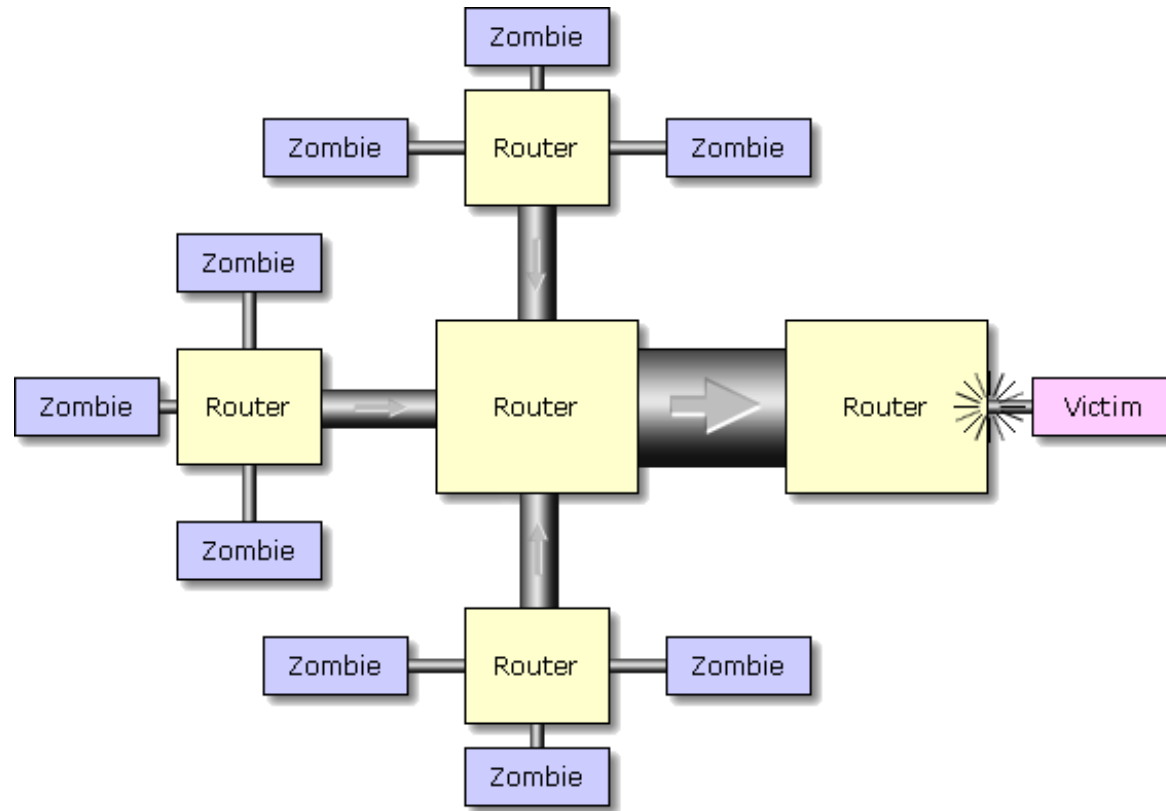


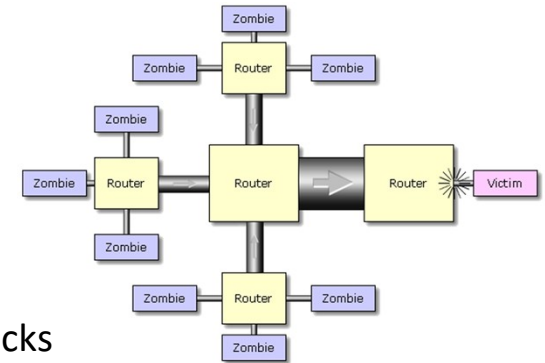
Figure 5: Simplified role model of the malware economy.

The Difficulty of Stopping DDoS Attacks



Detection of DDoS Attacks (academic)

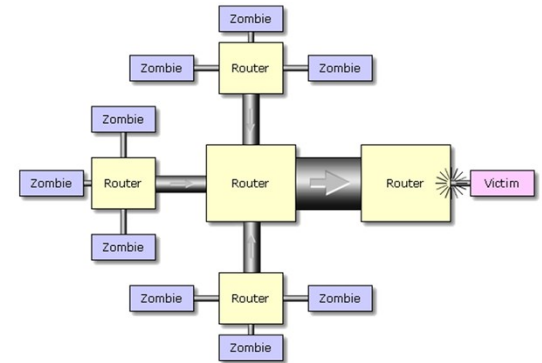
- Two main groups: **host-based** and **network based**
- Proposed network based detection mechanisms:
 - **MULTOPS**: packet rate between two hosts are proportional during normal operation
 - **SYN-rate detection**: Deviation of the SYN and FIN packet ratios indicate attacks
 - **Time-series analysis**: Build profiles of attacks and normal behavior
- These mechanisms can be distributed over the Internet and stop attacks close to the source
- Often possible for attacker to circumvent detection
 - E.g. randomize source addresses, ...
- Early detection and network defense mechanisms reduce damages



Tracing the source in DDoS attacks (proposed)

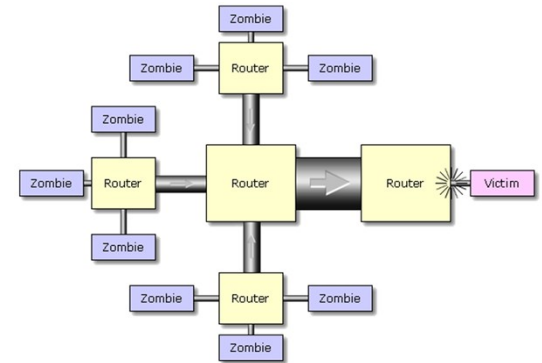
Research is going on in this area:

- OSPF
 - Use the existing routing protocol (OSPF) to tell routers to not forward datagrams to the victim – very coarse, may not work in reality
- Traceback
 - For each datagram forwarded by a router, the receiver may, with some small probability, also receive additional info about who forwarded it
- Pushback
 - Routers realizing they have to discard many datagrams to one destination can tell other routers to drop these datagrams as well
- Centertrack
 - Routing within a system (e.g. ISP) may be temporarily handled by special routers that are configured to handle tracing if an attack is seen



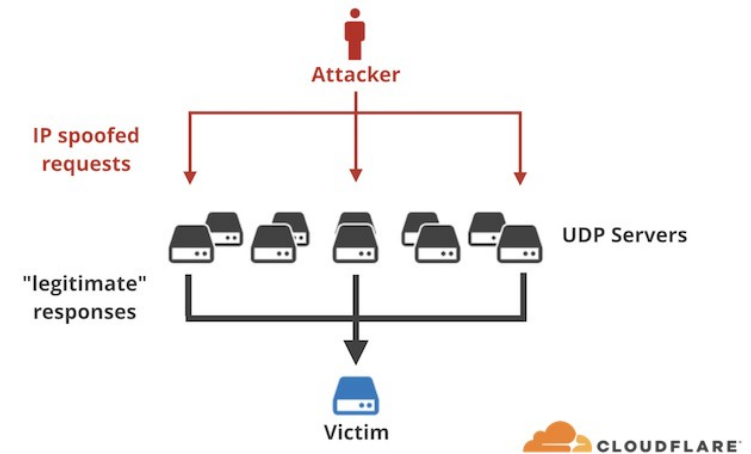
Defense against DDoS Attacks

- Requires help from ISPs; victim cannot do it alone
 - Cloud services are harder to attack – and can be scaled up if necessary
- **Ingress filtering** by ISP to stop attack packets
 - Even link to local ISPs might become overloaded
 - Zombie addresses may be too many to be filtered or stopped
- **Egress filtering** by companies and ISPs (will also detect own zombies)
- **DNS sinkholes** can be created
 - **Remove path to controllers** → packets are sent to black holes
 - Internet-wide sinkholes have been created. Small-scale sinkholes can also work
- Some ISPs that specialize in DDoS protection for customers
 - Can take over customer's firewall rules (drop all but ... at an earlier point)
 - Can allow all traffic from some prioritized networks (from selected IP addresses)
 - Traffic to customer can be re-routed to a special router/firewall directly at their incoming link
- Products from Cisco, Checkpoint, Akamai, Juniper, etc. claim to handle DDoS traffic



The memcached attack

- Github attacked Feb 2018 – **peak 1,35 Tbit/s**
- Memcached servers are used to remember/cache old database queries and replies
 - Uses UDP
 - Compares a key (a “string”) and remembers its “reply”
 - Uses UDP port 11211
- Servers not intended to be exposed to the Internet
- **Amplification factor 50,000 times !**
 - Observed 15 byte requests → 750 kByte to the victim
 - 95,000 memcached servers available worldwide...





LILY HAY NEWMAN SECURITY 03.01.18 11:01 AM

GITHUB SURVIVED THE BIGGEST DDOS ATTACK EVER RECORDED

ON WEDNESDAY, AT about 12:15 pm EST, 1.35 terabits per second of traffic hit the developer platform GitHub all at once. It was the most powerful distributed denial of service attack recorded to date—and it used an increasingly popular DDoS method, no botnet required.

The memcached DDoS attack

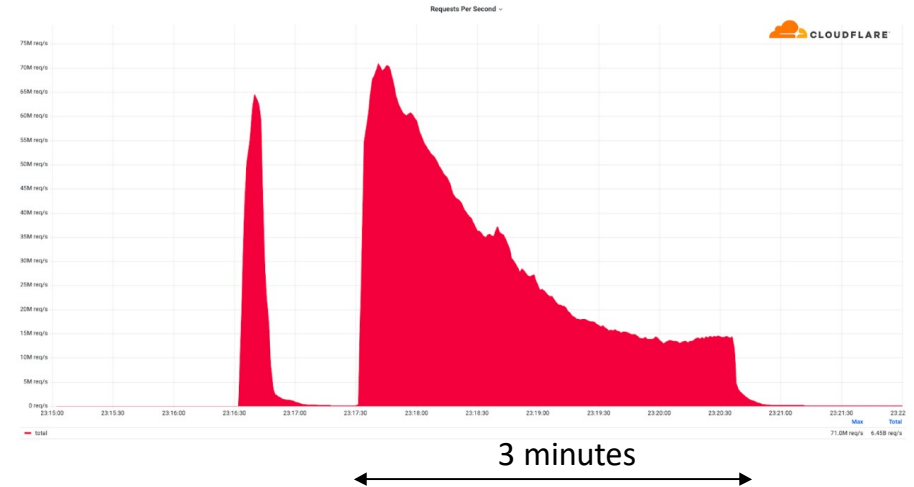
GitHub briefly struggled with intermittent outages as a digital system assessed the situation. Within 10 minutes it had automatically called for help from its DDoS mitigation service, Akamai Prolexic. Prolexic took over as an intermediary, routing all the traffic coming into and out of GitHub, and sent the data through its scrubbing centers to weed out and block malicious packets. After eight minutes, attackers relented and the assault dropped off.

The attackers also may have been hoping to extract a ransom. "The duration of this attack was fairly short," he says. "I think it didn't have any impact so they just said that's not worth our time anymore."



Cloudflare mitigates record-breaking 71 million request-per-second DDoS attack

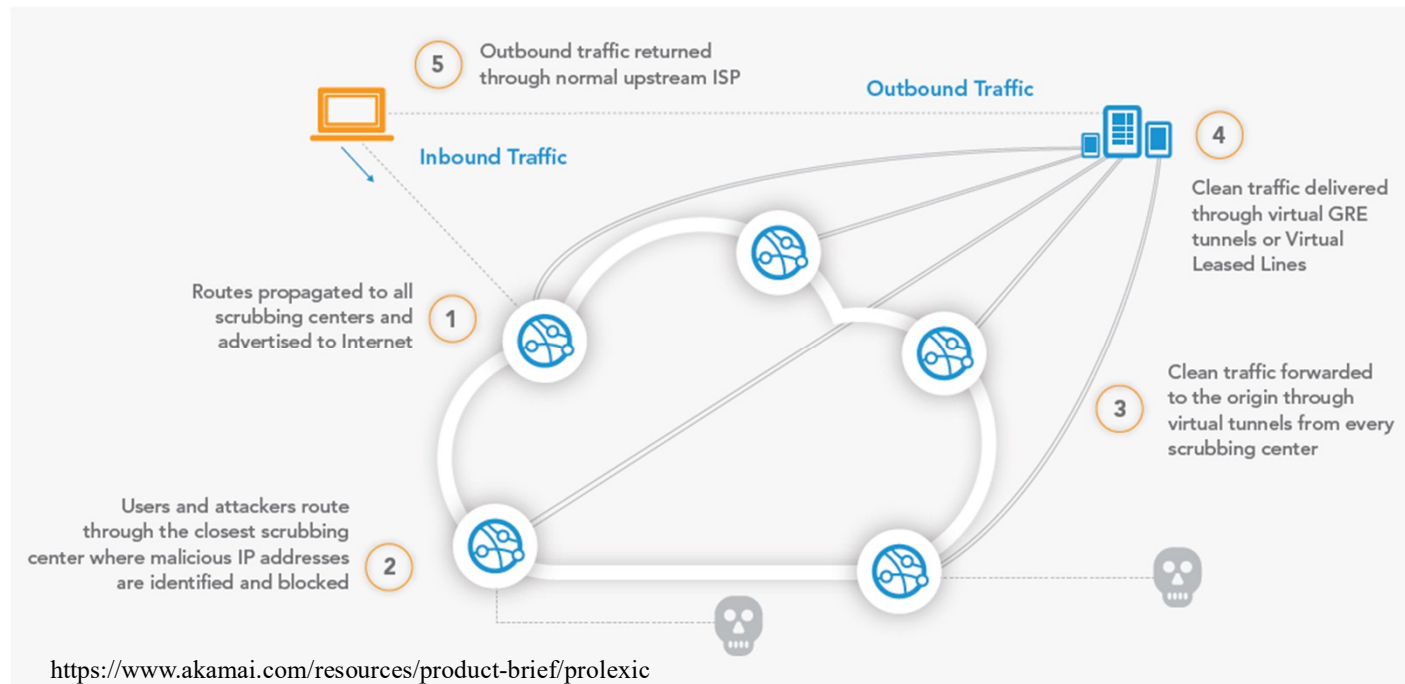
- Record-breaking attack 2023-02-13
- 50-70 million requests per second (rps)
 - 54% larger than previous record from 2022
- HTTP traffic to servers (protected by Cloudflare)
 - Originated from 30,000 different IP addresses
- *“The attacks originated from numerous cloud providers, and we have been working with them to crack down on the botnet”*



Akamai promises 100% DDoS protection...



- Uses 365,000 servers at 1,350 networks in 135 countries (2022)
- Legitimate traffic always close to scrubbing servers – Prolexic takes over
- Total edge capacity 200+ Tbps
- DoS attacks can be stopped close to source



These numbers are part of their marketing strategy, but could a small operator or actor compete with them and mitigate large attacks?

Akamai – DDoS attack in Europe 2022

- On Thursday, July 21, 2022, Akamai detected and mitigated the largest DDoS attack ever launched against a European customer on the Prolexic platform
- attack traffic peaking at 853.7 Gbps and 659.6 Mpps (packets per second) over 14 hours
- attacks consisting of UDP, UDP fragmentation, ICMP flood, RESET flood, SYN flood, TCP anomaly, TCP fragment, PSH ACK flood, FIN push flood, and PUSH flood, among others
- Distributed attack traffic suggests bad actors were leveraging a highly-sophisticated, global botnet of compromised devices to orchestrate this campaign
- No individual scrubbing center handled more than 100 Gbps of the overall attack

Summary, Types of DoS attacks

