Network Security: Denial of Service (DoS)

Tuomas Aura / Aapo Kalliola

T-110.5241 Network security
Aalto University, Nov-Dec 2012

Outline

1. DoS principles
2. Packet-flooding attacks on the Internet
3. Distributed denial of service (DDoS)
4. Filtering defenses
5. Most effective attack strategies
6. Infrastructural defenses
7. DoS-resistant protocol design
8. Research example

DoS principles

Denial of service (DoS)

- Goal of denial-of-service (DoS) attacks is to prevent authorized users from accessing a resource, or to reduce the quality of service (QoS) that authorized users receive
- Several kinds of DoS attacks:
  - Destroy the resource
  - Disable the resource with misconfiguration or by inducing an invalid state
  - Exhaust the resource or reduce its capacity

Resource destruction or disabling

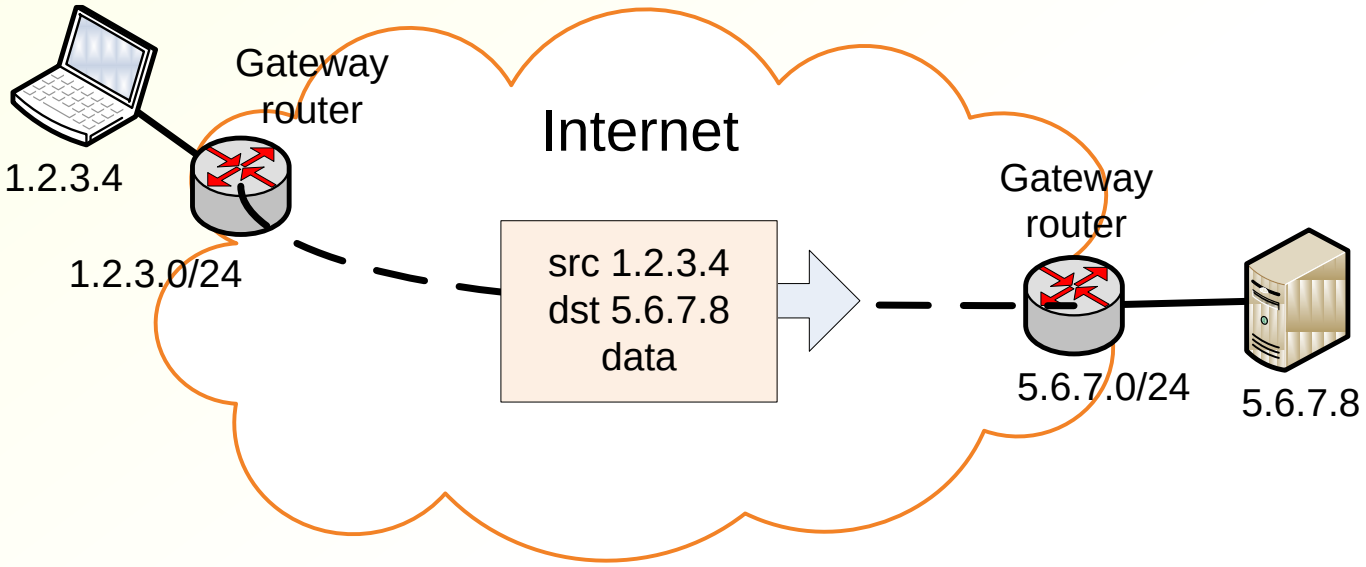
- Examples:
  - Cutting cables, bombing telephone exchanges
  - Formatting the hard disk
  - Crashing a gateway router
- These attacks often exploit a software bug, e.g.
  - Unchecked buffer overflows
  - Teardrop attack: overlapping large IP fragments caused Windows and Linux crashes
- Can be prevented by proper design and implementation

Resource exhaustion attacks

- Attacker overloads a system to exhaust its capacity
  → never possible to prevent completely in an open network
- Examples:
  - Flooding a web server with requests
  - Filling the mailbox with spam
- It is difficult to tell the difference between attack and legitimate overload (e.g. Slashdotting, flash crowds)
  - For highly scalable services, need to try to detect attacks
- Some resource in the system under attack becomes a bottleneck i.e. runs out first → Attacks can exploit a limited bottleneck resource:
  - SYN flooding and fixed-size kernel tables
  - Public-key cryptography on slow processors
  - Apache “range” header request bug

Packet-flooding attacks on the Internet

Internet characteristics



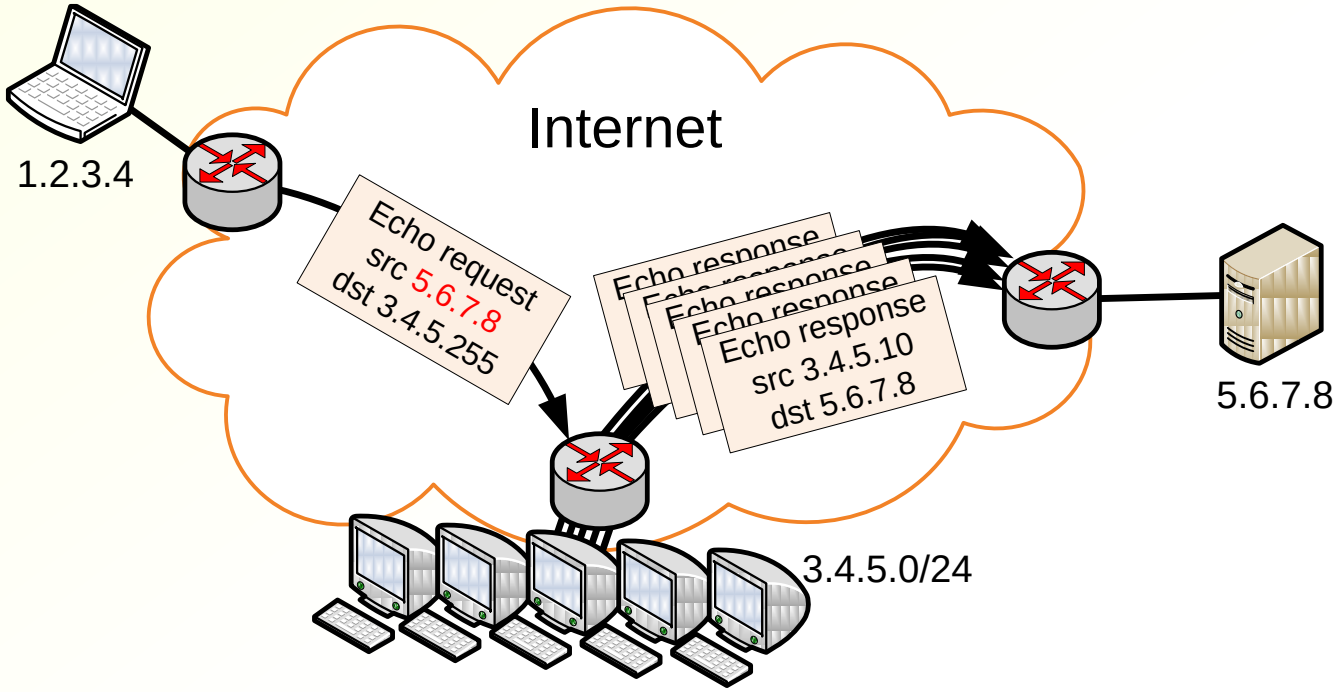
- Q: Why is the Internet vulnerable to DoS?
  - Open network: anyone can join, no central control
  - End to end connectivity: anyone can send packets to anyone
  - No global authentication or accountability
  - Flat-rate charging (mostly)
  - Unreliable best-effort routing; congestion causes packet loss
- Q: Could these be changed?

Packet-flooding attack

- Ping flooding: attacker sends a flood of ping packets (ICMP echo request) to the target
  - Unix command *ping -f* can be used to send the packets
- Any IP packets can be used similarly for flooding
- Packets can be sent with a spoofed source IP address
- Q: Where is the bottleneck resource that fails first?

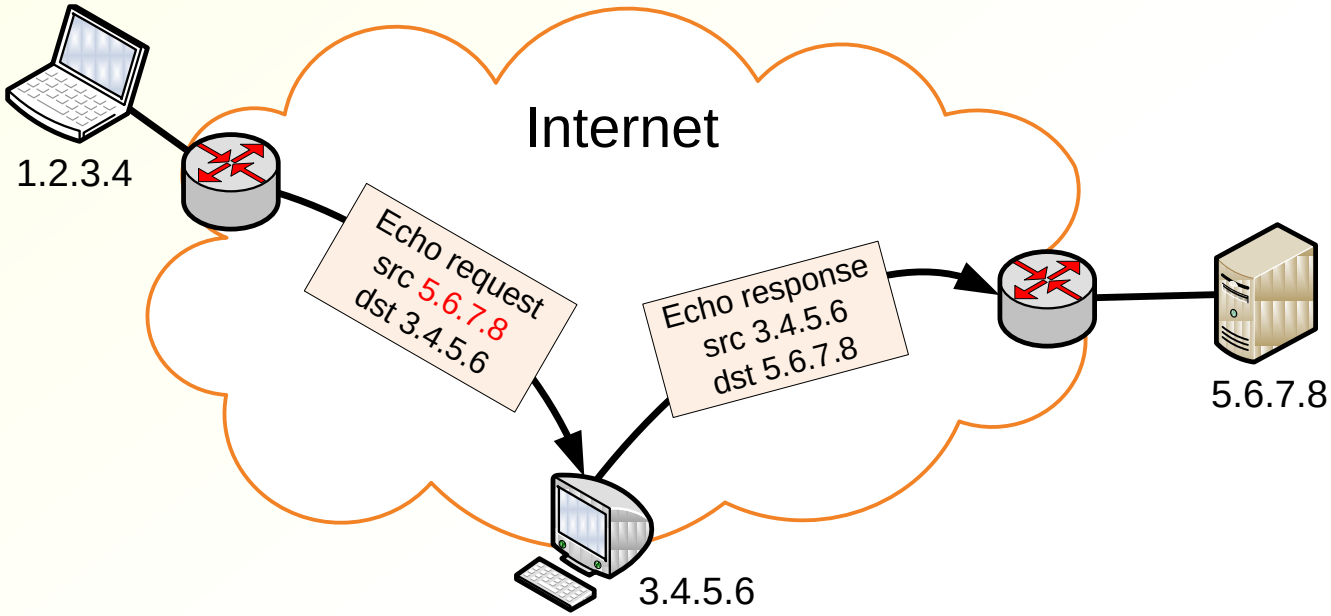
  Typically, packet-flooding exhausts the ISP link bandwidth, in which case the router before the congested link will drop packets
  - Other potential bottlenecks: processing capacity of the gateway router, processing capacity of the IP stack at the target host

Traffic amplification



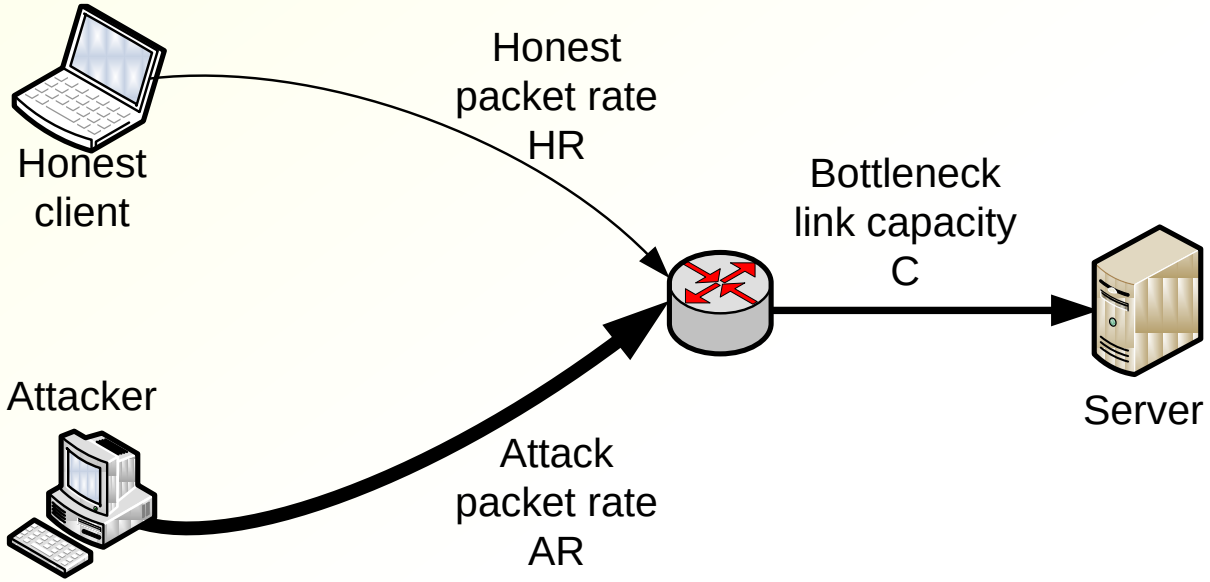
- Example: Smurf attack in the late 90s used IP broadcast addresses for traffic amplification
- Any protocol or service that can be used for DoS amplification is dangerous! → Non-amplification is a key design requirement

Traffic reflection



- Reflection attack: get others to send packets to the target
  - E.g. ping or TCP SYN with spoofed source address
  - DNS reflection + amplification: 64 byte query from attacker, ~3000 byte response to target
- Hides attack source better than just source IP spoofing

Attack impact



- When HR+AR > C, some packets dropped by router
- With FIFO or RED queuing discipline at router, dropped packets are selected randomly
- Packet loss = (HR+AR-C)/(HR+AR) if HR+AR > C; 0 otherwise

  When HR<<AR, packet loss = (AR-C)/AR

Attack impact

- Packet loss = (HR+AR-C)/(HR+AR) if HR+AR > C; 0 otherwise

  When HR<<AR, packet loss = (AR-C)/AR

→ Attacker needs to exceed C to cause packet loss

→ Packet-loss for low-bandwidth honest connections only depends on AR

→ Any AR > C severely reduces TCP throughput for honest client

→ Some honest packets nevertheless make it through:

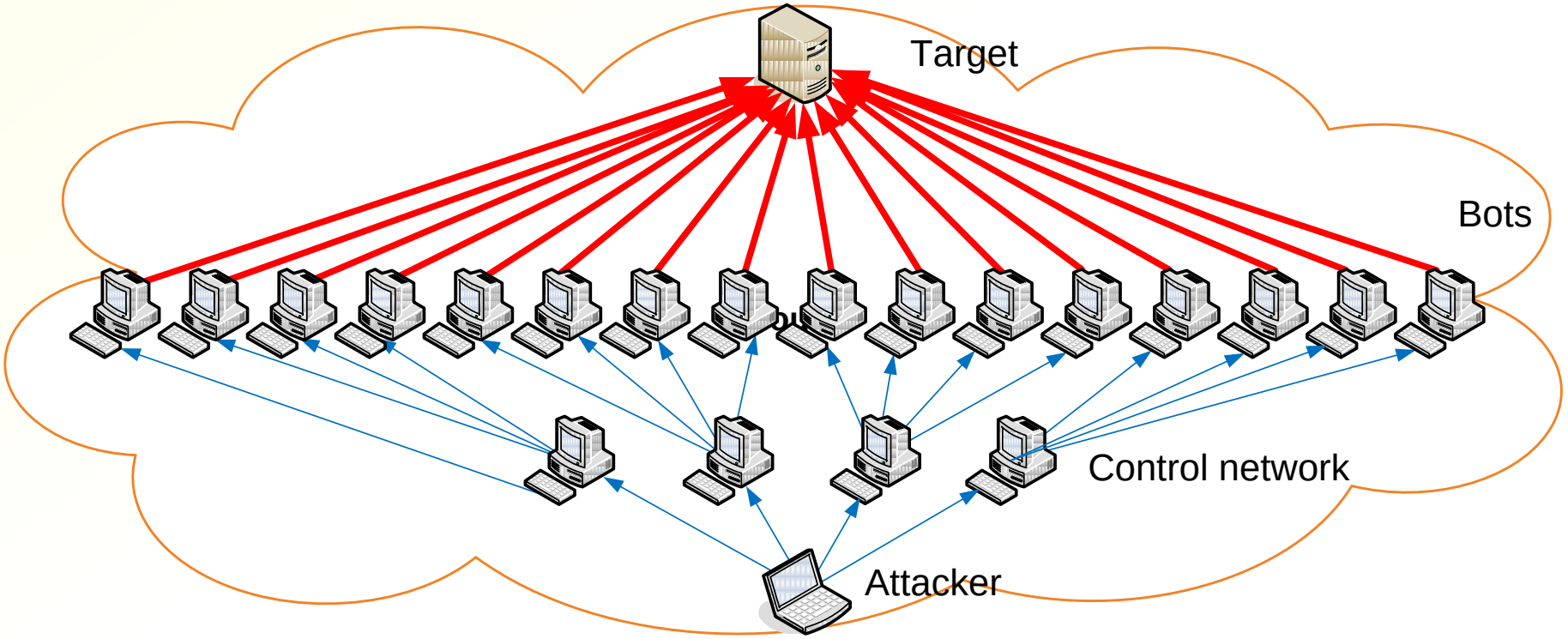
to cause 90% packet loss, need attack traffic AR = 10 × C,

to cause 99% packet loss, need attack traffic AR = 100 × C

Distributed denial of service (DDoS)

Botnet and DDoS

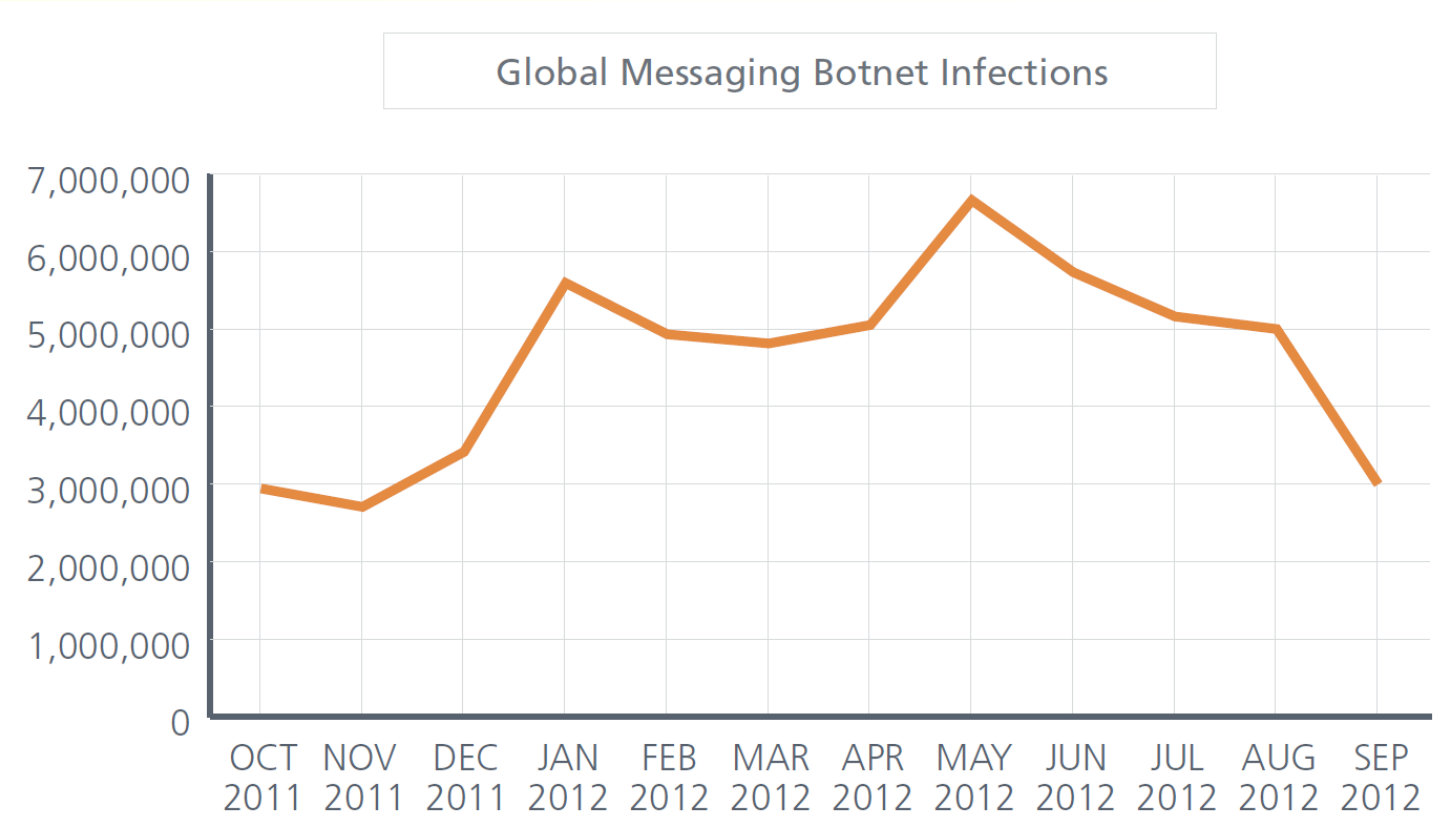
- Attacker controls thousands of compromised computers and launches a coordinated packet-flooding attack



Botnets

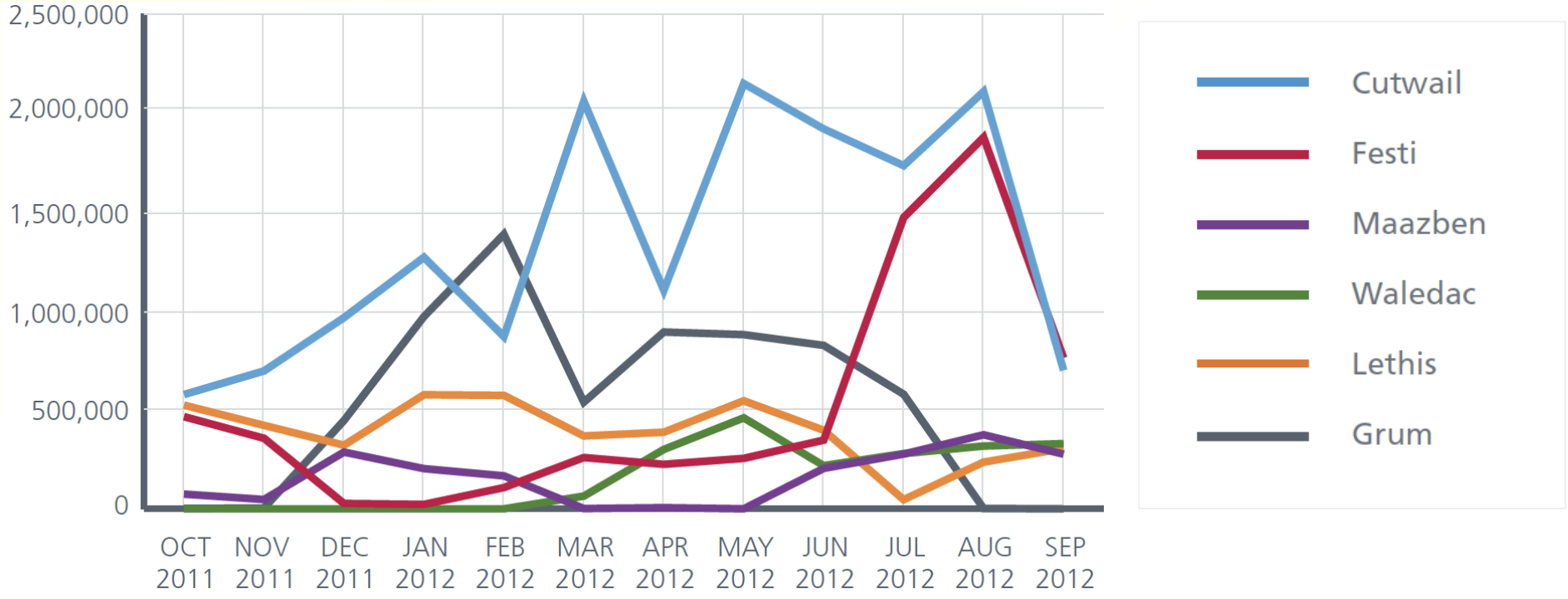
- Bots (also called zombies) are home or office computers infected with virus, Trojan, rootkit etc.
  - Controlled and coordinated by attacker, e.g. over IRC, P2P, Tor
  - Hackers initially attacked each other; now used by criminals
- Examples:
  - Storm, Conficker at their peak >10M hosts (probably)
  - BredoLab ~30M before dismantling
  - Cutweil/Pushdo/Pandex around 2M in August
- Dangers:
  - Overwhelming flooding capacity of botnets can exhaust any link; no need to find special weaknesses in the target
- Q: Are criminals interested in DDoS if they can make money from spam and phishing? What about politically motivated attacks or rogue governments?

Spambot infections



(from McAfee Quarterly Threat Report Q3/2012)

Different botnets



(from McAfee Quarterly Threat Report Q3/2012)

Not the whole picture…

- Only spamming botnets shown
- Lots of different botnets
  - ~1000 Zeus C&C servers (most prolific DIY botnet SW)
  - ~300 SpyEye C&C servers
  - 1-2 million ZeroAccess bots (ad-click fraud)
  - etc..
- DDoS as a service – $50 for 24-hour DDoS

Botnets in news

- "Officials see Iran, not outrage over film, behind cyber attacks on US banks" (NBC News/20.9.2012)
- "DDoS attacks: 150Gb per second and rising" (ZDNet/2.10.2012)
- "DDoS sinks The Pirate Bay" (itnews/14.10.2012)
- "Anti-Kremlin website complains of DDoS attacks" (TheRegister/5.12.2011)
- etc.

- Burma DDoS'd in 2010 before elections
  - International bandwidth ~45Mbps, attack 10-15Gbps

Filtering defenses

Filtering DoS attacks

- Filtering near the target is the main defense mechanisms against DoS attacks
  - Protect yourself → immediate benefit
- Configure firewall to drop anything not necessary:
  - Drop protocols and ports no used in the local network
  - Drop “unnecessary” protocols such as ping or all ICMP, UDP etc.
  - Stateful firewall can drop packets received at the wrong state e.g. TCP packets for non-existing connections
  - Application-level firewall could filter at application level; probably too slow under DoS
  - Filter dynamically based on ICMP destination-unreachable messages

(Q: Are there side effects?)

Flooding detection and response

- Filter probable attack traffic using machine-learning methods
- Network or host-based intrusion detection to separate attacks from normal traffic based on traffic characteristics
- Limitations:
  - IP spoofing → source IP address not reliable for individual packets
  - Attacker can evade detection by varying attack patterns and mimicking legitimate traffic

    (Q: Which attributes are difficult to mimic?)

Preventing source spoofing

- How to prevent spoofing of the source IP address?
- Ingress and egress filtering:
  - Gateway router checks that packets routed from a local network to the ISP have a local source address
  - Generalization: reverse path forwarding
  - Selfless defenses without immediate payoff → deployment slow
- IP traceback
  - Mechanisms for tracing IP packets to their source
  - Limited utility: take-down thought legal channels is slow; automatic blacklisting of attackers can be misused
- SYN cookies (we'll come back to this)

Other defenses

- Extra capacity
  - More link capacity, beefier server
- Optimize
  - Replace resource-consuming content with lighter static content
- Distribute
  - Deploy more servers or reverse proxies
  - Have a true distributed network (Akamai, Cloudflare , ..)
- Buy mitigation
  - Prolexic, Arbor Networks, ..

Most effective attack strategies

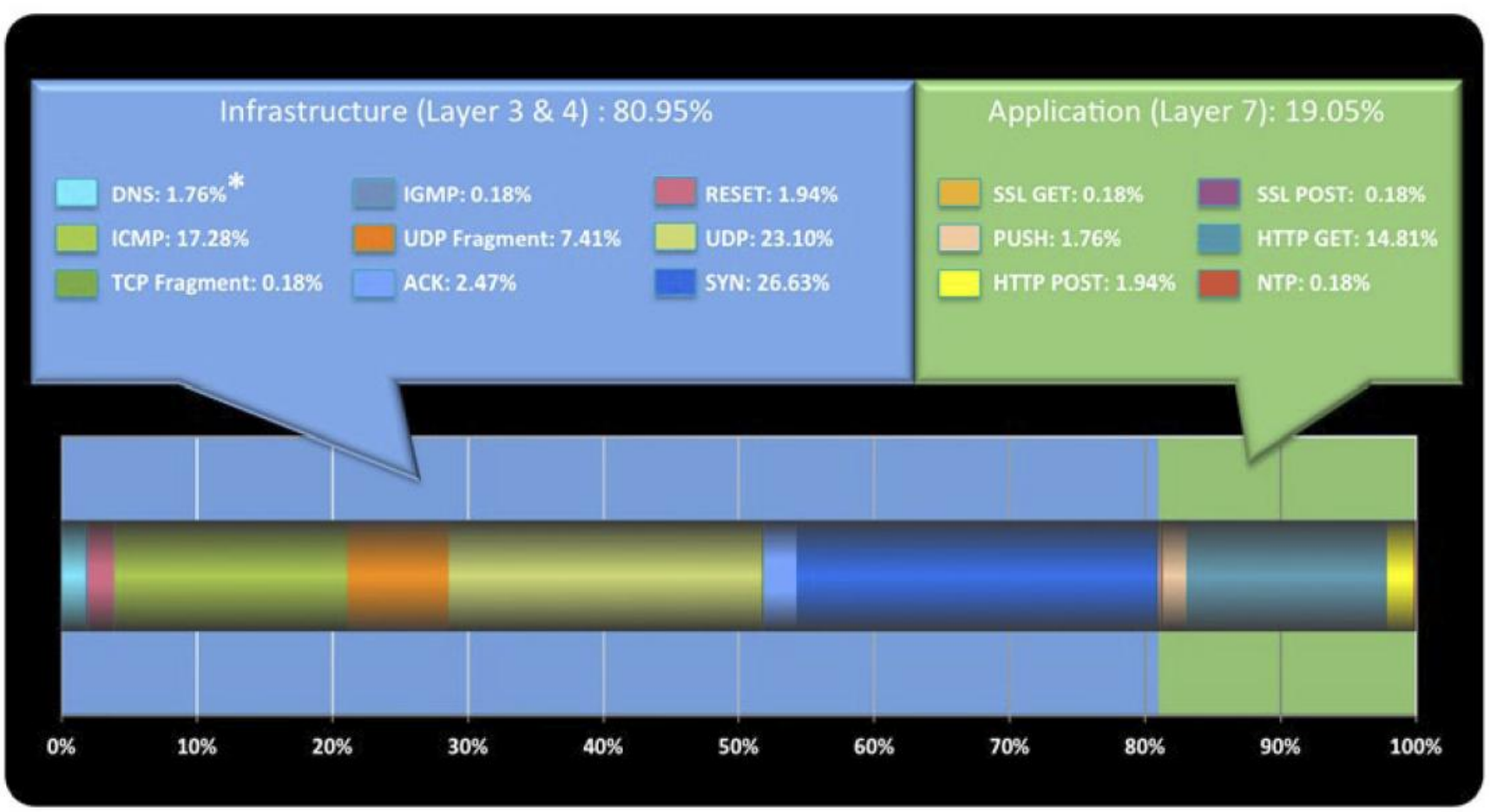
SYN flooding

- Attackers goal: make filtering ineffective → honest and attack packets dropped with equal probability
- Target destination ports that are open to the Internet, e.g. HTTP (port 80), SMTP (port 25)
- Send initial packets → looks like a new honest client
- SYN flooding:
  - TCP SYN is the first packet of TCP handshake
  - Sent by web/email/ftp/etc. clients to start communication with a server
  - Flooding target or firewall cannot know which SYN packets are legitimate and which attack traffic → has to treat all SYN packets equally

DNS flooding

- DNS query is sent to UDP port 53 on a DNS server
- Attack amplification using DNS:
  - Most firewalls allow DNS responses through
  - Amplification: craft a DNS record for which 60-byte query can produce 4000-byte responses (fragmented)
  - Query the record via open recursive DNS servers that cache the response → traffic amplification happens at the recursive server
  - Queries are sent with a spoofed source IP address, the target address → DNS response goes to the target
  - Millions of such queries sent by a botnet

In practise



(from Prolexic Quarterly Global DDoS Attack Report Q2/2012)

Infrastructural defenses

Over-provisioning

- Increase bottleneck resource capacity to cope with attacks
- Recall:

  Packet loss = (HR+AR-C)/(HR+AR) if HR+AR > C; 0 otherwise

  When HR<<AR, packet loss = (AR-C)/AR

→ Does doubling link capacity C help? Depends on AR:

- If attacker sends 100×C to achieve 99% packet loss, doubling C will result in only 98% packet loss
- If attacker sends 10×C to achieve 90% packet loss, doubling C will result in only 80% packet loss
- If attacker sends 2×C to achieve 50% packet loss, doubling C will result in (almost) zero packet loss

QoS routing

- QoS routing mechanisms can guarantee service quality to some important clients and services
- Resource reservation, e.g. Intserv, RSVP
- Traffic classes, e.g. Diffserv, 802.1Q
  - Protect important clients and connections by giving them a higher traffic class
  - Protect intranet traffic by giving packets from Internet a lower class
- Prioritizing existing connections
  - After TCP handshake or after authentication
- Potential problems:
  - How to take into account new honest clients?
  - Cannot trust traffic class of packets from untrusted sources
  - Political opposition to Diffserv (net neutrality lobby)

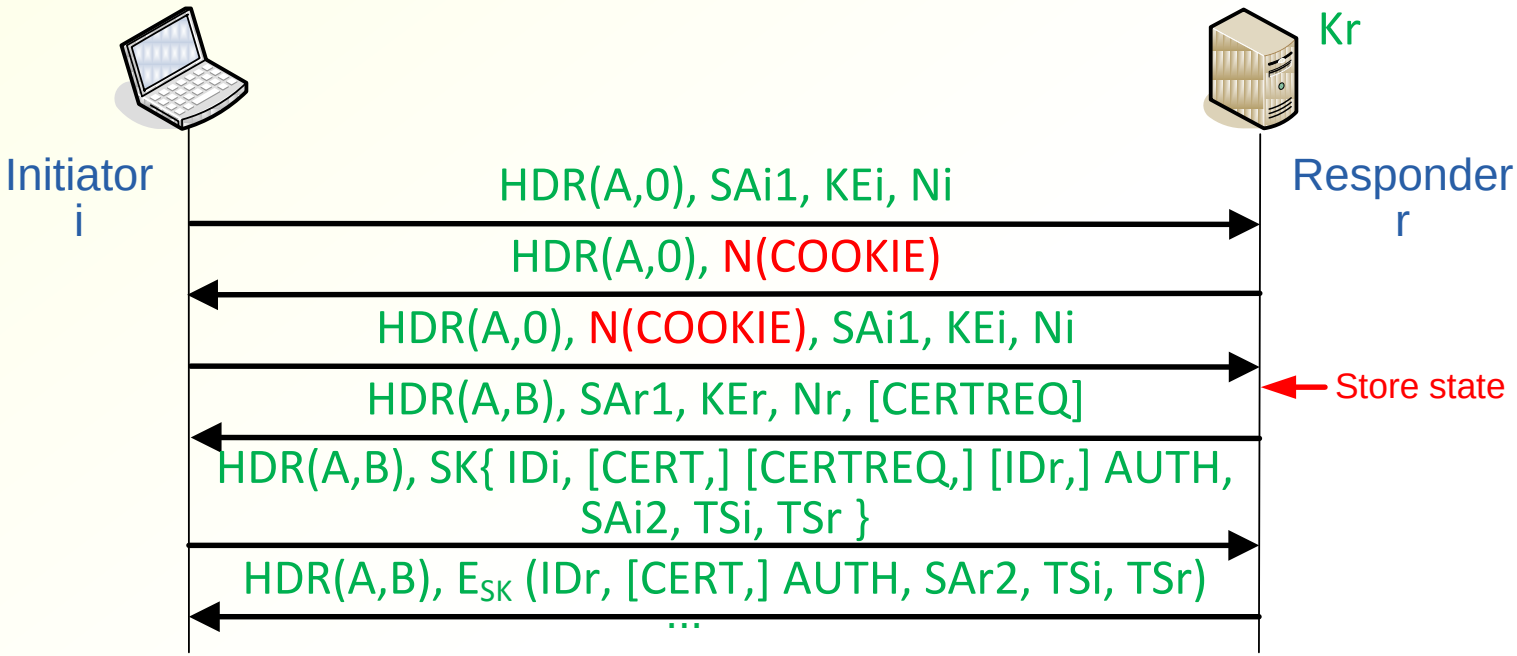
Some research proposals

- IP traceback to prevent IP spoofing
- Pushback for scalable filtering
- Capabilities, e.g. SIFF, for prioritizing authorized connections at routers
- New Internet routing architectures:
  - Overlay routing (e.g. Pastry, i3), publish-subscribe models (e.g. PSIRP)
  - Claimed DoS resistance remains to be fully proven

- Problems?

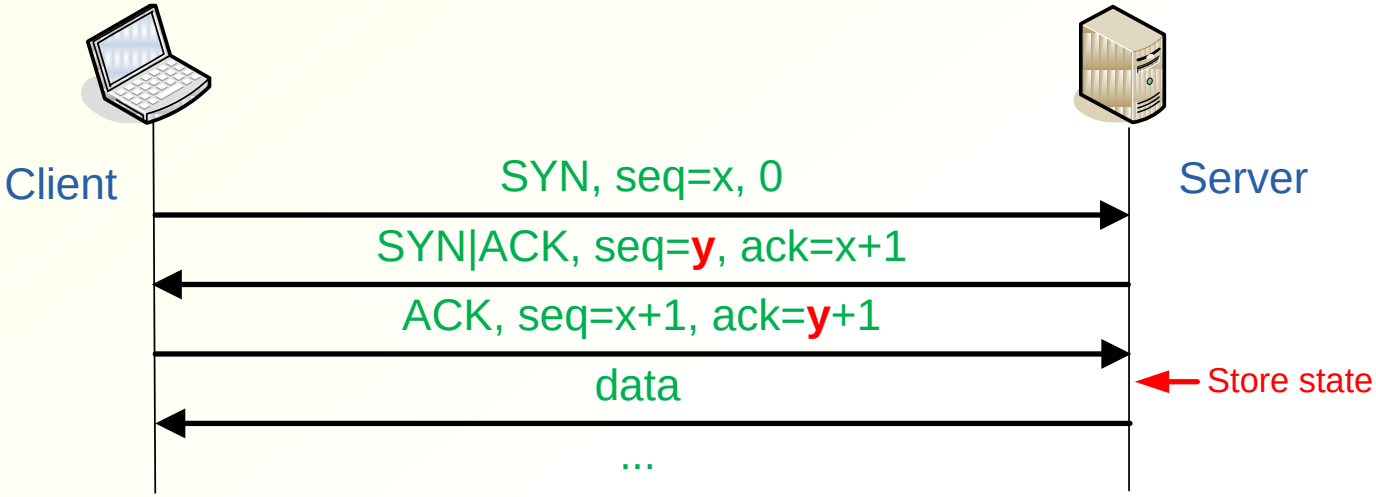
DoS-resistant protocol design

Stateless handshake (IKEv2)



- Responder stores per-client state only after it has received valid cookie:
  COOKIE = hash(Kr , initiator and responder IP addresses)
  where Kr is a periodically changing key known only by responder
  → initiator cannot spoof its IP address
- No state-management problems caused by spoofed initial messages
  (Note: memory size is not the issue)

TCP SYN Cookies

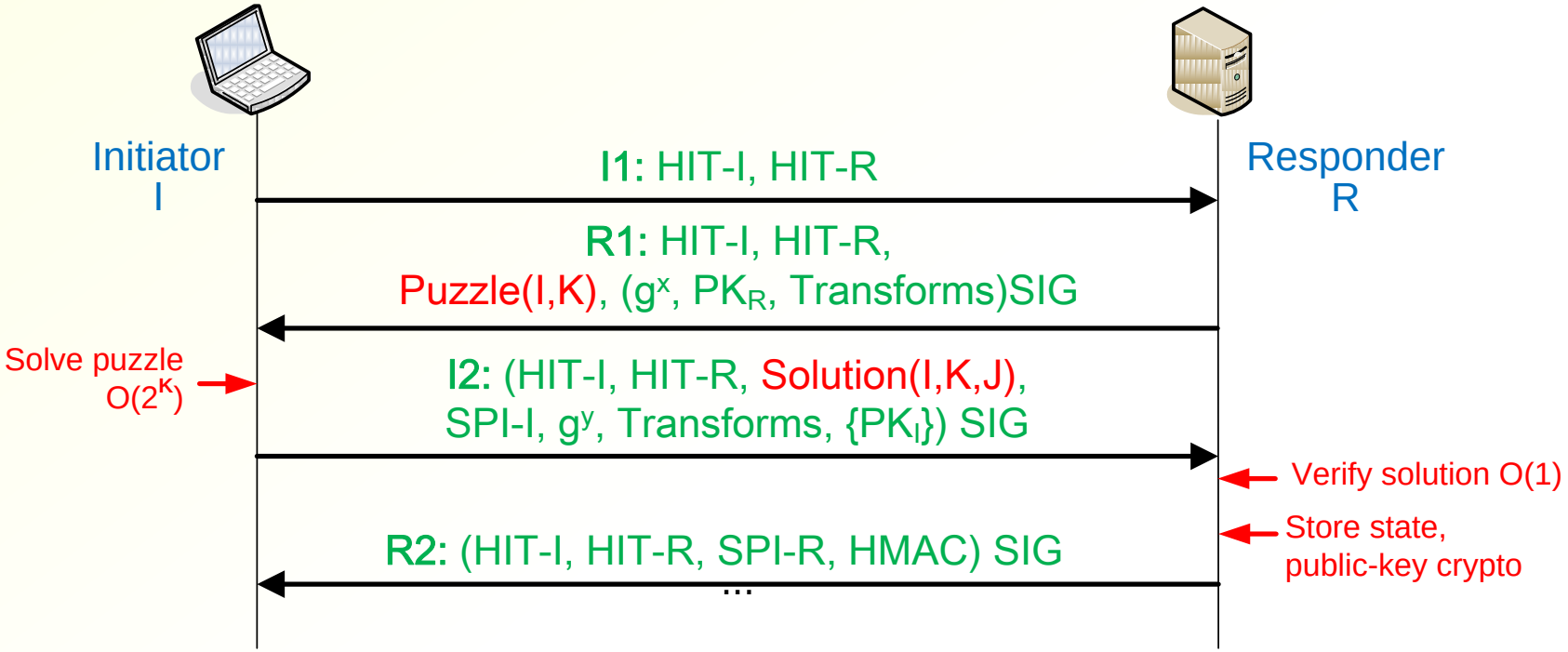


- Random initial sequence numbers in TCP protect against IP spoofing: client must receive msg 2 to send a valid msg 3
- SYN cookie: stateless implementation of the handshake;

  $y = \text{hash}(K_{server}, \text{client addr, port, server addr, port})$

  where K_{server} is a key known only to the server.
- Server does not store any state before receiving and verifying the cookie value in msg 2
- Sending the cookie as the initial sequence number; in new protocols, a separate field would be used for the cookie

Client puzzle (HIP)



- Client "pays" for server resources by solving a puzzle first
- Puzzle is brute-force reversal of a K-bit cryptographic hash; puzzle difficulty K can be adjusted according to server load
- Server does not do public-key operations before verifying the solution
- Server can also be stateless; puzzle created like stateless cookies

Prioritizing old clients

- One way to cope with overload: give priority to old clients and connections, reject new ones
- Filtering examples:
  - Remember client IP addresses that have completed sessions previously, completed handshake, or authenticated successfully
  - Prioritize TCP connections from address prefixes that have had many clients over long time
    (bots are scattered all over the IP address space)
- Protocol design:
  - Give previous clients a credential (e.g. key) that can be used for reconnecting

Cryptographic authentication

- Idea: authenticate packets and allow only authorized ones
  - IPsec ESP
  - Filter at firewall or end host
- Problems:
  - Requires a system for authorizing clients
  - First packet of the authentication protocol becomes the weak point

➔Difficult to use authentication to prevent DoS

Research example: Automated traffic filtering

Scenario

- Number of normal users dwarfed by the attacker bot count
  - x10 ... x1000000
- Attacker is a geographically distributed botnet
  - Difficult to differentiate manually from normal users based on traffic rates or geolocation
- Attacker sends valid requests to the server, aiming to overload the server capacity
  - CPU, memory, database, uplink bandwidth

What to do we have?

- Normal traffic features
  - Source IP, Destination IP, TTL ...
  - Requested resource
  - Request frequency
  - Request consistency
- Attack ←→ Normal dissimilarities
  - Source hierarchy
  - Accessed resource
  - etc?

Learning and filtering

- Create a model of the normal traffic
- Detect attack
- Start filtering requests
- Revert to normal operations once the attack has subsized

- Results: >50% of legitimate traffic served (in simulations)

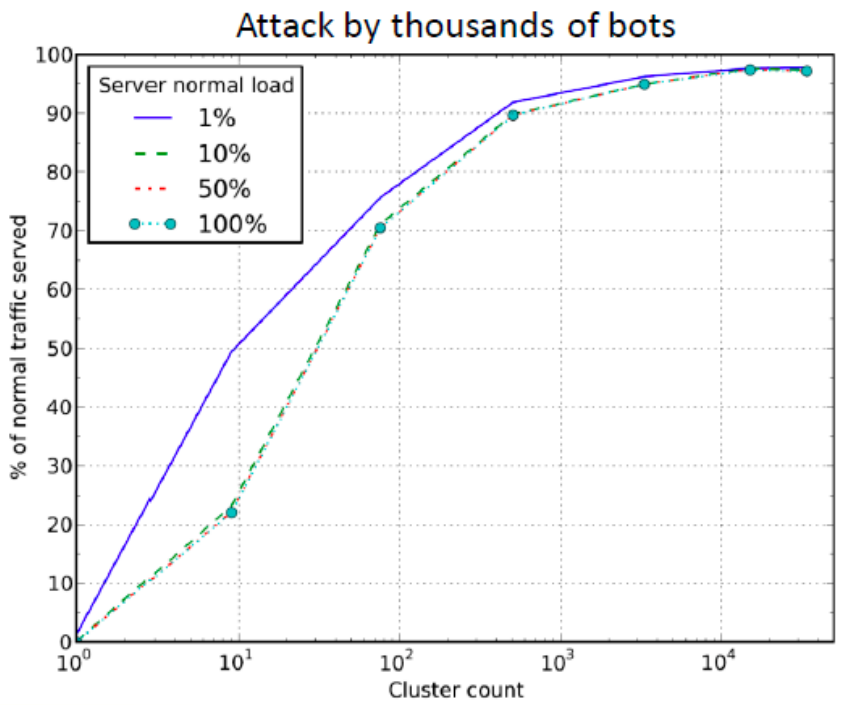
- DDoS vs. Flash crowd?

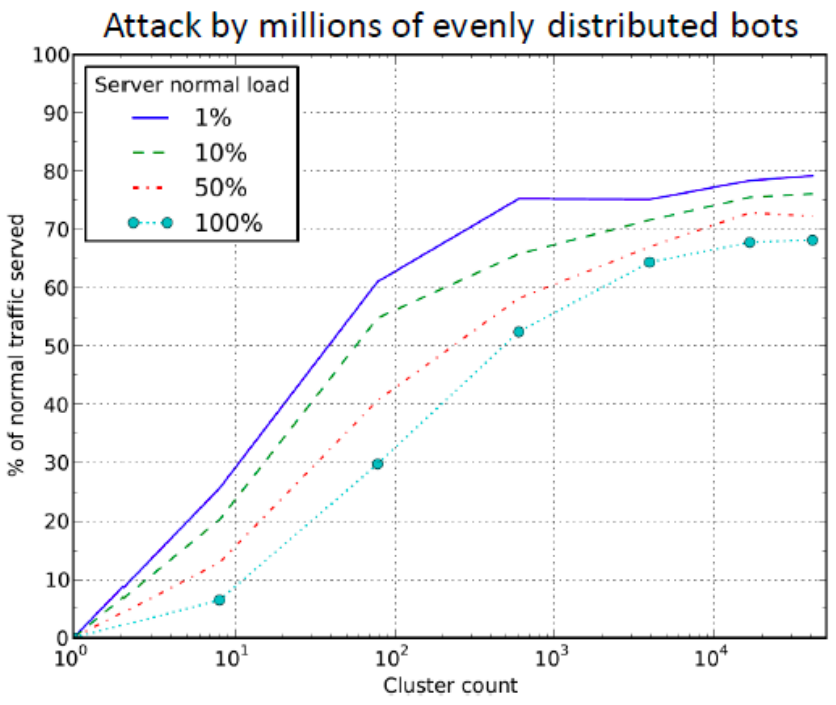
Hierarchical cluster model

- Normal traffic model = hierarchical clusters of request or packets based on features, mainly source IP address
- Provably optimal filtering strategy:
  - Cluster priority = ratio of normal and current traffic in cluster
  - During attack, serve requests in clusters with highest priority

Simulations

- Variable server normal load, attack traffic exceeds normal traffic by a factor of 10e6.

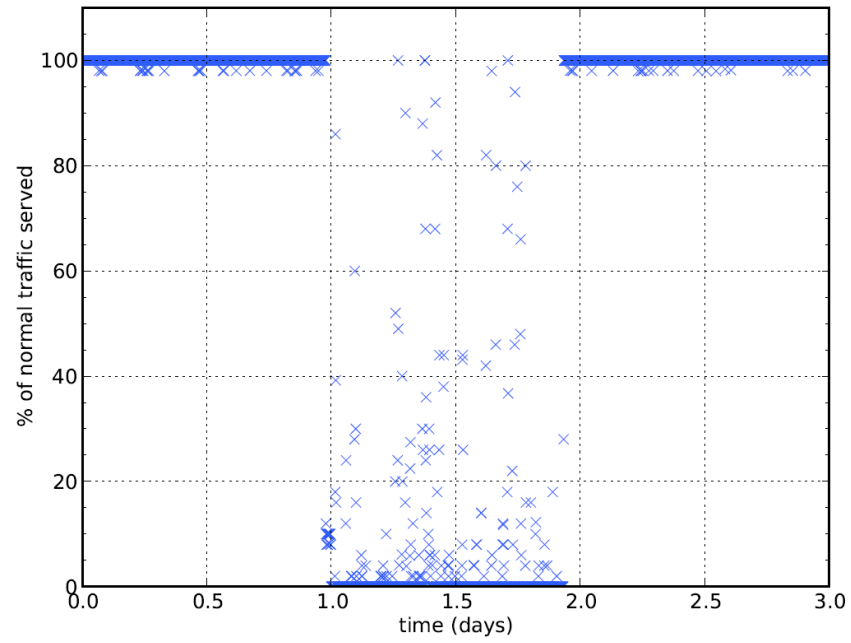




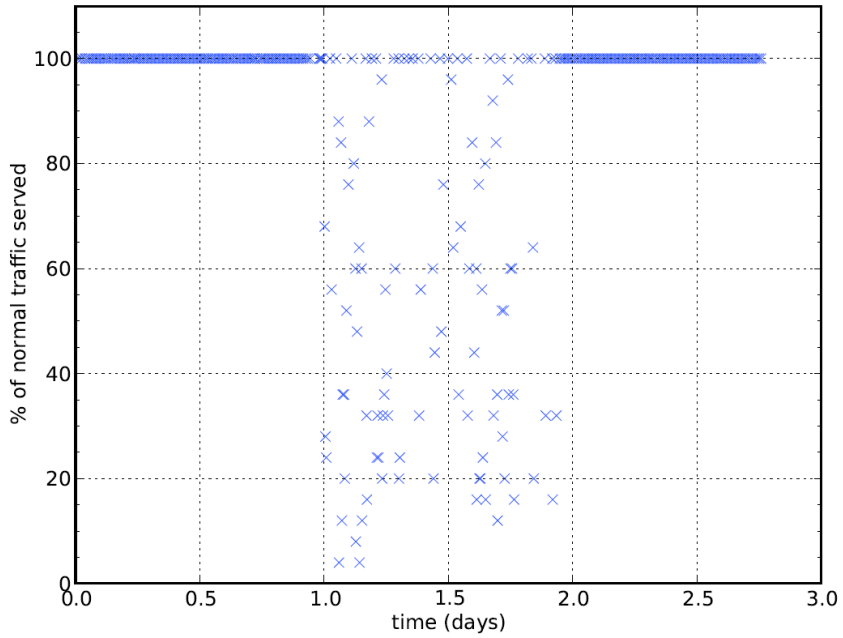
Implementation tests

- Attack scenario: server runs normally at 50 % capacity;
  DoS attack exceeds 10 times the server capacity
  → ~10% of honest requests served
- Filtering deployed → 40..100 % of honest request served

unprotected



protected



Further reading

- DDoSattacks and defense mechanisms: classification and state-of-the-art, Douligeris C. and Mitrokotsa A.
  - http://www.sciencedirect.com/science/article/pii/S1389128603004250
- Prolexic Q2 2012 DDoS Attack Report (requires registration)
  - http://ww.prolexic.com/attack-report
- DDoS and other anomalous web traffic behavior in selected countries, Banks K.B. et al
  - http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6197004&tag=1