




**QUEEN'S
UNIVERSITY
BELFAST**



Denial of Service - Part 2



Dr. Sandra Scott-Hayward

CSC3064 Lecture 18

School of Electronics, Electrical Engineering and Computer Science

Session Overview

- ❑ DDoS Defenses/Countermeasures
 - ❑ Client puzzles
 - ❑ Ingress filtering
 - ❑ Traceback/Edge Sampling
 - ❑ Rate-limiting

References:

Jacobson, Douglas. *Introduction to network security*. CRC Press, 2008.

Schäfer, Günter, and Michael Rossberg. *Security in Fixed and Wireless Networks*. John Wiley & Sons, 2016.

Stallings, William. *Network security essentials: applications and standards*. Pearson Education India, 2007



DDoS Countermeasures

In general, three lines of defense against DDoS attacks:

Attack prevention and preemption
(before the attack)

- These mechanisms enable the victim to endure attack attempts without denying service to legitimate clients

Attack source
traceback and
identification (during
and after the attack)

- This is an attempt to identify the source of the attack as a first step in preventing future attacks

Attack detection and filtering
(during the attack)

- These mechanisms attempt to detect the attack as it begins and respond immediately

Recall: DDoS attack techniques

- Resource Destruction
- Resource Reservation
- Resource Depletion

Defenses

Defenses against resource destruction (disabling services):

- Hacking:
 - Good system administration
 - Firewalls, logging & intrusion detection systems
- Implementation weakness:
 - Code reviews, stress testing, etc.
 - Software updates
- Protocol deviation:
 - Fault tolerant protocol design
 - Error logging & intrusion detection systems
 - “DoS-aware protocol design”, e.g. be aware of possible DoS attacks when reassembling packets

Defenses

Defenses against resource depletion:

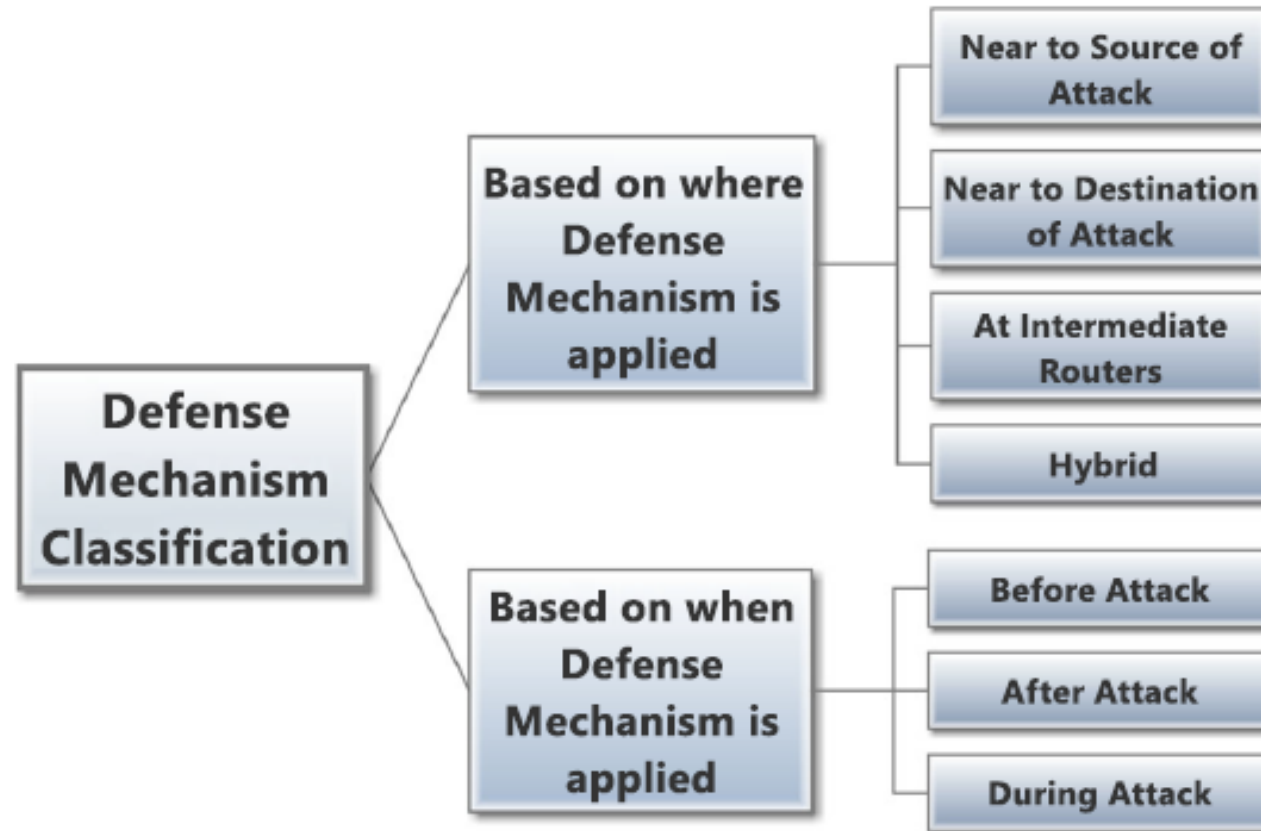
- Generally:
 - Rate Control (ensures availability of other functions on same system)
 - Authentication & Accounting
- Do not perform expensive operations, reserve memory, etc., before authentication
- Expensive computations: careful protocol design, verifying the initiator's "willingness" to spend resources itself (e.g. "client puzzles")
- Memory exhaustion: stateless protocol operation (e.g. verify source of request)

Defenses

Concerning origin of malicious traffic:

- Defenses against single source attacks:
 - Disabling address ranges (helps if addresses are valid)
 - Might also be misused by forged addresses...
- Defenses against forged source addresses:
 - Ingress Filtering at ISPs (in an ideal world...)
 - “Verify” source of traffic (e.g. with exchange of “cookies”)
 - Tracing back the true source of packets with spoofed addresses
- Widely distributed DoS:
 - Anycast infrastructure, as in DNS
 - Distributed data centers & content delivery networks
 - ISP filters with advanced methods to distinguish between bot and honest client (e.g. by verifying JavaScript is correctly executed etc.)

Classification of DDoS defense mechanisms



Gupta, B. B., and Omkar P. Badve. "Taxonomy of DoS and DDoS attacks and desirable defense mechanism in a cloud computing environment." *Neural Computing and Applications* 28, no. 12 (2017): 3655-3682.

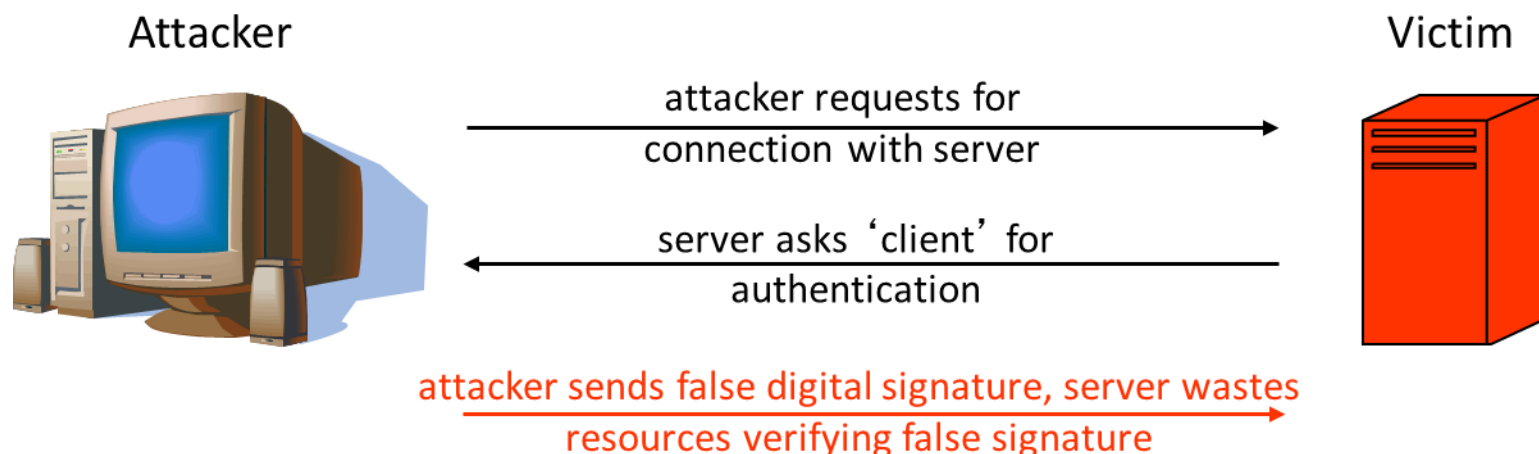
Classification of DDoS defense mechanisms

- Near Source of attack:
 - Low accuracy, Difficult to distinguish between attack traffic and legitimate traffic, who pays/maintains the service?
- Near Destination of attack:
 - Low efficiency, attack traffic already reached/almost reached the victim
- At intermediate routers:
 - Network-based => high processing power and high storage at routers
- Hybrid:
 - Distributed approach, e.g. detection at victim and response (protection) at source

Recall: Resource Depletion with CPU Exhaustion

Category *CPU exhaustion by expensive computations:*

- E.g. 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

Countering CPU exhaustion with Client Puzzles

Basic idea: Slow down the attacker

- Upon a new request, a server generates a new task (“client puzzle”) that the client has to solve before it will be served
- Client puzzles can be easily generated and verified by a server, while clients must use a significant amount of computational resources in order to solve them
- Furthermore, the puzzles' difficulty can be easily scaled based on factors such as server load or server trust of the client – based on DoS attack volume

Drawback:

- Requires changes to both clients and servers
- Honest clients must also spend resources on solving client puzzles
 - Low power, legitimate clients ...

Client Puzzles

Example scheme:

- The server generates two random numbers N_S and X' and computes a cryptographic hash value $h = H(N_S, X')$ of them
- The server then provides the client with one of the random numbers N_S and k bits (for example 8 bit) of the hash value
- The client must then guess random numbers and compute cryptographic hash values until k bits of a resulting hash value match the value that has been supplied by the server
- As cryptographic hash functions can not be inverted, the client on average has to try 2^{k-1} different random numbers until they find one number X so that k bits of $H(N_S, X)$ match the value provided by the server
- However, in order to generate and check the client puzzle, the server just needs to compute the cryptographic hash function twice

Examples

TCP connection floods

Example challenge: $C = \text{TCP server-seq-num}$

- First data packet must contain puzzle solution
 - Otherwise TCP connection is closed

SSL handshake DoS:

- Challenge C based on TLS session ID
- Server: check puzzle solution before RSA decrypt.

Source Identification

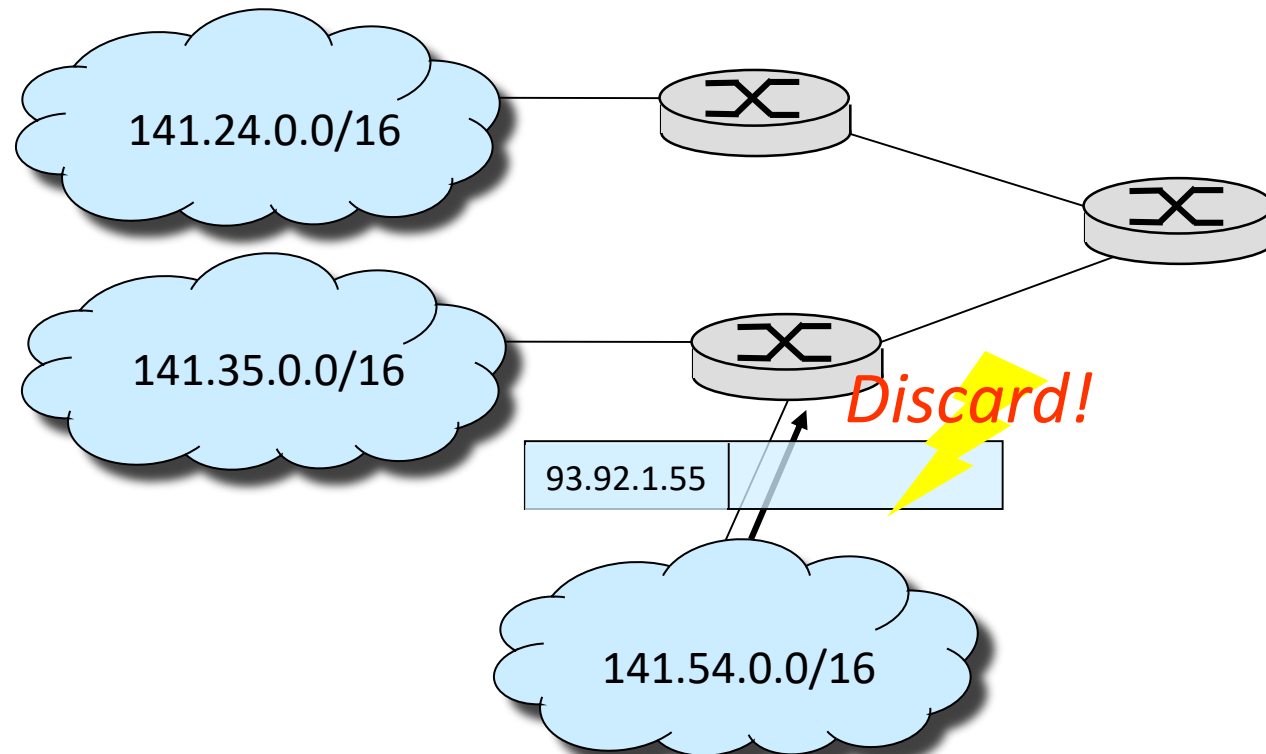
Goal: Identify packet source

Ultimate goal: Block attack at the source

Methods: Ingress Filtering, Traceback ...

Ingress Filtering (RFC 2827 – BCP 38)

Routers block arriving packets with illegitimate source addresses.



Ingress Filtering (RFC 2827)

- To reduce the address space that can be used by the attacker by filtering the packets at the edge of the network
- Ingress filtering:
 - Incoming packets with a source address belonging to the network are blocked
 - Incoming packets from the public Internet with a private source address are blocked
- Egress filtering:
 - Outgoing packets that carry a source IP address that does not belong to the network are blocked

Ingress Filtering (RFC 2827)

Problems:

1. Issues with Mobile IP (some protocols require a foreign address)
2. Larger management overhead at router-level
3. Potentially large processing overhead at access routers
(e.g. large ISP running a large AS with numerous IP ranges and DHCP)
4. Universal deployment needed (if 10% of ISPs do not implement, there's no defense)

And: ISPs do not really have an incentive to block traffic...

- Paid by number of transmitted bytes, which traffic is “malicious” and which is not, “malicious” to whom?

Traceback – Simple Method

Goal: Identify the source address (or at least the ingress point) and the attack path of a packet without relying on the source address information
(given set of attack packets, determine path to source)

How? Change routers to record information in packets

Write path into network packet

- Each router adds its own IP address to packet
- Victim reads path from packet

Problem:

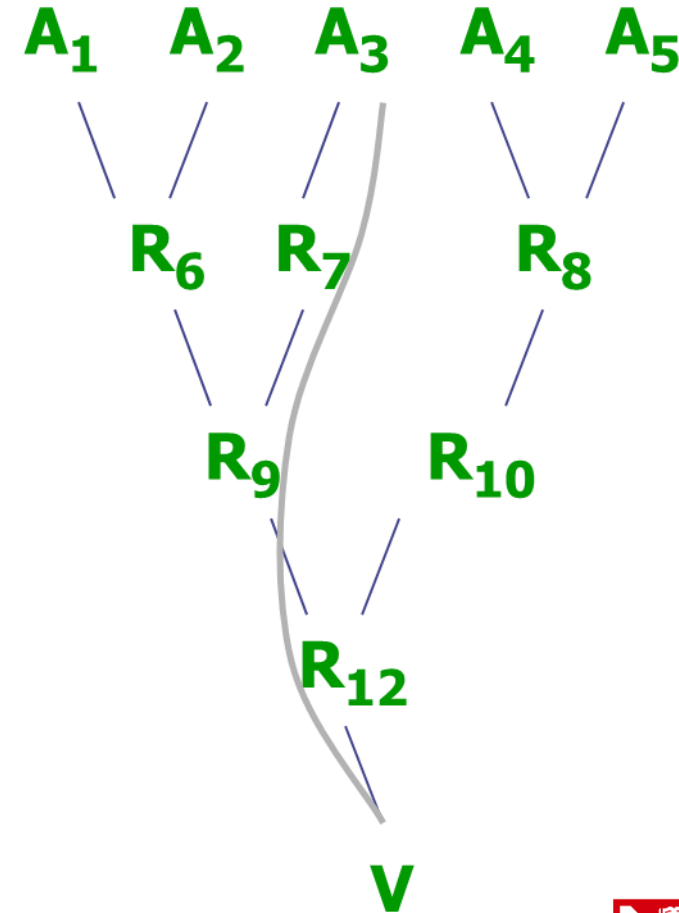
- Requires space in packet, Path can be long, No extra fields in current IP format
 - Changes to packet format too much to expect

Traceback – Better Idea

DDoS involves many packets on the same path

Store one link in each packet

- Each router probabilistically stores own address
- Fixed space regardless of path length



Edge Sampling

- Data fields written to packet:
 - Edge: *start* and *end* IP addresses
 - Distance: number of hops since edge stored
- Victim can deduct graph of edges and reconstruct attack tree

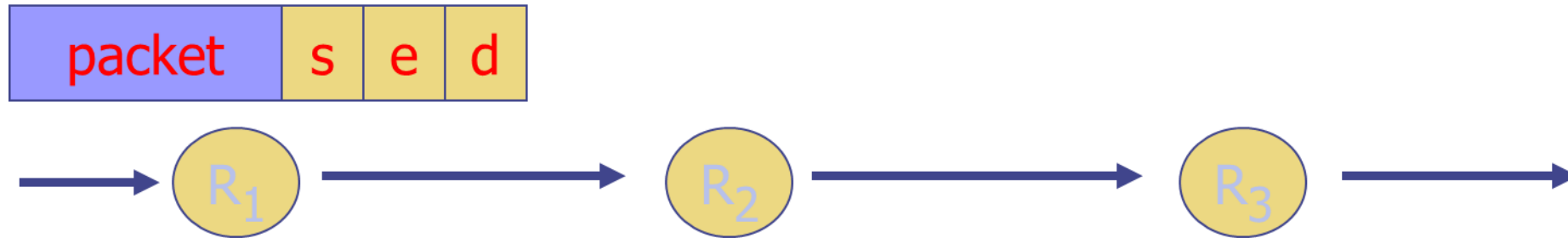
- Marking procedure for router R

```
for each packet w
  let x be a random number from [0..1)
  if x < p then
    write R into start address (w.start)
    write 0 into distance field (w.distance)
  else
    if w.distance == 0 write R into end field (w.end)
    increment w.distance
```

Edge Sampling

Packet received

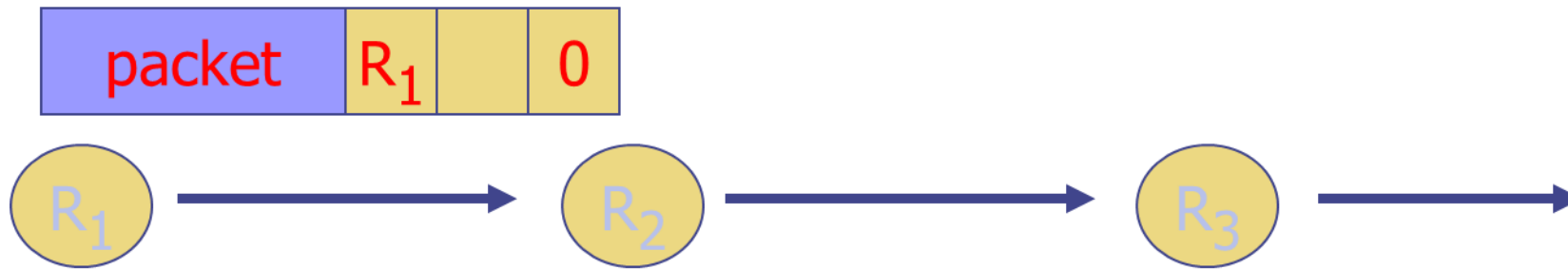
- R_1 receives packet from source or another router
- Packet contains space for start, end, distance



Edge Sampling

Begin writing edge

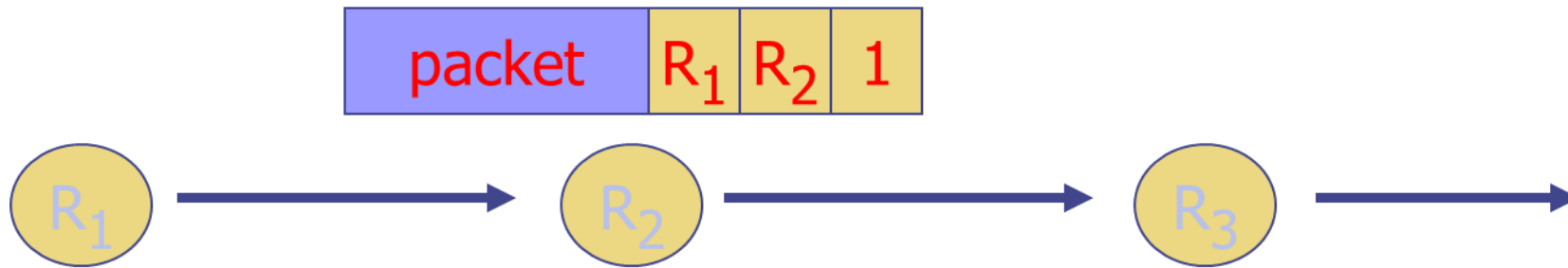
- R_1 chooses to write start of edge
- Sets distance to 0



Edge Sampling

Finish writing edge

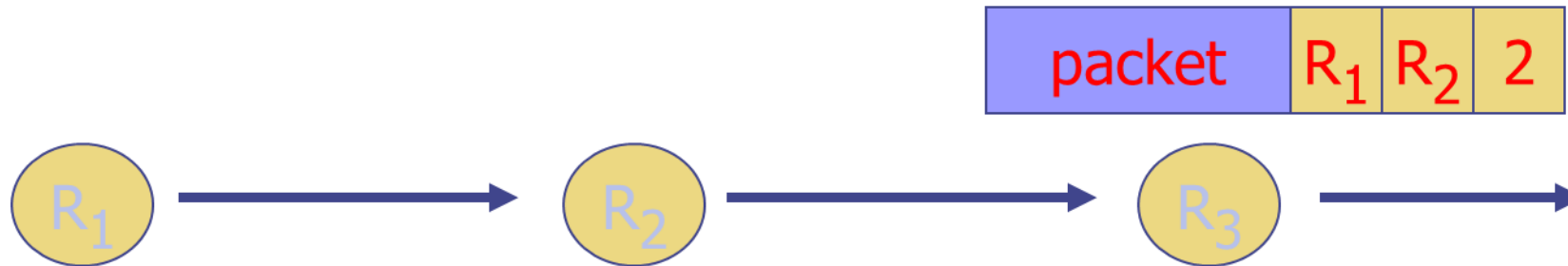
- R_2 chooses not to overwrite edge
- Distance is 0
 - Write end of edge, increment distance to 1



Edge Sampling

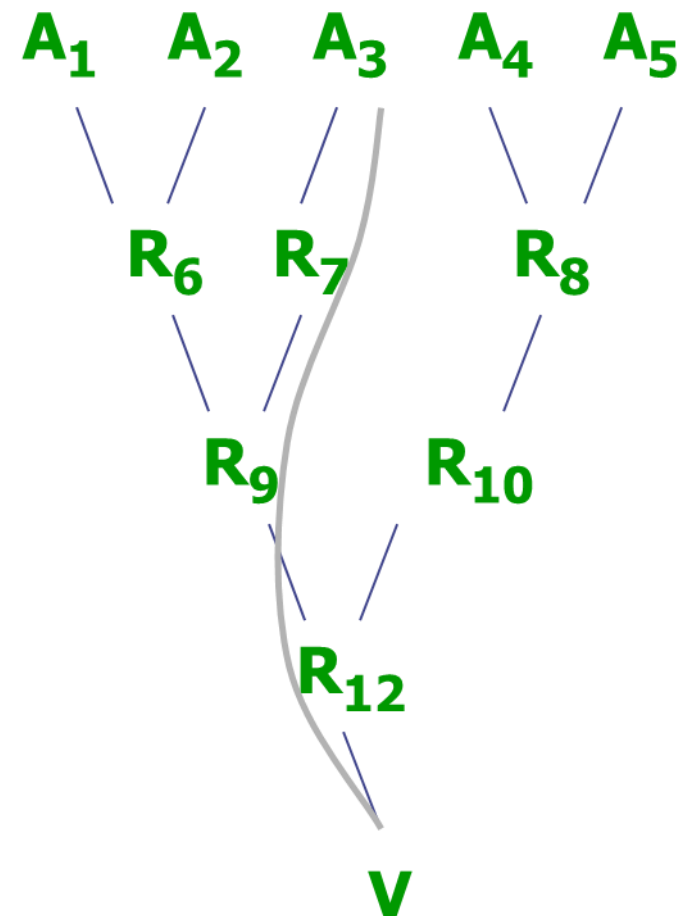
Increment distance

- R_3 chooses not to overwrite edge
- Distance > 0
 - Increment distance to 2



Path reconstruction

- Extract information from attack packets
- Build graph rooted at victim
 - Each (start,end,distance) tuple provides an edge
- # packets needed to reconstruct path
 - $E(X) < \frac{\ln(d)}{p(1-p)^{d-1}}$
 - where p is marking probability, d is length of path



Path reconstruction

Path Reconstruction Procedure at victim v :

```
let G be a tree with root v
let edges in G be tuples (start, end, distance)
for each packet w from attacker
    if w.distance = 0 then
        insert edge (w.start, v, 0) into G
    else
        insert edge (w.start, w.end, w.distance) into G

remove any edge (x,y,d) with  $d \neq d(x,v)$  in G
extract path  $(R_1, \dots, R_j)$  by enumerating acyclic paths in G
```

Where to store the edge?

Edge information is compressed, fragmented, and encoded in the identification field of the IP header



Note: incompatible with IP fragmentation (also uses the ID field)

Version	Header Length
Type of Service	
Total Length	
Identification	
Flags	Fragment Offset
Time to Live	
Protocol	
Header Checksum	
Source Address of Originating Host	
Destination Address of Target Host	
Options	
Padding	
IP Data	

Edge Sampling: Pros/Cons

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

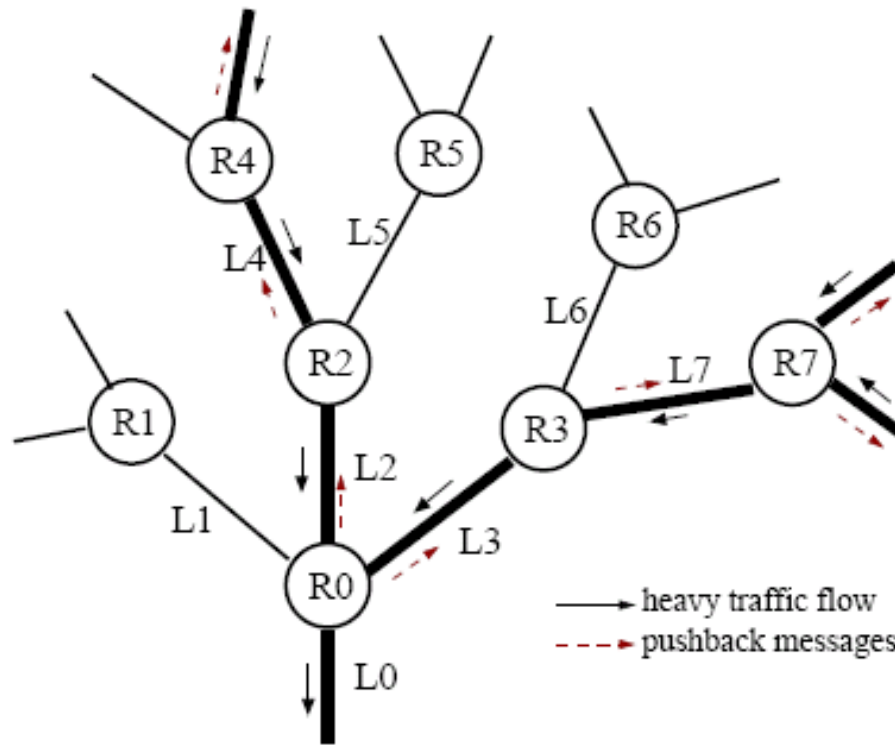
- 😞 Works mainly for bandwidth exhaustion attacks
 - Many packets needed for reconstructing attack path
- 😞 Victim under attack needs large amount of memory (many packets!) and processing time
- 😞 In order to avoid spoofing, authentication needed (PKI, signatures)

Rate-limiting

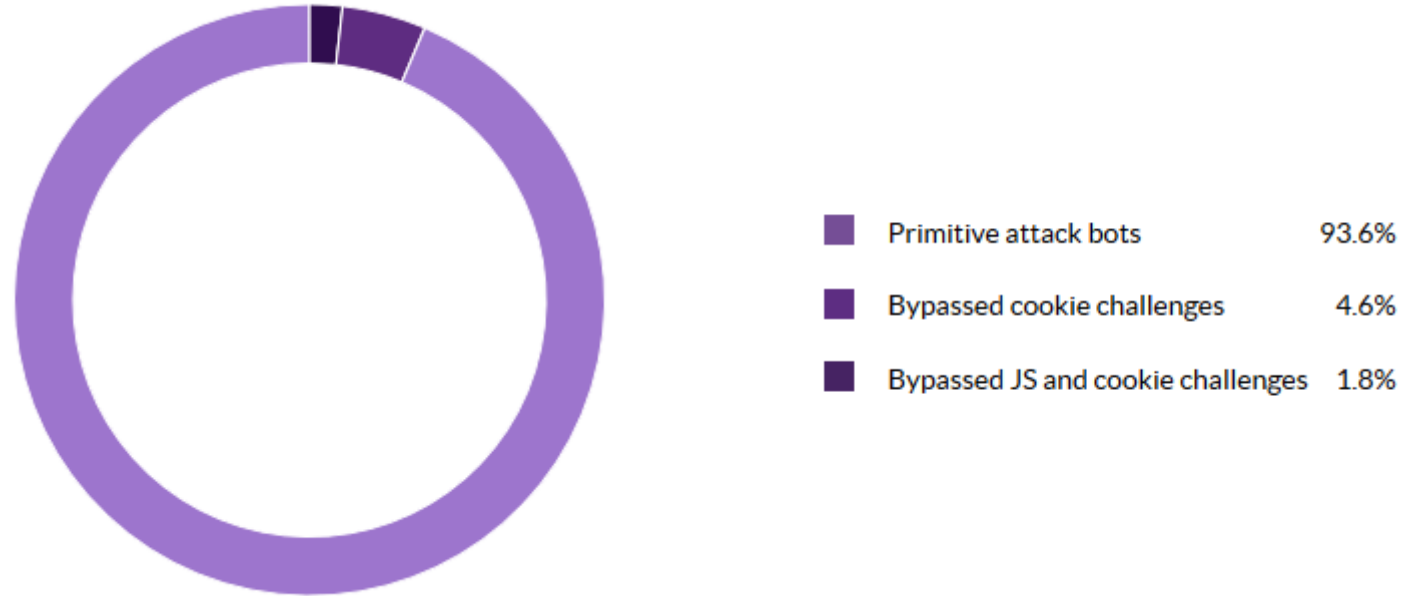
- Allow a router to control the transmission rate of specific flows
- Can be used to control network congestion
- If packets arrive at a higher rate, they will be queued or dropped
- If attack flows can be identified, they can be rate-limited
- Rate-limiting mechanisms are deployed when the attack detection has a high number of false positives or cannot precisely characterize the attack stream.
- Problem: Legitimate users will experience degraded service

Propagating rate-limit requests

Iteratively block attacking network segments



Bot capabilities



Application layer attack traffic that can bypass cookie challenges has increased.

Source: <https://www.incapsula.com/ddos-report/ddos-report-q3-2017.html>

Multi-vector attacks



Multi-vector attacks are DDoS incidents where an attacker uses different protocols for the DDoS assault, such as SYN, TCP, UDP, ICMP, NTP, DNS, and others.

Source: <https://www.incapsula.com/ddos-report/ddos-report-q3-2017.html>

Sophisticated attacks

Step 1: DDoS – Flood the network, Fill logging systems, Hide a unique event

Step 2: Sensitive Data Exfiltration

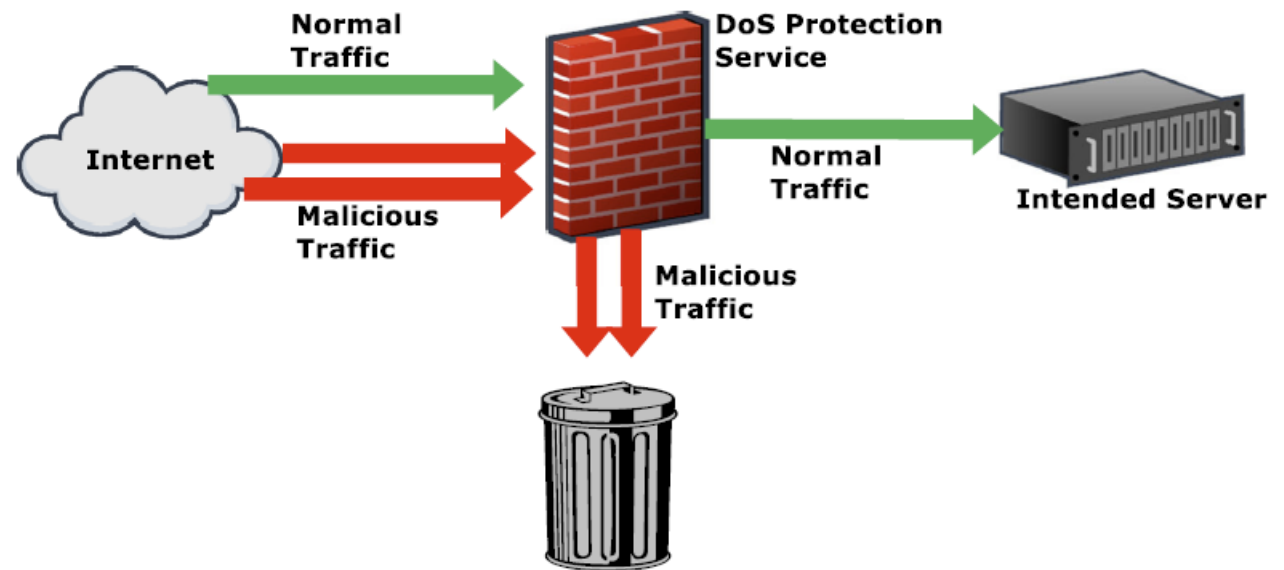
Distract from the real attack ... called Dark DDoS

DDoS Mitigation Services (middleboxes)

Network Security Vendors

For example,

- CloudFlare
- Arbor Networks
- Radware
- Akamai
- NexusGuard
- F5
- ...



Summary

- Increasing dependence of modern information society on availability of communication services
- While some DoS attacking techniques can be countered with “standard” methods, some can not:
 - Hacking, exploiting implementation weaknesses, etc. may be countered with firewalls, testing, monitoring etc.
 - Malicious protocol deviation & resource depletion is harder to defend against
- Designing DoS-resistant protocols is crucial:
 - 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 such as authentication, cookies, client puzzles, etc.

Summary

- DDoS Countermeasures/Defences
- Client Puzzles
- Ingress Filtering
- Traceback/Edge Sampling
- Rate-Limiting
- Recent DDoS attack methods

Questions?

Next Session: Cloud/Virtualization Security
Tuesday, 26 March 2019