Distributed Denial of Service

CS557

A DoS-limiting Network Architecture
X. Yang, D. Wetherall, and T. Anderson, 2005
Denial of Service (DoS)

- Exhaust resources at (or near) the target
  - Resources: CPU, Memory, Bandwidth
    - Legitimate clients cannot access the target server
    - Special case: Distributed Denial of Service (DDoS)
- Should we care?
  - For us: interesting problem; difficult to address.
  - For others: monetary loss, infrastructure security.
Example: Bandwidth Exhaustion DDoS Attack
Defending DDoS Attacks

• Filtering
  – Ingress filtering, Pushback, Traceback

• Information hiding
  – Secure overlay services (SOS), i3

• Proof of work
  – Congestion puzzles, Defense by offense

• Network capabilities
  – Stateless Internet flow filtering (SIFF),
  Traffic validation architecture (TVA)
Network Capabilities

• Fundamental change to the Internet service model!

• Sender must have authorization from receiver to send traffic
  – Receiver decides what traffic it wants to receive or not receive
  – Network enforces receiver’s decision
Phase 1: Request Capabilities

pre-capabilities

source
destination
Pre-Capabilities

- Cryptographically generated at each router $R$
  - Each router can independently verify its own pre-capability
- Timestamp + $\text{Hash}(\text{SrcIP}, \text{DstIP}, \text{time}, R_{\text{secret}})$
  - $\text{SrcIP}, \text{DstIP}$ tie the capability to a flow
  - $R_{\text{secret}}$ : secret key only known to the router (the same secret is used for all pre-capabilities)
    - $R_{\text{secret}}$ changed twice per timestamp roll over
Phase 2: Authorizing a source

The diagram illustrates the process of authorizing a source. The attacker is indicated by the flag with a skull and crossbones, pointing towards the source. The source and destination are connected by a path, with the attacker attempting to bypass the network. The host-capability is shown with a SYN flag indicating a potential security risk.
Host-Capability

- Cryptographically generated at the destination using pre-capabilities
- Timestamp + Hash(pre-capabilities, N, T)
  - N is the number of packets authorized per capability
  - T is the time period for which the capability is valid
- Routers track N, and T
Phase 3: Send Traffic

- Source
- Attacker
- BOGUS
- Pre-capability verify
- Host-capability verify
- Destination
Traffic Classes

• Traffic classes
  – Request
    • Request packets (such as TCP SYN)
  – Regular
    • Packets with capabilities
  – Demoted
    • Packets with invalid capabilities
  – Legacy

• Separate bandwidth allocated to regular and request traffic at each router
Denial of Capabilities (DoC)

- Attacker sends flood of request packets
  - Legitimate requests get lost before reaching the destination

- TVA solution:
  - Path identifiers (Pi)
Path Identifiers

• Routers include Pi bits into request packets
  – Similar to pre-capabilities

• Next downstream router fair-queues on the Pi bits inserted at upstream routers
  – Number of Pi queues = number of upstream routers
Simulation Topology

10 legitimate clients

1 ~ 100 attackers

bottleneck link

10Mbps, 10ms

destination

10ms

colluder

10ms
Simulation Results (1)

Legacy traffic floods
Simulation Results (2)

Request traffic floods
Summary

• Capabilities make fundamental changes to the Internet to defend DDoS attacks
  – Sender needs authorization from receiver to send traffic
• Capabilities setup is challenging
  – Denial of Capability attacks
• Is this a viable architecture? Incremental deployment? Co-existence with current architecture?
Cossack: Coordinated Suppression of Simultaneous Attacks
Cossack Overview

- Distributed set of watchdogs at network perimeter
  - Local IDS
  - Group communication
  - Topology information (when available)
- Fully distributed approach
  - Peer-to-peer rather than master-slave
  - Attack-driven dynamic grouping of watchdogs
  - Attack correlation via coordination with other watchdogs
  - Independent, selective deployment of countermeasures
Cossack: A Simplified View
Attacks Begin

attacker

watchdog

target
Watchdogs Communicate Using YOID
Watchdogs Install Filters and Eliminate Attack
Detecting Source Spoofed Attacks
Cossack Components

• Watchdog
• Local IDS
  – Snort (with Snort Plugin)
  – Any other available IDS
• Network Inspector
• Yoid Application Layer Multicast
Watchdog Actions

- Continuously monitors destination-based statistics to detect local attacks
- Requests source-based statistics for suspect flows
- Shares attack information with other watchdogs in networks originating the attack
- Watchdogs in attacking networks perform careful analysis of offending flows and take countermeasures
Cossack Watchdog Architecture
Cossack Snort Plugin

• Calculates aggregate packet rates in real-time
  – Grouped by <source, destination, protocol, port>
• Relays packet rates to watchdog
• Receives requests from watchdog to change averaging window to help identify suspicious traffic
Cossack Plugin Operation

Packet Averages Grouped by Destination Address

Packet Flow Statistics
Request more stats
Cossack Plugin Operation
Cossack Network Inspector

Tool to determine detection thresholds for watchdogs

- Interfaces with the Cossack snort plugin
- Collects aggregate level network traffic statistics
  - Traffic filters created using snort rules
Cossack Performance

- Response time: 5 – 30 seconds
- Insensitive to attack type
Final Thoughts

• Why is DDoS defense so hard?
  – Technical issues?
  – Policy?
  – Legal?

• What types of DDoS defenses are out there?
  – Arbor- local presence
  – Akamai – distribute content
  – CloudFlare – anycast
  – See
    https://www.youtube.com/watch?v=4BPibf6C35E