Capability-Based Denial of Service Defense

Xiaowei Yang
Tom Anderson
David Wetherall
(yxw, tom, djw)@cs.washington.edu

6

Background

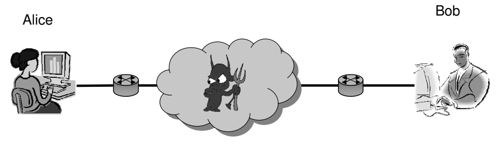
7

Network Security

- Confidentiality and integrity
  - Protection against unauthorized access to or modification of information
- Resource Availability
  - Protection against the denial of service to legitimate users

8

Confidentiality and Integrity



- **Successful defense with cryptographic techniques**
  - **Encryption, public-key signature**
  - **TLS, IPSec**

9

Network Security

- Confidentiality and integrity
  - Protection against unauthorized access to or modification of information
- Resource Availability
  - Protection against the denial of service to legitimate users

10

Denial of Service (Dos) Attacks

- Consumption of shared resources
  - Bandwidth, memory, or CPU time
- Disruption of configuration information
  - routing
- Disruption of physical network components

11

Why Attack?

- Vandalism
- Egotism
- Cyber mafia
  - “In March, 2004, a sustained campaign of DoS attacks was launched against Britain's top 20 gambling sites after extortion demands were met and rejected.”
- Cyber war

Attacker’s Goals

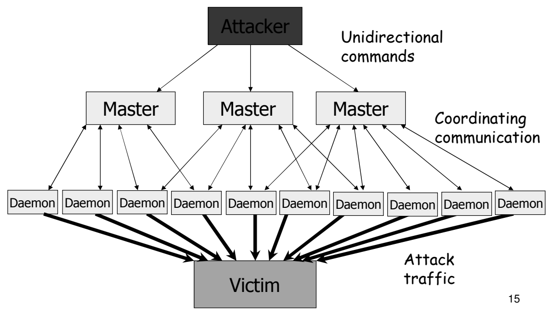
- Hide
- Maximize damage

These goals are essential to understand what makes an attack effective

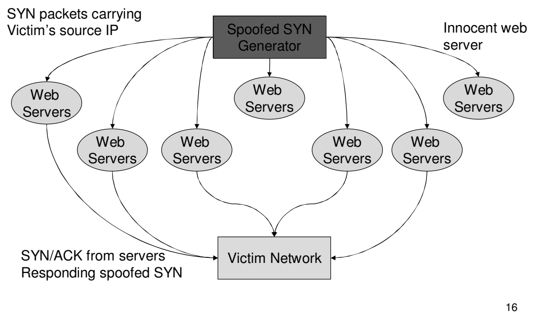
How to Attack

- Flooding traffic
  - Spoof address to hide
  - Distributed DoS to hide and to maximize damage
    - Multiple (weak) machines against (strong) victim
- Attack on bandwidth, server memory or CPU

Ex1: bandwidth attack using botnets



Ex2: bandwidth attack using reflectors


SYN packets carrying Victim’s source IP
Spoofed SYN Generator
Innocent web server
Web Servers
Victim Network
SYN/ACK from servers Responding spoofed SYN

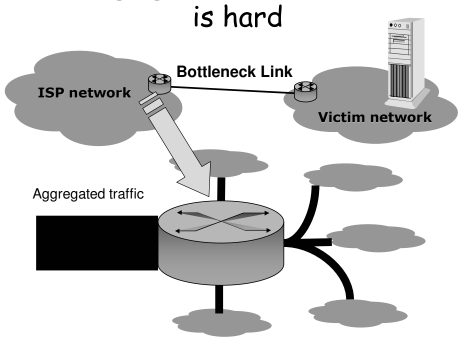
More flooding attacks

- Memory attack
  - TCP SYN flooding
- CPU attack
  - Unnecessary computation, e.g., TLS attack
- Easier to mitigate
  - Proper protocol design: cookies, puzzles

Defense against bandwidth attacks was not successful.

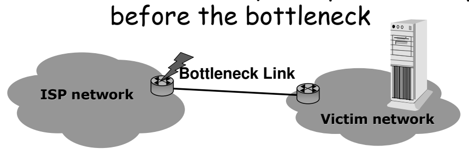
18

Defending against bandwidth attacks is hard



19

Effective defense requires packets drop before the bottleneck



- But
  - ISPs are not willing to deploy complex filters for each client
  - ISPs have no strong incentive; they charge clients for the traffic

20

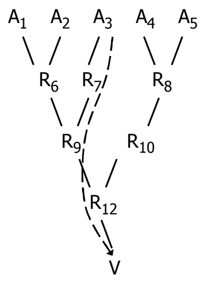
Address validation is insufficient

- Zombies
- One spoofable network allows attacks to happen

21

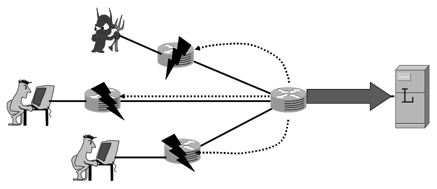
IP traceback is too little too late

- Does not prevent DoS
- Need to catch on the order of one thousand packets to construct a path
- Ineffective in dealing with DDos
  - Computationally expensive to construct attack paths



22

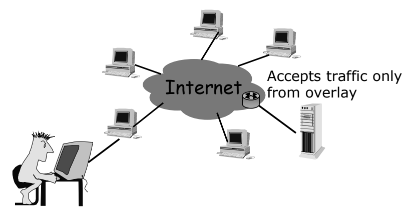
Pushback lacks discrimination



- Both legitimate sources and attackers suffer.

23

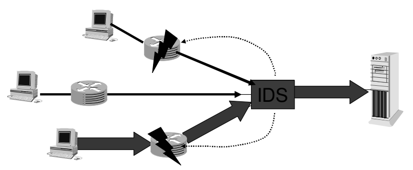
Overlay filtering only protects limited destinations



- Overlay nodes apply filters and authenticate clients.
- They need to know destination policies.
- Overlay nodes can be comprised and DDoSed.

24

Intrusion detection system (IDS) threatens openness



- Automated IDS alarms on unusual traffic
  - Clamps down on attacks and new applications alike
  - In the limit this leads to a closed system without innovation.

25

Our position

26

A clean architectural solution

- It’s time to rethink a basic premise – that anyone can send packets anywhere, any time. If not, the long-term consequence will be no security and no openness.
- We argue for a capability-based architecture that contains the damage of DDOS (security) yet allows applications to exchange any packets they want (openness)

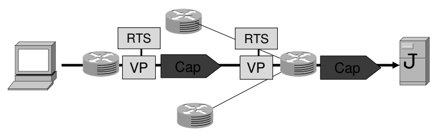
27

The Need for Capabilities

- Observe that:
  - Only destinations know which packets are legitimate
  - Only the network can shed load before it is excessive
- End result:
  - Network filtering must be based on destination control
  - Authorization needs to be explicit so it can be checked throughout the network, i.e., packets carry capabilities

28

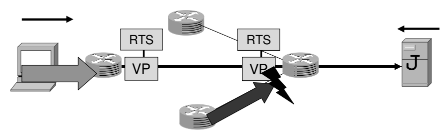
Key Idea: A Capability-based Internet



- Goal is to tightly contain the impact of attackers.
- Add two elements to the architecture:
  - Short-lived capabilities carried within packets
    - E.g. send 50 packets in the next 10 seconds
  - (Logical) boxes (Request-To-Send/Verification Point) filter packets based on capabilities

29

Sketch of Using Capabilities

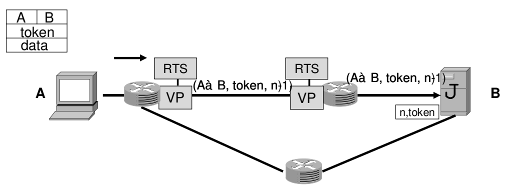


1. Source requests permission to send
2. Destination grants capabilities to authorize sending
3. Source places capabilities on packets and sends them
4. Network filters packets at VPs based on capabilities

Challenges

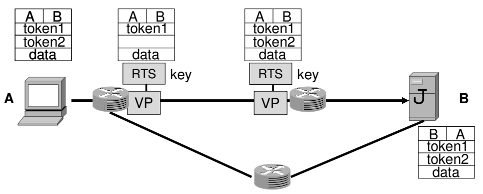
- How to securely and efficiently implement capabilities
  - Granted by destinations, checked by VPs
  - Must ensure an attacker cannot forge a capability
- How to protect the RTS channel
  - A sender can send RTS packets without permissions from a destination

Capability implementation: the stateful approach



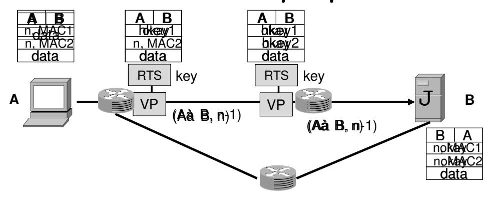
- Destination installs short-lived tokens at VPs
- Packets carry tokens
- Drawbacks
  - Unbounded flow state at VP
  - Token installation traverses reverse path

Capability Implementation: the stateless approach



- RTS stamps a short-lived token: hash(A,B)$_{key}$
- Destination returns the tokens
- Packets carry per-RTS tokens
- Drawbacks
  - Large header overhead
  - Destination cannot control the amount of traffic sent to it

Our current proposal



capability granting
1. RTS stamps a temporary key: hkey=hash(A,B)$_{key}$
2. Destination returns capabilities: (n, MAC(A,B,n)$_{hkey}$)

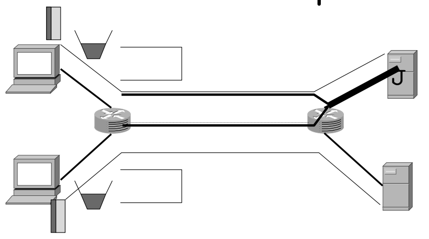
flow verification
3. Sender sends a packet with capabilities
4. VP stores flow state if it has memory

5. Sender sends packets without capabilities
6. Sender periodically renews capabilities

Protecting the RTS channel

- Easier problem than protecting the data channel
  - RTS channel has highly constrained usage
  - RTS channel has limited bandwidth

Our Current Proposal

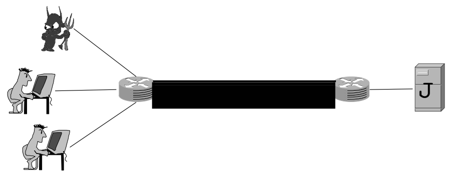


- Per "user" queuing
- Rate-limiting RTS traffic
- Authorized data traffic and rate-limited RTS traffic has a higher priority than legacy traffic

Policy Issues

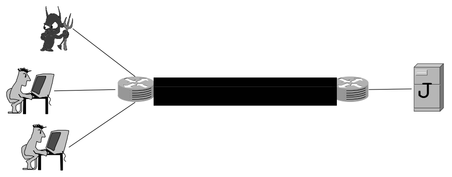
- Destinations decide whether or not to authorize
  - So what's the policy?
- Simple policies can be effective
  - We have policies today, e.g., firewalls
  - à View capabilities as programming the network-wide firewall
- Destinations can cut off misbehaving sources
  - Just don't renew capabilities

High-level of Protection



- Number of good users is N
- Number of Attackers is A
- With destination policy and capability
  - If the capacity of RTS channel > N + A; then
    - Good share of the bottleneck link ~ 100%

Best protection without destination policy



- Number of good users is N
- Number of Attackers is A
- Without destination policy
  - Best protection with per-user fair queuing: good share of bottleneck link is N/ (N+A)

Incremental deployment

- Incremental deployment brings incremental benefit
  - One RTS/VP deployed at an ISP suffices to filter out attack traffic for a customer
  - Traffic from a capability-enabled client has higher priority than legacy traffic
- A shim layer between IP and TCP
  - Backward compatible with legacy routers
- RTS's/VPs are bumps-in-the-wire

Conclusion

- It's time to rethink a basic Internet premise – that anyone can send to anywhere, any time – to improve security and preserve openness
- We argue for a new capability-based architecture:
  - Capabilities make destination authorization decisions explicit
  - Packets can be checked for permission throughout the network