Internet Denial-of-Service Attacks

Mark Handley

Part 1
The Denial-of-Service Problem

Denial-of-Service (DoS) Attacks

- A DoS attack is where one or more machines target a victim to prevent the victim doing useful work.
- Victim can be:
  - Network Server
  - Network Client
  - Router
  - Link
  - Entire network
  - Company
  - ISP
  - Country

Internet Architecture

- Original Internet was closed, relatively homogeneous community.
- Internet was *not* designed for attack
  - Contrary to popular opinion!
- Security has been retrofitted.
  - Encryption, authentication sort of work (when we bother to enable them).
  - DoS is the *hardest* form of attack to deal with.
- Almost all Internet services are vulnerable to DoS attacks of sufficient scale.

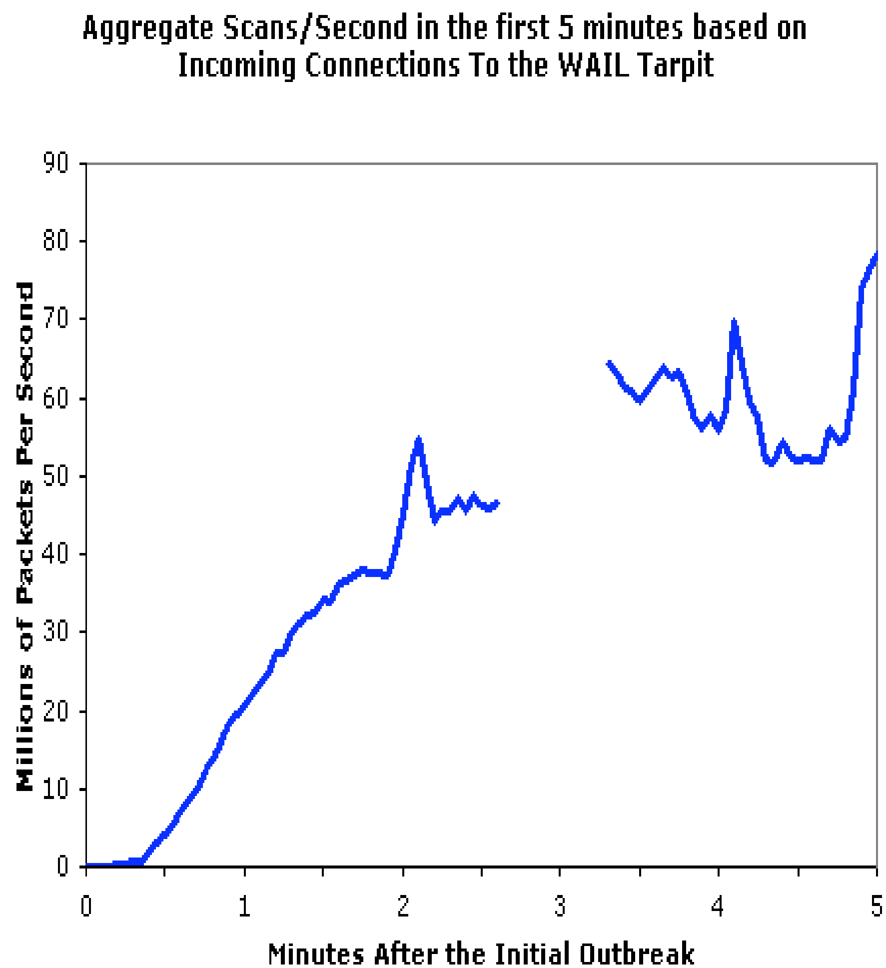
Sufficient scale?

- In many cases a victim can be disabled by a single attacking host.
  - A well-connected PC can source nearly 1Gb/s of (fairly dumb) attack traffic.
  - Few machines can sink 1Gb/s of attack traffic and do useful work if they have to process those packets in any significant way.
  - Few sites have 1Gb/s access links.
- In almost all cases sufficient scale can be achieved by compromising enough end hosts.
  - Worms, viruses, remote-controlled attack bots.
  - Use those compromised hosts to launch DoS attacks.
  - Attack networks of 10,000 hosts not so hard to create.

[Source: Nick Weaver, Silicon Defense]

Slammer Worm

- Infected ~75,000 machines in 10 minutes
- Full scanning rate in ~3 minutes
  - >55 Million IP addrs/sec
- Initial doubling rate was about every 8.5 seconds
  - Local saturations occur in <1 minute



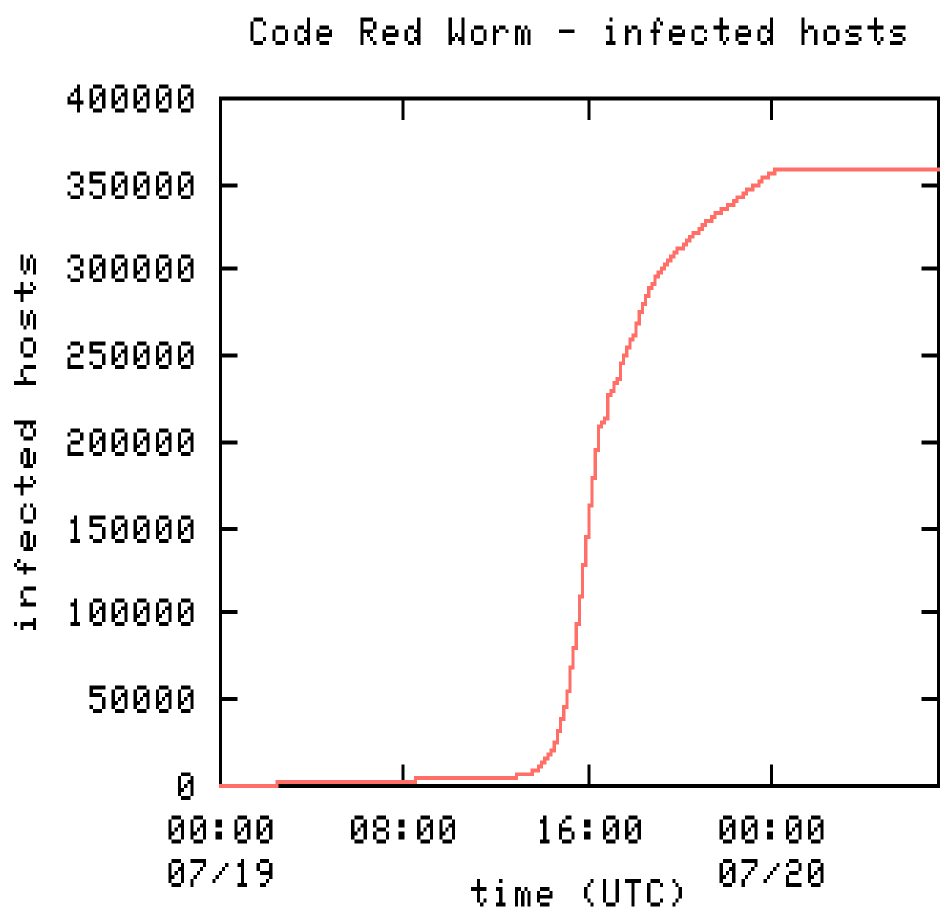
[Source: CAIDA]

Code Red Worm

Code Red required about 13 hours to spread worldwide

Other techniques can be even faster:

- Eg, “Warhol Worm” → 15 minutes
- Sapphire → 10 minutes



Flash Worm

- Use permutation scanning
- Use pre-computed hit-list of likely victims.
  - Realistic to infect every vulnerable host on the Internet less than 30 seconds after worm release.

- See "*How to 0wn the Internet in Your Spare Time*", S. Staniford V. Paxson, N. Weaver Proc. *11th USENIX Security Symposium, 2003*

DoS Attacks on End Systems [1]

- Exploit poor software quality.
  - Eg. *ping-of-death*
  - OS crashes when sent a fragmented ICMP echo request whose fragments totalled more than the 65535 bytes allowed in an IP packet.

- Not a serious architectural problem:
  - Once code is fixed, problem is solved.

DoS Attacks on End Systems [2]

- Application resource exhaustion:
  - Available memory
  - Available CPU cycles
  - Disk space
  - Number of processes or threads
  - Max number of simultaneous connections configured.
- Some resources are self-renewing.
  - Eg CPU cycles
- Some are not: effects persist after attack stops.

DoS Attacks on End Systems [2]

- TCP SYN flood
  - Essentially a memory exhaustion attack.
  - Victim instantiates state for half-open connections.
  - Exacerbated by IP source address spoofing.
- TCP ACK flood
  - Essentially a CPU exhaustion attack.
  - Busy server with many connections spends a lot of CPU cycles searching for the right TCB for these spoofed packets.

Notes on CPU Exhaustion

- Strong authentication mechanisms don’t prevent CPU exhaustion attacks.
  - Often the authentication mechanism itself is CPU intensive.
- Poors OS handling of network events can make things worse.
  - Livelock due to network interrupts.
  - OS should switch to polling network devices when busy.

Attacks on Ongoing Communications

- If an attacker can see the data traffic from a TCP connection, they can trivially reset the connection.
  - Transport or App. level security (SSL, ssh) doesn't help.

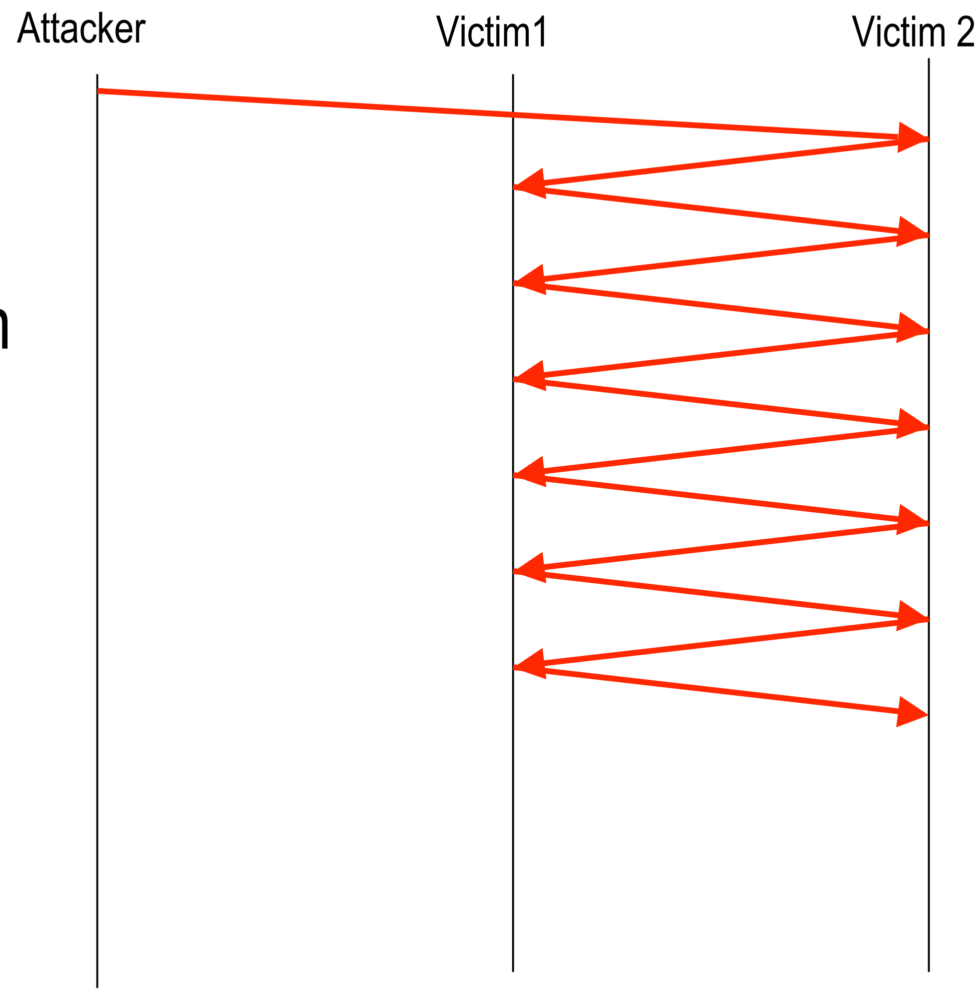
- Even if they can't see the traffic, they may be able to predict sequence numbers well enough to reset a connection whose existence they can deduce.
  - Eg. BGP peering.
  - May require a lot of packets.
  - Good initial sequence number randomization is critical.
  - At high speeds, TCP window is very large and attack becomes easy, even with randomized sequence numbers.

Use the victim's own resources

- Send packet to UDP echo port of victim 1.

  Spoof src address of victim 2, src port of victim 2's UDP chargen server.

  Victim 1 and 2 bounce packets back and forward DoSing each other.



Triggered Lockouts, Quota Exhaustion

- Some password mechanisms lock the victim out after a number of failed attempts.
  - Trivial DoS.
- Many services have quotas.
  - Eg. bandwidth quota for web hosting.
  - Exhaust quota, deny service until next accounting period.
  - In the absense of quotas, finanical DoS may be possible.

DoS Attacks on Routers

- Most end-host attacks work against router control processors.

DoS Attacks via Routing Protocols

- Overload routing table with lots of spoofed routes.
  - Too much memory required.
  - BGP has very poor overload semantics.
- Attack destination by announcing better route.
- Cause routing churn, cause BGP route-flap damping to suppress victim's routes for significant time.
- Cause routing loop, cause traffic to loop overloading links.
- Probably many more.

DoS Attacks via IP Multicast

Ramen worm:

- Poorly written randomized address scanner.
  - Didn’t notice that class D addresses were multicast.
- Caused many multicast routers to instantiate state for all these new sources to all these new multicast groups.
  - Particularly MSDP, but also PIM-SM.
  - Big multicast meltdown.

- Basically ASM (any-source multicast) model is fatally vulnerable to DoS.

DoS Attacks via SSM Multicast

- Vulnerabilities much less than ASM.
  - Stateholding attacks on routers.
  - Bandwidth DoS on links leading to attacker (self-DoS).

- Sender-based attacks are not possible.
  - Receiver needs to request traffic.
  - Source-address spoofing is hard because of multicast RPF checks needed for tree-building.

Attacks on Router Forwarding Engines.

- Two forms of forwarding engine:
  - Use a forwarding *cache*
  - Have all routes in forwarding engine.
- Forwarding caches are vulnerable to thrashing attacks, or memory exhaustion attacks if they can't hold the whole routing table.

- May be possible to overload the comms between the forwarding engine and router control processor.
  - Unpredictable results.

Local DoS Attacks

- Exhaust DHCP address pool
- Respond faster than DHCP server
- ARP spoofing
- Broadcast storms
- 802.11:
  - Spoof basestation.
  - Exhaust basestation association pool
  - Deauthenticate or disassociate victim (even with WEP!)

Common theme: *robust autoconfiguation is very hard.*

DoS Attacks via DNS

- No-one knows IP addresses.
  - Deny DNS, deny access to the site.
- Anti-spam measures require DNS lookup of From address in email.
  - Deny DNS, cause *outgoing* email to be dropped.
- DNS cache poisoning.
  - If a DNS server will relay for an attacker, the attacker can (with high probability) insert anything they want into the DNS server's cache.

DoS Attacks on Links

- Bandwidth exhaustion.
  - Simple congestion attack on traffic.
- Congestion may cause routing packets to be lost.
  - Cause routing adjacency to be dropped.
  - 100% packet loss if no alternative path.
  - Route flap if alternative path exists (BGP flap damping!)

DoS Attacks on Firewalls.

- Similar to end-system attacks.
  - Exhaust memory in stateful firewalls.
  - Cause CPU exhaustion.

- CPU exhaustion isn’t so easy if the firewall is simple.
  - Possible computational complexity attack with pathelogical traffic.
  - Cause hash-table performance to go from O(1) to O(n) by causing the f/w to instantiate state for n flows that all lands in the same hash bucket.

Spam and Black-hole Lists

- All spam is a DoS attack on email users.
- All spam-filtering is a DoS attack on spam!
  - The borderline between spam and legitimate email is narrow and fuzzy.
- All too easy to get someone put in some of the less selective black-hole lists.
  - Really hard for them to prove their innocence and get removed.
- May be possible to train a victim's adaptive spam filters so that they drop selected legitimate messages.

Externalities

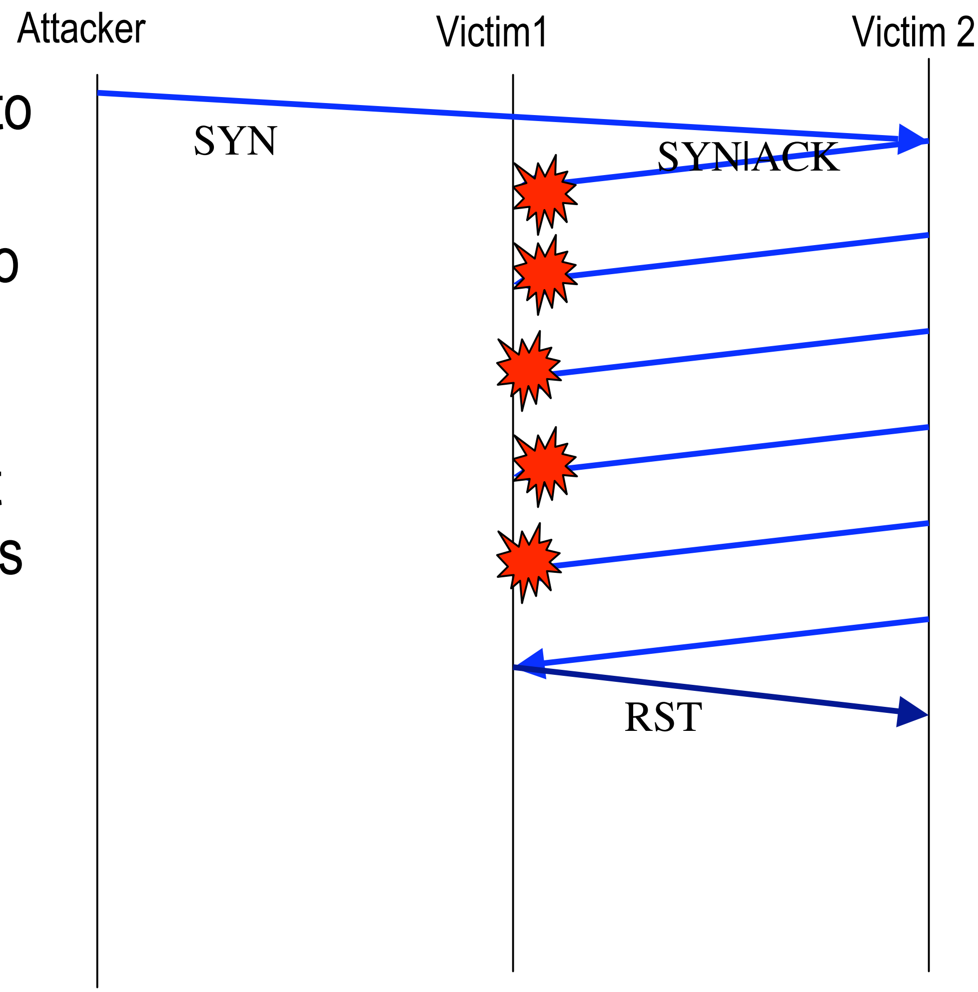
- Physical DoS
  - Power, cables, etc.
- Social Engineering DoS
  - Convince an employee to make a detrimental change.
- Legal DoS
  - Cease-and-desist letters, etc.

Attack Amplifiers [1]

- smurf attack
  - Spoofed ICMP echo request to subnet broadcast addr.
  - All hosts on subnet respond to victim
- DNS reflection.
  - Spoof DNS request.
  - Large DNS response goes back to victim.

Attack Amplifiers [2]

- bang.c
  - Send spoofed TCP SYN to arbitrary server.
  - Server sends SYN|ACK to victim.
  - Server retransmits the SYN|ACK many times if it gets no response (such as if the victim is overloaded and dropping lots of packets).



Lessons [1]

- Don’t create an attack amplifier.
  - Small responses to requests from unverified hosts.
  - RTX in initial handshake performed by client only.
  - Perform ingress filtering to prevent spoofing.
- Don’t hold state for unverified hosts
  - Or at least be able to not hold this state.
- Take care regarding state-lookup complexity
  - The attacker may control the state.
- Avoid livelock
- Use unpredictable values for session IDs.

Lessons [2]

- Authenticate routing adjacencies
  - Perhaps the only place for strong auth in the DoS space
- Isolate router-to-router traffic.
- Engineer graceful routing degradation.
- Use source-specific multicast.
  - ASM is dead. Get over it.
- Autoconfiguration is really hard.
- Establish a monitoring framework.
  - When you're being attacked, it's too late to figure out what normal traffic looks like.

draft-iab-dos-00.txt

Internet Denial of Service Considerations, Jan 2004, Internet Architecture Board, Mark Handley (editor)

Part 2:
Musings on DoS Resistant Internet Architectures

Simple idea

- Divide address space into client addresses and server addresses.
  - Client address can't send to a client address.
  - Server address can't send to a server address.

- Note: some hosts may need both, but most hosts don't need both to be globally routable.
  - Peer-to-peer is a problem.

Separate Client and Server Address Spaces

Advantages:

- Reduces threat from worms.
  - Must travel client -> server -> client
  - Requires two vulnerabilities.
  - Server -> client is a slow process (contagion).
  - honeypots can detect client -> server phase.

- bang.c, smurf (and similar) not possible or severely limited.
- Reflection attacks on servers prevented.

Client Addresses

- Client addresses don't need to have any global significance.
- Can use a concatenation of local IDs that is constructed as packet travels from client to server.
  - Sufficient to route packets back to client.

Path-based Client Addresses

- Clients are protected from DoS attack.
  - Except from someone they initiate connections to.
  - Assuming an attacker can't figure out how to piece together a path from their server address to a passive client.
- Source-spoofing is extremely limited.
  - Provides a solid basis for pushback mechanisms.
- Prevents all reflection attacks against remote targets.

State Setup Bit

- Packets that set up communication state (especially connection setup) need to set a ***state-setup*** header bit.
  - Generic protocol-independent way of identifying packets that need validation.
  - Packets without this bit can be dropped by stateful middleboxes (firewalls) if state doesn't exist.
  - Server addresses cannot send such packets.
- Introduce a generic ***nonce request/response mechanism*** that can be used to verify an IP address.
  - Middleboxes or end-systems can use this when they receive a state-setup packet (without instantiating state).
- Rate limit state setup packets from each client.

Pushback

- Add a **pushback** mechanism to throttle traffic from an attacker to an overloaded server (or link to a server).
  - Non-global client addresses make this hard to use to attack a client.
  - Limited ability to spoof client addresses means this can pushback most of the way to the attacker.

Redirect

- Need a cheap stateless way to redirect a client to an alternative server.
  - After accepting the TCP connection is too late.
- **Generic IP-level redirect message**?
  - Perhaps delegate the sending of such messages to a firewall to load-balance when heavily loaded.
  - Allows on-demand mirroring to a third-party (probably commercial) server when unusual load experienced.

DoS Resistant Multicast

Remaining problem with SSM is clients joining too many groups and causing stateholding attack on routers. Possible solutions:

- **Crytographically generated addresses** with IPv6.
  - Only sender can generate a valid multicast addr but routers can verify. Somewhat expensive to check though.
- **Active group announcement channels.**
  - Each unicast route has associated with it an announcement channel.
  - Lists all source/group pairs active in that domain.
  - Router receives a Join msg for (S,G) and joins the corresponding announcement channel. Only forwards join if (S,G) is announced.
- In any event, only server addresses can send, only clients can receive multicast.

DNS

- Internet is critically dependent on DNS.
- The core of DNS cannot be secured against DoS attacks of sufficienly large scope.
  - Anycast DNS helps, but not sufficent.
- General idea:
  - Multicast all the TLDs and SLDs (signed by a trusted key).
  - Local DNS servers receive this data and cache it.
  - No request/response at all in the core.
    - Still needed at the edge though.

Assymetric Costs

- General strategy is to allow the server to make it expensive for the client to make a request.
  - Eg. CPU puzzles.

- Again, need a way to indicate to the client what they have to do to be served before the server wastes CPU cycles or state.
  - Perhaps add to nonce-echo request?
  - Perhaps advertise in routing?

Observations

- In such a world, servers are *more expensive* for ISPs to support than are clients.
  - clients are largely invulnerable to unsolicited attack.
  - servers are advertised as available, so attract incoming requests.
- Probably this is true today, but the distinction isn't clear.

- Likely implication: connecting a client is cheap, connecting a server is expensive.
  - Some ISPs charge this way today, but for business rather than technological reasons.

- However, servers cannot perpetrate attacks, so the followup costs for an ISP may be cheaper. Economics really unclear here.

Limitations

- A very distributed (> 1M attacking hosts) DoS attack is still very hard to defend against.
  - Lots of state required to pushback towards all of them.
- Link-saturation DDoS attacks on core links hard to defend against.
  - No common destination address for pushback.
- Routing protocols still vulnerable.
- In principle, a victim can't tell the difference between a flash crowd and a DoS attack.
  - Pushback only useful if you can identify good from bad.
  - Goal should be to minimize collateral damage.