Towards DoS-resistant Internet Architecture

Prof. Mark Handley
University College London

Denial-of-Service

- Attacker attempts to prevent the victim from doing any useful work.
  - Flooding Attacks
  - Exploiting Software Weaknesses
- Flooding Attack:
  - Send sufficient traffic to overload network link, router, host, firewall, or any other Internet system.
  - Limited resource can be link capacity, CPU, memory, disk space, quota, or pretty much any other consumable.

Dealing with Flooding

1. Detect flooding attack
2. Ask the network to stop sending you the bad traffic.
3. Attacker's ISP disconnects them.

Complicating Factors

- Source addresses in packets may be spoofed (unless the attack requires a full TCP connection).
  - Hard to tell the real source of the attack, especially if there are thousands of real sources.
  - Need to be careful not to shut down legitimate traffic in response to a spoofed attack.
- Network paths are normally asymmetric.
  - This is a property of destination-based routing.
  - Also “necessary” for normal economics of ISP peering.

Complicating Factors

- Network moves packets; has no knowledge of connections, flows or applications.
  - This is normally a good thing.
- Attack can be at any level:
  - Flooding a network link with packets.
  - Flooding a server with packets.
  - Flooding a server with connections.
  - Flooding a server with higher-level requests requiring more work.

Complicating Factors

- Some attacks are obviously malicious:
  - TCP SYN flood.
  - DNS reflection attack.
- Some attacks are very hard to distinguish from a flash crowd:
  - HTTP-level attack on web server, with a few requests coming from each of many hundreds of thousands of hosts.
  - *Slashdot* or DDoS?
- Generally speaking, the lower level the attack, the more obvious it is that it is actually an attack.
- Higher-level attacks usually require a full connection, so can't be spoofed. Might not be malicious though.

Threat Model

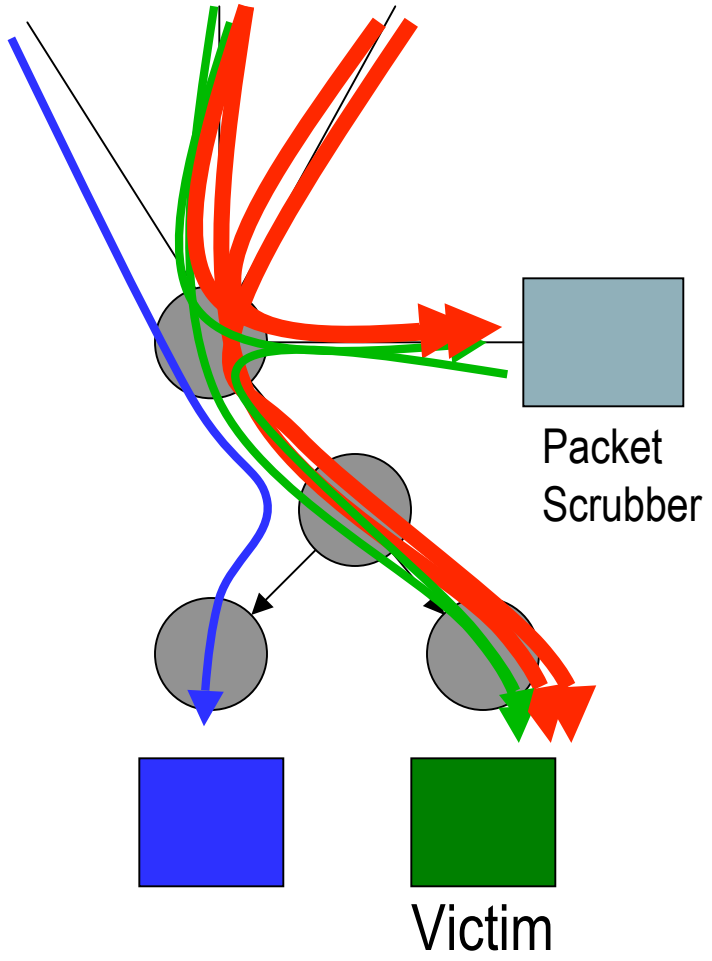
- Thousands of machines compromised:
  - Rapidly spreading worms
  - Automated scanning by bots
  - Viruses
- Compromised machines used for distributed DoS attacks:
  - Attack traffic can total many gigabits/second.
- Source-address spoofing.
  - Actually not very common because not necessary.
- Reflection attacks
  - Serve as amplifiers
  - Obfuscate attack origin.

Current “Solution”

- Very ad-hoc.
- Mostly manual intervention by ISPs to blackhole specific traffic.
- Some deployment of “scrubbers”.
  - When under attack, re-route traffic via scrubbers which use deep-packet inspection to pass traffic they believe to be good.
  - Usually scrubber located in last-hop ISP; occasionally as a third-party service.
- No way to shut down hostile traffic near to its source, automatically trace spoofed traffic, or even to automatically inform the source ISP.

Packet Scrubbers: Distinguish “good” from “bad” traffic

1. Detect
2. Activate: Auto/Manual
3. Divert only victim’s traffic
4. Filter only DoS traffic



Towards evolvable solutions

- It is important that in providing short-term solutions we don't sacrifice the future evolution of the Internet.
- Anything that goes in the middle of the network should be:
  - Application independent.
  - Impose minimal dependencies on transport protocols.

Architectural Approaches to DoS Defense

- Ask for permission to send before sending.
- Charge for congestion.
- New addressing and routing models.
- Control points and receiver-initiated filters.

Approach 1:
Request Permission to Send

- Require receiver to allocate a “token” to send before sender can send actual data.
  - Similar to legacy phone network.

- Needs a request channel:
  - But can DoS the request channel.
  - Easier to defend request channel - semantics are more well defined.
  - Requires enforcement of tokens in network core.
- Substantial change from current Internet architecture.
- May fit poorly with apps that aren’t inherently flow-based (such as DNS)

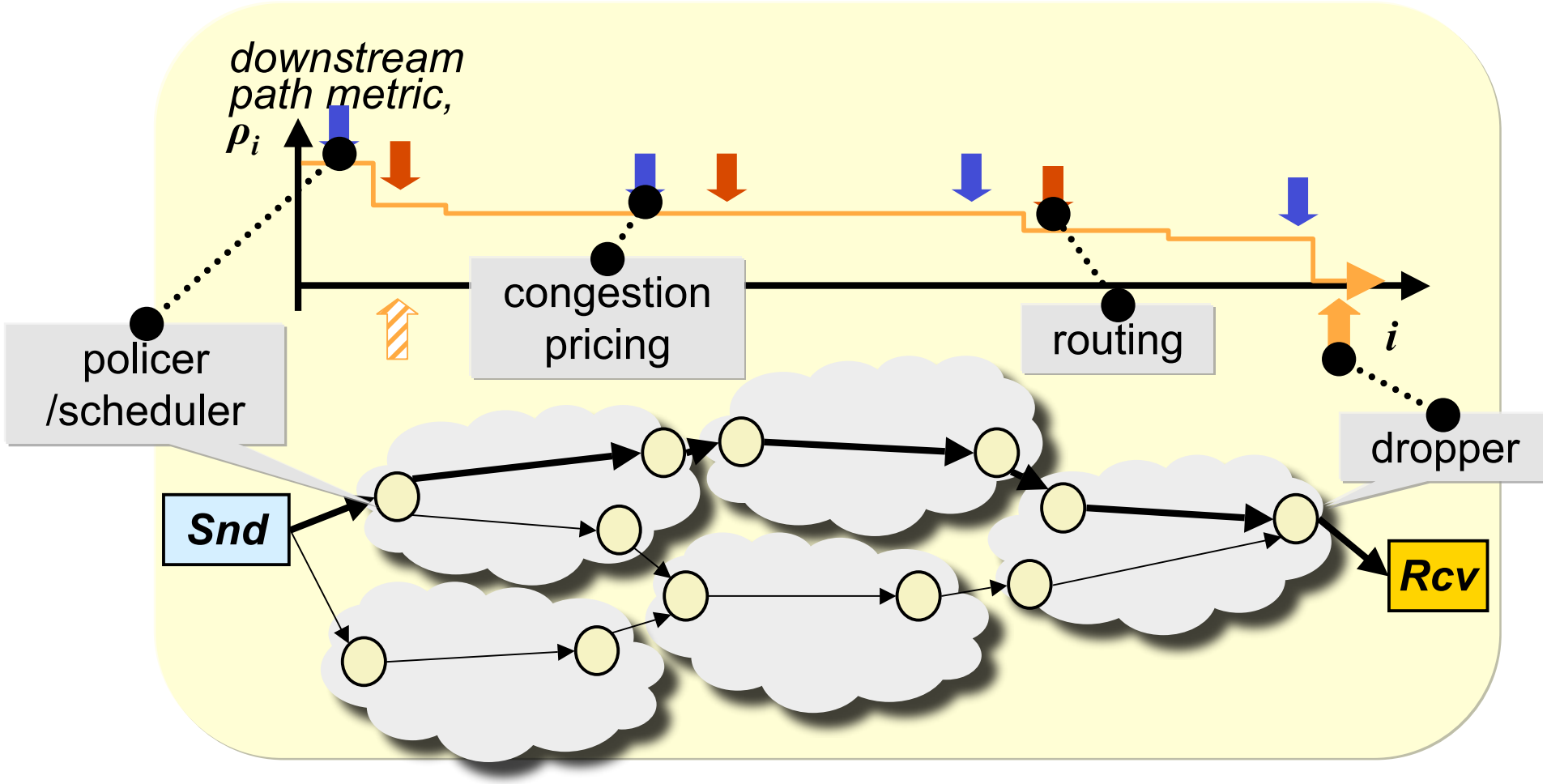
Approach 2:
Charging for Congestion

- Pass the cost of the attack back towards the sender.
  - Need some way to verify that cost was incurred.
  - See Briscoe's *re-feedback* for one way to do this.

- Probably can't charge the actual sender.
  - Granny's PC catches a virus and clocks up a huge bill.

- May be able to push the cost to origin ISP.
  - Provides a strong motivation for them to police their customers.

Re-feedback incentive framework



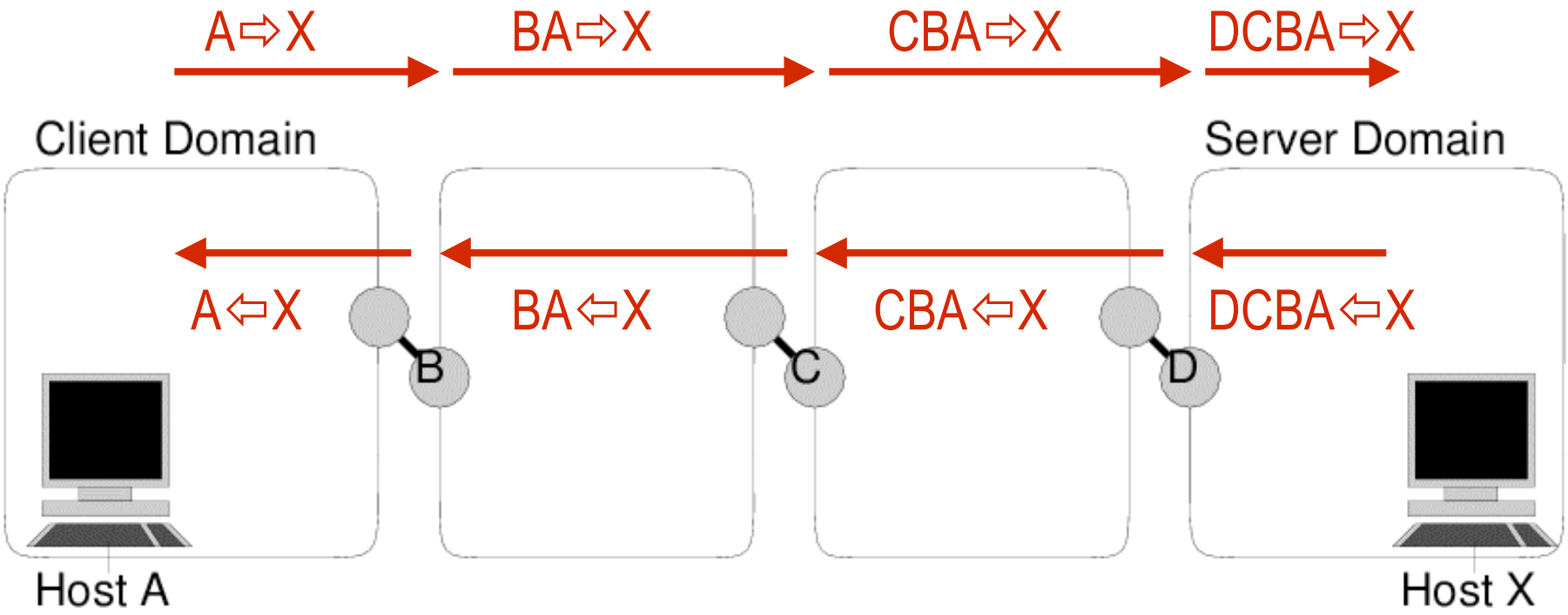
Approach 3:
New Addressing and Routing Models

Source-address spoofing is a significant part of the DoS problem.

Asymmetric routing makes it harder to identify anomalous traffic patterns.

- Both could be tackled by new addressing or routing schemes.
  - Use path-based addresses for client-addresses that do not need to be globally unique.
  - Use source-routed paths, so that the destination can read the path from each packet.
- Such changes would greatly ease tracing and shutting down attacks, and prevent reflection attacks.
- Very significant change to the architecture - unlikely to be deployed.

Path-based Addressing



Approach 4:

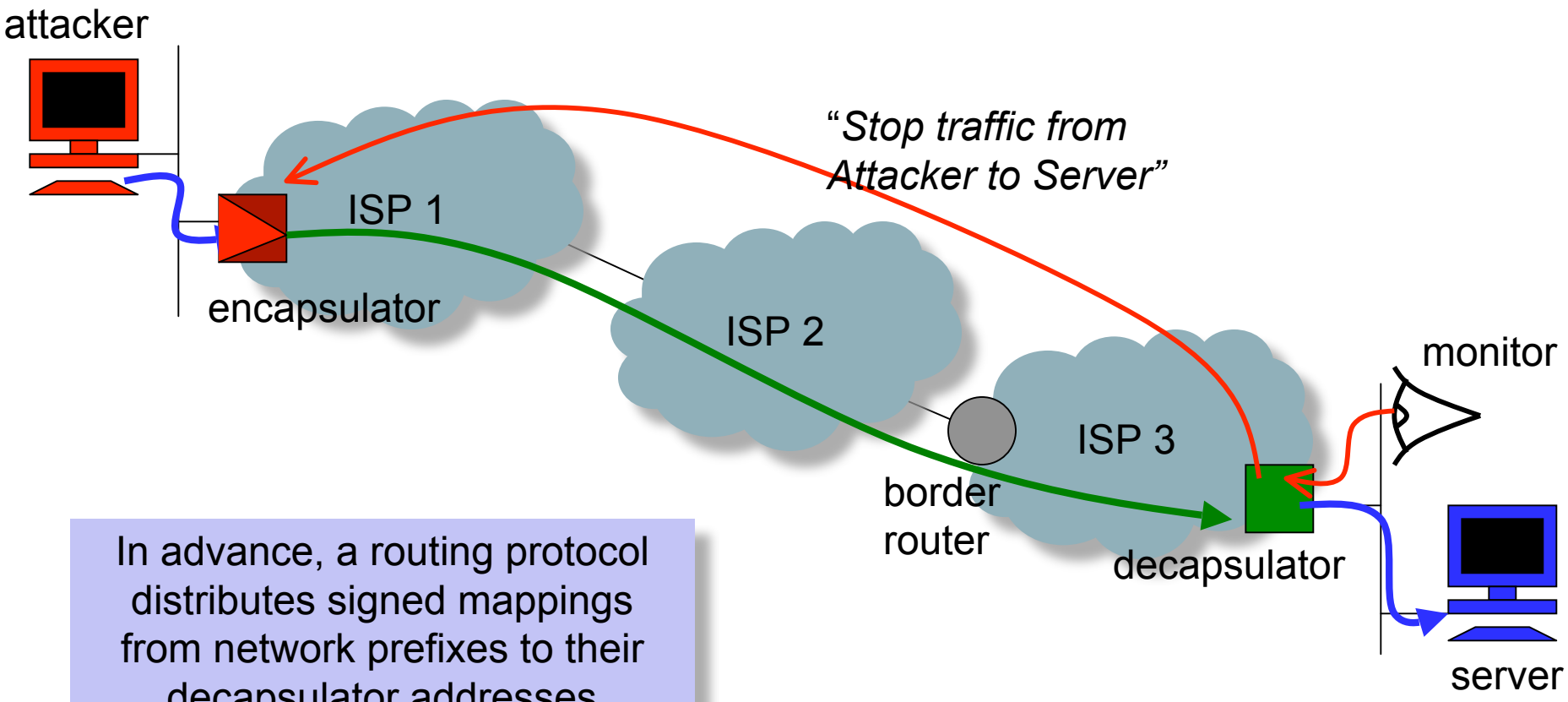
Control Points and Receiver-initiated Filters

Basic idea:

1. *Route traffic through control points in the network.*
2. *Control point encapsulates traffic and forwards it on to a decapsulator close to the receiver.*
3. *Encapsulation header serves to identify control point.*
4. *Receiver can request traffic destined to it to be blocked at the control point.*

- Various different instantiations are possible, depending on where the control points are located.
- Depends on receiver being able to decide which traffic it doesn't want.
  - Simple for lower-layer attacks, harder for higher-level attacks.

Edge-to-edge Encapsulation



In advance, a routing protocol distributes signed mappings from network prefixes to their decapsulator addresses

Summary

- A range of architectural solutions could make a big difference to the difficulty of dealing with DoS attacks.
  - Most suffer from serious deployment hurdles.
- One exception might be Edge-to-edge Tunnelling.
  - Simple, low cost, incrementally deployable.
  - Incentives reasonably well aligned.
  - Not a 100% solution, but pretty good.
- Really need a combination of technical and non-technical means.