Network Security (and related topics)

EE122 Fall 2012

Scott Shenker

http://inst.eecs.berkeley.edu/~ee122/

Materials with thanks to Jennifer Rexford, Ion Stoica, Vern Paxson and other colleagues at Princeton and UC Berkeley

Next Week

- No sections
- No lecture on Thursday
- On Tuesday we will talk about SDN

Agenda

- Project 3 Q/A (10)
- Network Security (20)
- Dealing with Persistent Route Failures (20) [Colin]
- More network security (15)
- Datacenter Congestion Control (15, if time)

Project 3 Q/A

- Last chance to grill Panda…..

Network Security

narrowly defined….

My definition of “network security”

- “network security” ≠ “security in a connected world”
  - For the latter, take CS 161 (*spectacular course!*)
- If network magically transfers data between known parties, there is no “network security” problem
- There are *many* other security problems
  - Distributed system (if A lies to B, does system crash?)
  - Operating system (Can A’s system be compromised?)
  - …
- But these may not require network solutions

Examples: Non-network security issues

- Browser “drive-by” exploits
- Server vulnerabilities
- Spam
- Phishing
- Account theft
- ….

Two Kinds of Network Security Goals

- **Core concern**: accomplishing communication
  - *Getting the data from A to B intact*
  - *Knowing it was from intended party, to intended party*

- **Also**: Keeping bystanders as ignorant as possible
  - *Making sure C, D, etc. don't know what A and B did*

Core Security Requirements

- **Availability**: Will the network deliver data?
- **Authentication**: Who is sending me data?
- **Integrity**: Do messages arrive in original form?
- **Provenance**: Who is responsible for this data?
  - *Not who sent the data, but who created it*
  - Important because communication may not be directly between actors, but through intermediaries

Keeping Bystanders Ignorant

- **Privacy**: can others read data I send?
- **Anonymity**: can I avoid revealing my identity?
- **Freedom from traffic analysis**: can someone tell when I am sending and to whom?
- *Today, will ignore latter two and focus on privacy*
- *But first, how would **you** achieve these two goals?*
  - *Assume all the crypto you want….*

Back to other goals

- Availability
- Authentication
- Integrity
- Provenance
- Privacy

Public Key Crypto Provides

- Way to authenticate yourself: **signature**
- Way to ensure privacy: **encryption**
  - with rcvr's public key
- Way to verify integrity: **hash function** (or MAC)
- Way to verify provenance: **signature**
- *In short, crypto provides all but availability!*
  - *Will return to availability later, focus on crypto for now*

On Cryptography and Identities

Crypto is about algorithms…..

- …algorithms that enable or prevent certain actions
  - *Enable* authentication and provenance
  - *Prevent* eavesdropping and undetectable tampering
- But security also requires tying actions to identities
  - Who is contacting me?
- And identities are not purely algorithmic

Three Aspects of Identities

- **Real-world identities (RWI)**
  - This is who you are in the real world
  - RWI established by social interactions
    - Direct experience
    - Referrals from friends
    - ….
- **Names**
  - Used in network protocols (e.g., DNS, URLs)
- **Keys**
  - Used by crypto

Security requires binding all three…

- Protocols: to ensure that they are interacting with appropriate entity, ***name must be bound to key***
  - *When accessing CNN.com, I need to know CNN's key in order to make sure that I'm not being spoofed*
- Humans: to ensure that they are interacting with appropriate entity, ***name must be bound to RWI***
  - *I need to know that CNN.com is the news organization based in Atlanta, not the Canadian Numismatic Network*
- *Once names are bound to both keys and RWI*
  - *Then keys and RWI are indirectly bound together*

Current Approach

- Google, human interactions: bind RWI to names
  - Works pretty well when you start with RWI and find name
  - Works less well when presented with name…
    - …and you are left to guess the RWI (phishing!)
- Certificate authorities bind names to keys
  - Binding is done via digital certificates
  - This does not work well…

The evolution of a cynic….

- "*Commercial certificate authorities protect you from anyone from whom they are unwilling to take money.*"
  - Matt Blaze 2001

- "*A decade ago, I observed that commercial certificate authorities protect you from whom they are unwilling to take money. That turns out to be wrong; they don't even do that much.*"
  - Matt Blaze 2010

Deeper problem with this approach

- Network: needs binding between names and key
  - Fetches data based on name
  - Authenticates based on keys
- Human: needs binding between RWI and name
  - Human makes decisions based on RWI
  - Humans must be involved in anything concerning RWI
- Current approach requires external authority to make the binding the network needs
  - Ties network infrastructure to external authorities

An Alternative Approach

- Use self-certifying names
  - Make your name the hash of your public key
  - Then the binding between names and keys is inherent
  - The network need not turn to external authorities

- Binding between RWI and names is flexible
  - Requires human-level interactions and judgements
    - How do I decide a name represents my brother?
    - Does same mechanism give name representing Barack Obama?
  - Already done reasonably well by Google, etc.
  - But is independent of low-level network mechanisms
    - So it can evolve!
    - Different people can use different mechanisms

Trust vs Identity

- Knowing who you are dealing with is different than trusting them
- Trust is a completely different concept, that should lie outside the architecture
- We often refer to mechanisms that bind names to keys or RWIs as “trust” mechanisms
  - Terrible terminology

Back to security goals

- Availability
- ~~Authentication~~
- ~~Integrity~~
- ~~Provenance~~
- ~~Privacy~~

Protecting Availability

How can availability be harmed?

- Problems in basic protocols
  - Persistent outages due to natural events (Colin)
- External vulnerabilities in basic protocols
  - Attackers can prevent protocols from functioning
- Internal vulnerabilities in basic protocols
  - If attackers compromise routers, can prevent network from functioning
- Denial-of-service attacks
  - Overwhelming the data plane with traffic

Colin will present recent research…

How can availability be harmed?

- ~~Problems in basic protocols~~
  - ~~Persistent outages due to natural failures (Colin)~~
- External vulnerabilities in basic protocols
  - Attackers can prevent protocols from functioning
- Internal vulnerabilities in basic protocols
  - If attackers compromise routers, can prevent network from functioning
- Denial-of-service attacks
  - Overwhelming the data plane with traffic

Examples of external vulnerabilities

- TCP:
  - Spoofing RST: requires knowing port/seq. no
  - Spoofing data: requires knowing port/seq. no
  - Cheating CC: reducing available bandwidth
- DHCP:
  - Spoof DHCP: can set host's DNS server and gateway
    - See all a host's traffic
    - Redirect connections to site's of your choosing
- DNS:
  - Cache poisoning

Note: semantics can be guarded

- If crypto is used everywhere (which it isn't), then you can always prevent hosts from being fooled
- But you can't prevent them from wasting time with incorrect accesses, etc., and thereby not getting the data they want in a timely fashion….

How can availability be harmed?

- ~~Problems in basic protocols~~
  - ~~Persistent outages due to natural failures (Colin)~~
- ~~External vulnerabilities in basic protocols~~
  - ~~Attackers can prevent protocols from functioning~~
- Internal vulnerabilities in basic protocols
  - If attackers compromise routers, can prevent network from functioning
- Denial-of-service attacks
  - Overwhelming the data plane with traffic

Example of internal vulnerability: BGP

- Why Google Went Offline Today and a Bit about How the Internet Works (November 6, 2012)

- “Someone at Moratel likely "fat fingered" an Internet route. PCCW, who was Moratel's upstream provider, trusted the routes Moratel was sending to them. And, quickly, the bad routes spread. It is unlikely this was malicious, but rather a misconfiguaration or an error evidencing some of the failings in the BGP Trust model.”

BGP: Naïve Trust Model

- BGP assumes routes are valid
  - Even when they are clearly not!
- How could we fix this?
  - Based on what we have discussed today

Solution to BGP Problem

- Bind prefixes to ASes
  - Registry of some kind
- Bind keys to ASes
  - Using certificate authorities
- Each route announcement must have signatures for each step (including originating prefix)

Easier solution

- Have BGP route on AS names
- Have AS names be self-certifying
- No external binding needed!

How can availability be harmed?

- ~~Problems in basic protocols~~
  - ~~Persistent outages due to natural failures (Colin)~~
- ~~External vulnerabilities in basic protocols~~
  - ~~Attackers can prevent protocols from functioning~~
- ~~Internal vulnerabilities in basic protocols~~
  - ~~If attackers compromise routers, can prevent network from functioning~~
- Denial-of-service attacks
  - Overwhelming the data plane with traffic

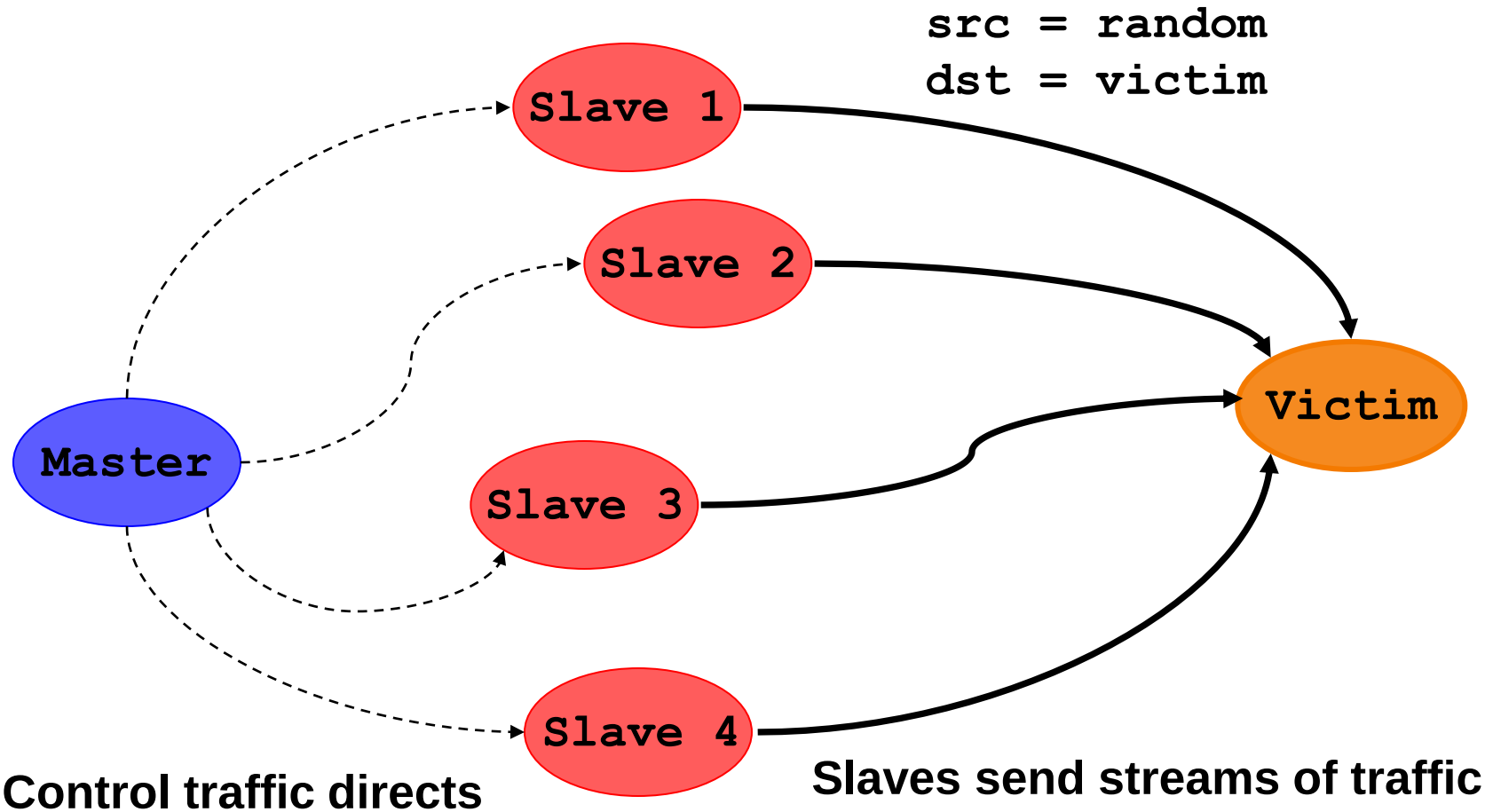
Denial of Service (DoS)

- Attacker prevents legitimate users from using something (network, server)
- Motives?
  - Retaliation
  - Extortion (e.g., betting sites just before big matches)
  - Commercial advantage (disable your competitor)
  - Cripple defenses (e.g., firewall) to enable broader attack
- Often done via some form of **flooding**
- Can be done at different semantic levels
  - Network: clog a link or router with a huge rate of packets
  - Transport: overwhelm victim's ability to handle connections
  - Application: overwhelm victim's ability to handle requests

DoS: *Network* Flooding

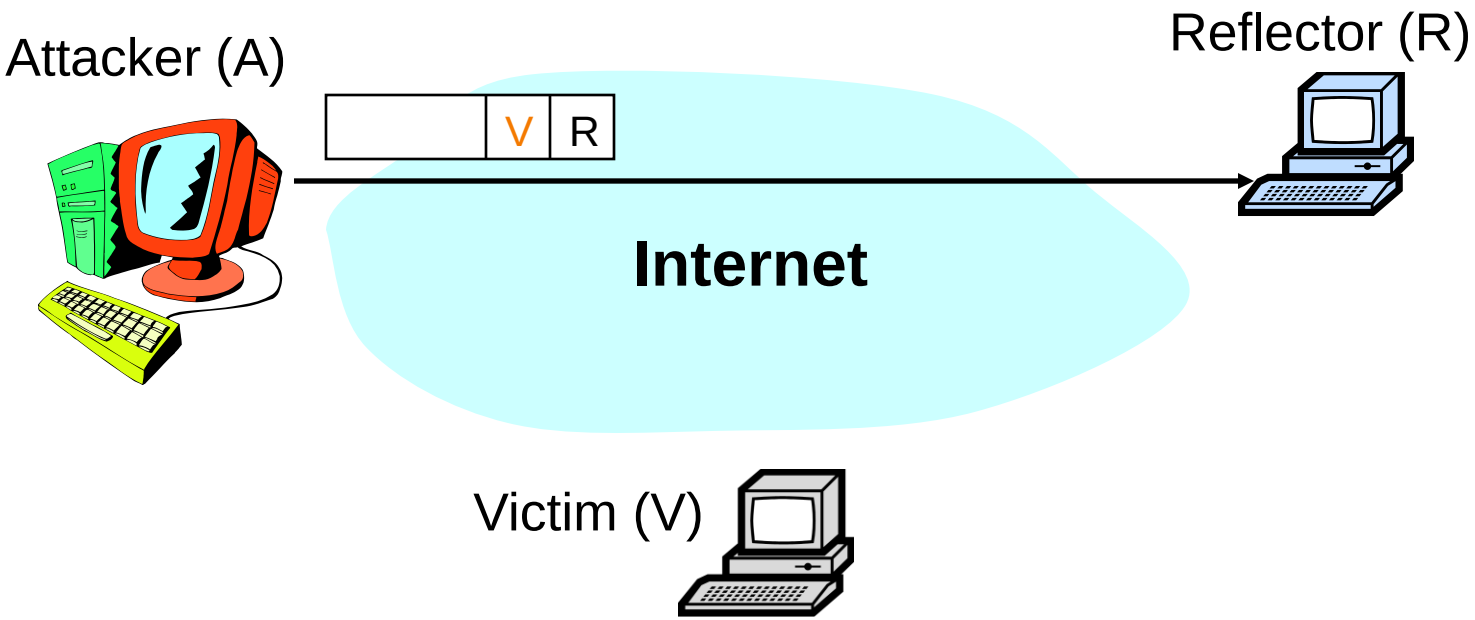
- Goal is to clog network link(s) leading to victim
  - Either fill the link, or overwhelm their routers
  - Users can't access victim server due to congestion
- Attacker sends traffic to victim as fast as possible
  - It will often use (many) spoofed source addresses …
- Using multiple hosts (*slaves,* or *zombies*) yields a *Distributed Denial-of-Service* attack, aka **DDoS**
- Traffic is *varied* (sources, destinations, ports, length) so no simple filter matches it
- If attacker has enough slaves, *often doesn't need to spoof* - victim can't shut them down anyway! :-(

Distributed Denial-of-Service (DDoS)



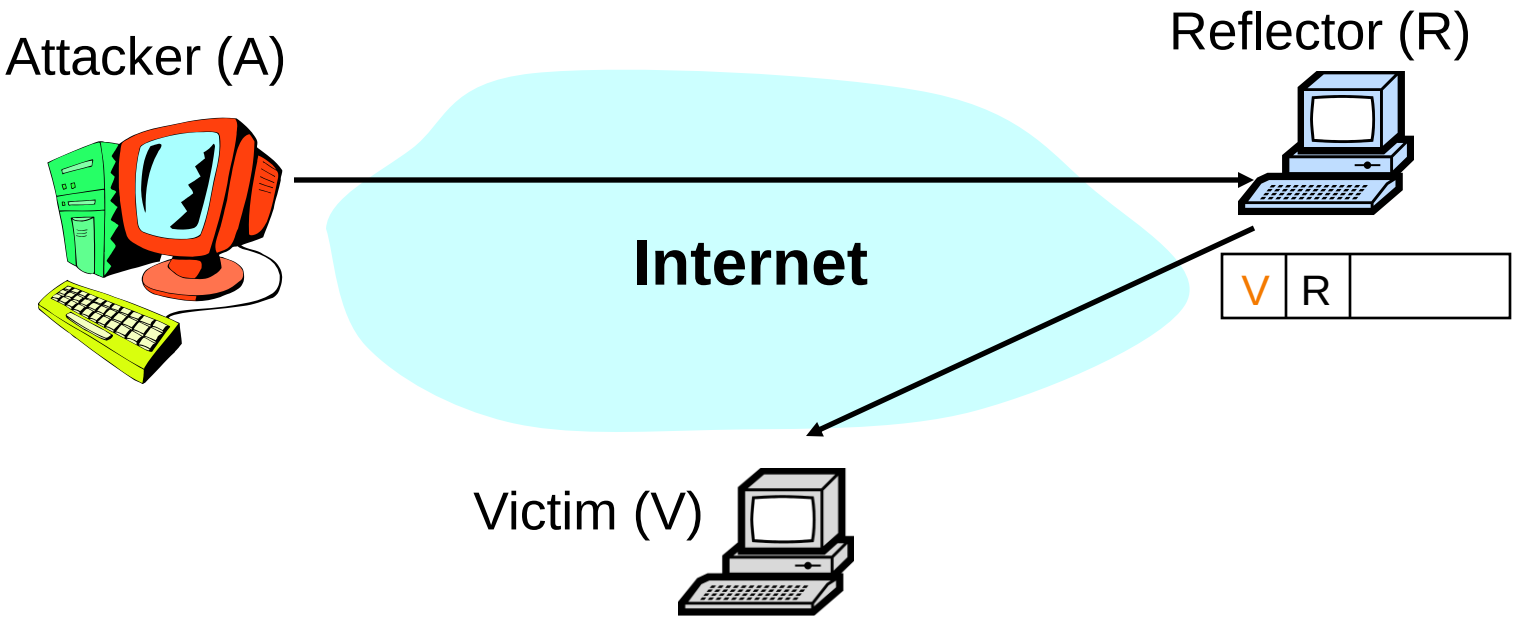
Very Nasty DoS Attack: Reflectors

- *Reflection*
  - Cause one *non-compromised* host to help flood another
  - E.g., host A sends DNS request or TCP SYN with source V to server R.

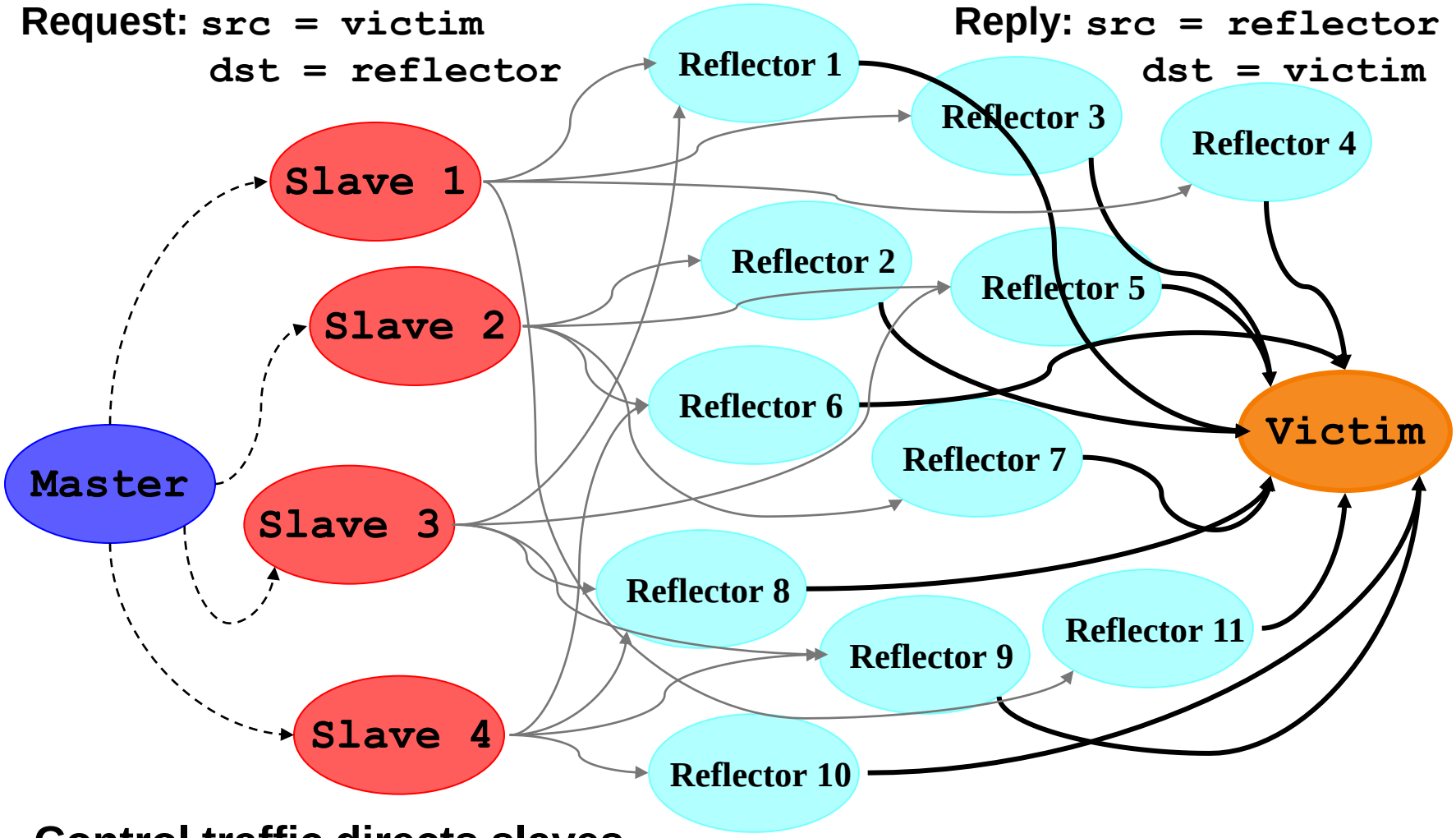


Very Nasty DoS Attack: Reflectors

- Reflection
  - Cause one *non-compromised* host to attack another
  - E.g., host A sends DNS request or TCP SYN with source V to server R.
  - R sends reply to V



Diffuse DDoS: Reflector Attack



Control traffic directs slaves at victim & reflectors

Reflectors send streams of non-spoofed but unsolicited traffic to victim

Defending Against Network Flooding

- How do we defend against such floods?
- Answer: basically, we don't! **Big** problem today!
- Techniques exist to trace spoofed traffic back to origins, but this isn't useful in face of a large attack
- Techniques exist to filter traffic, but a well-designed flooding stream defies stateless filtering
- Best solutions to date:
  - **Overprovision** - have enough raw capacity that it's hard to flood your links
    - Largest confirmed botnet to date: 1.5 million hosts (old!)
    - Floods seen to date: 40+ Gbps (old!)
  - **Distribute** your services - force attacker to flood many points
    - E.g., the root name servers