Security Part One:
Network Attacks and Countermeasures

Xin Zhang

Flashback: Internet design goals

1. Interconnection
2. Failure resilience
3. Multiple types of service
4. Variety of networks
5. Management of resources
6. Cost-effective
7. Low entry-cost
8. Accountability for resources

Where is security?

Why did they leave it out?

- Designed for connectivity
- Network designed with implicit trust
  - No “bad” guys
- Can’t security requirements be provided at the edge?
  - Encryption, Authentication etc.
  - End-to-end arguments in system design

Security Vulnerabilities

- At every layer in the protocol stack!
- Network-layer attacks
  - IP-level vulnerabilities
  - Routing attacks
- Transport-layer attacks
  - TCP vulnerabilities
- Application-layer attacks

IP-level vulnerabilities

- IP addresses are specified by the source
  - Spoofing attacks!
- Use of IP address for authentication
  - e.g., .rhosts, some web sites
- Some IP features that have been exploited
  - Fragmentation Attacks
  - Smurf Attacks

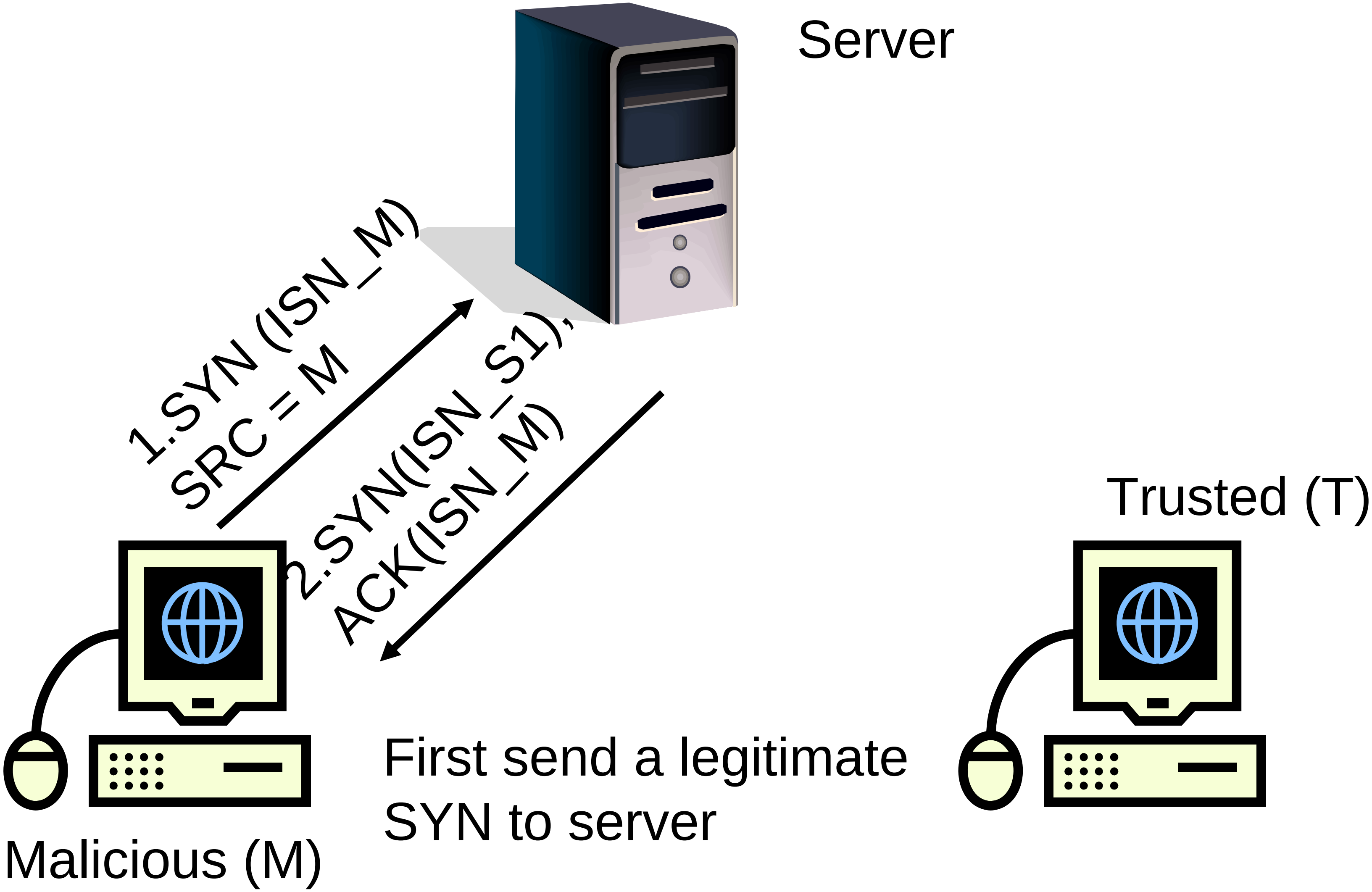
Routing attacks

- Divert traffic to malicious nodes
  - Black-hole attack
  - Dropping or Eavesdropping
- How to implement routing attacks?
  - Distance-Vector
    - Announce low-cost routes
- BGP vulnerabilities
  - Prefix hijacking
  - Path alteration

TCP-level attacks

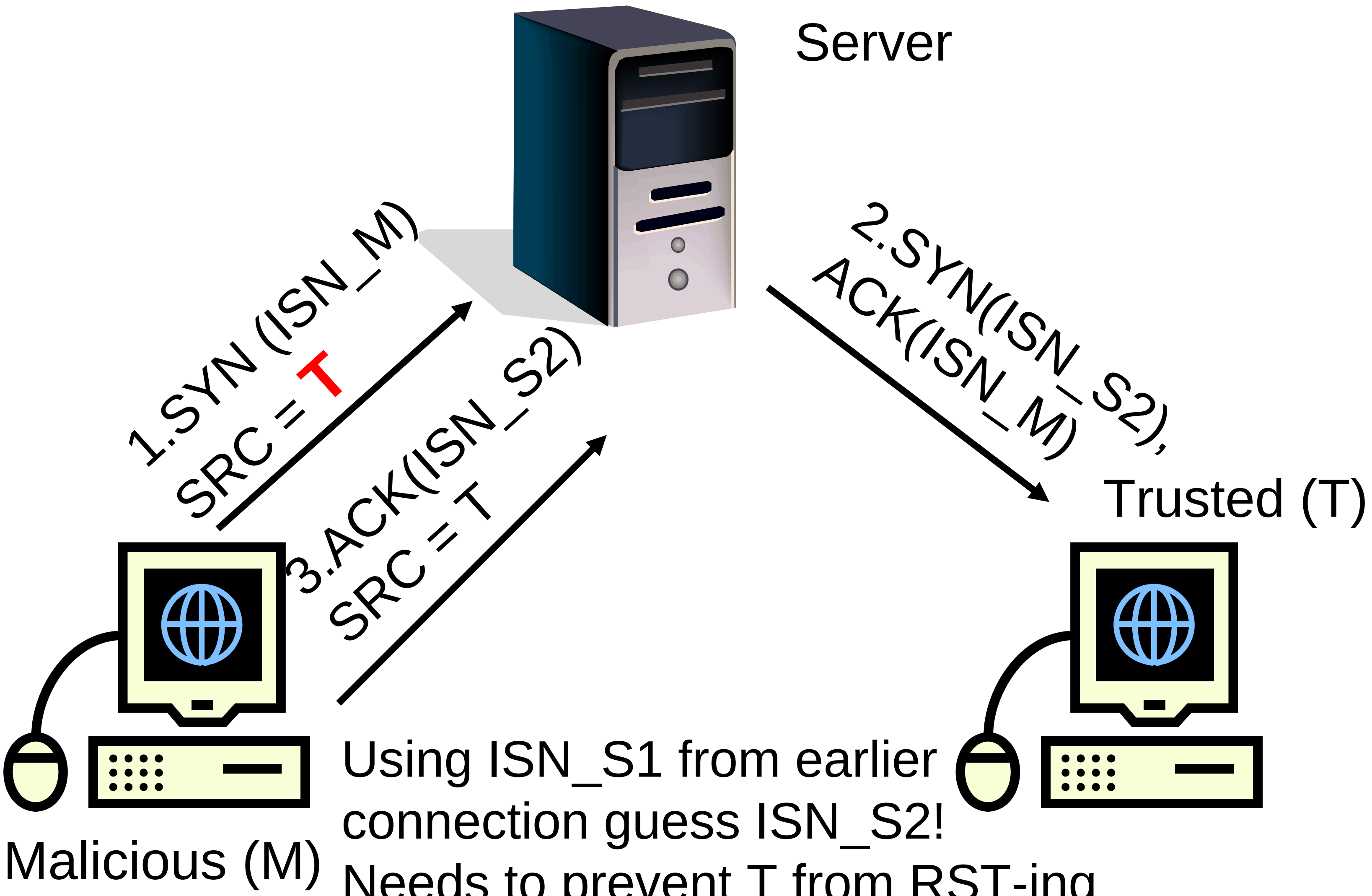
- SYN-Floods
  - Implementations create state at servers before connection is fully established
  - Limited # slots get exhausted
- Session resets
  - Close a legitimate connection
- Session hijack
  - Pretend to be a trusted host
  - Sequence number guessing

Session Hijack



First send a legitimate SYN to server

Session Hijack



Using ISN_S1 from earlier connection guess ISN_S2!
Needs to prevent T from RST-ing

Outline

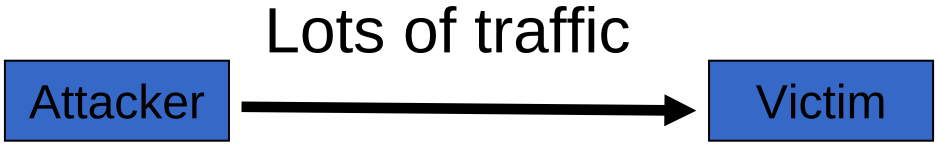
- Security Vulnerabilities
- ***Denial of Service***
- Worms
- Countermeasures: Firewalls/IDS

Denial of Service

- Make a service unusable, usually by overloading the server or network
- Disrupt service by taking down hosts
- Consume host-level resources
  - E.g., SYN-floods
- Consume network resources
  - E.g., UDP/ICMP floods

Simple DoS

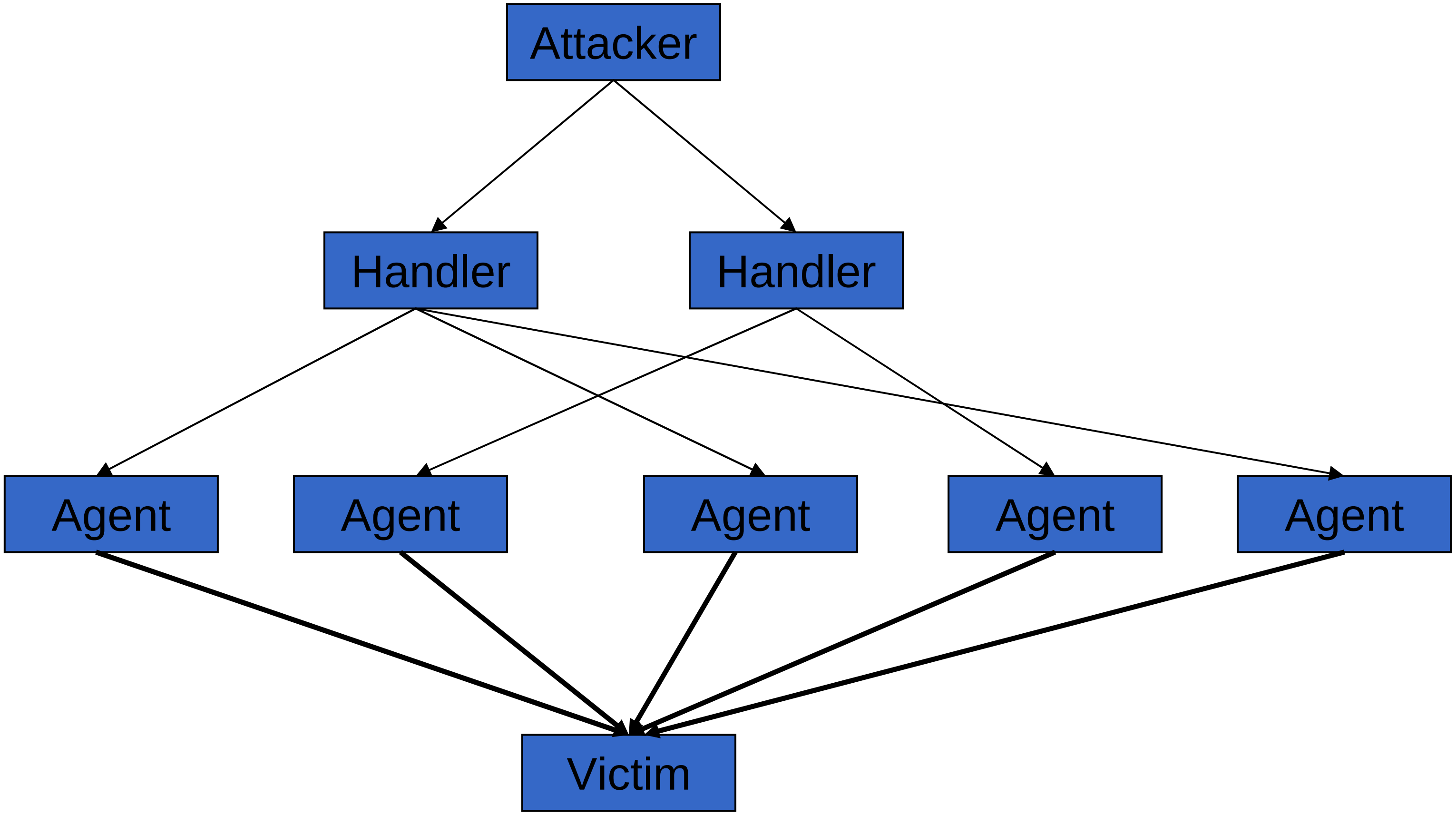
- Attacker generates lots of traffic



- Think of a simple solution?

- Attacker usually spoofs source address to hide origin

Distributed DoS

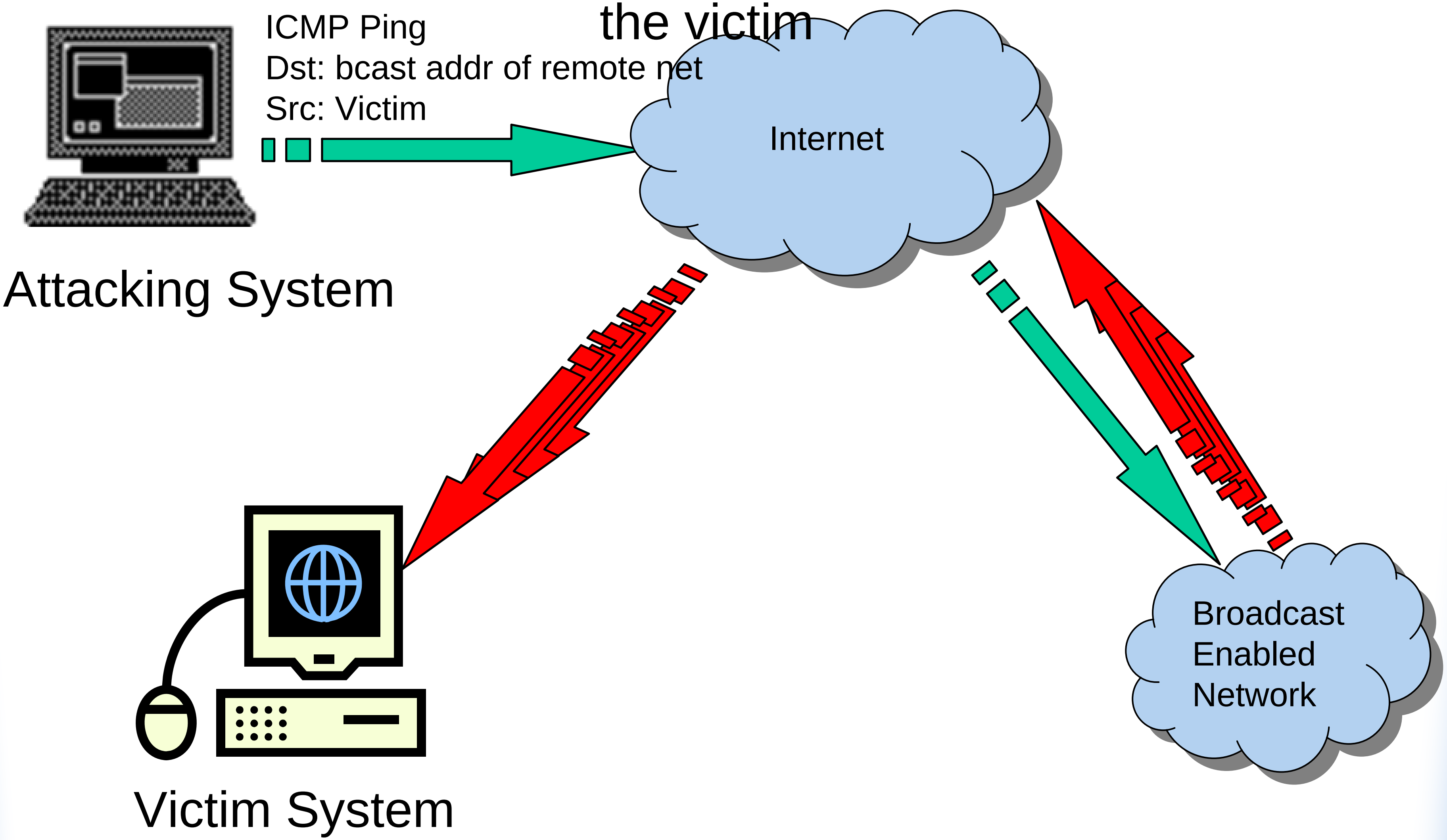


Distributed DoS

- Handlers are usually high volume servers
  - Easy to hide the attack packets
- Agents are usually home users with DSL/Cable
  - Already infected and the agent installed
- Very difficult to track down the attacker
  - Multiple levels of indirection!
- Aside: How to distinguish DDoS from a Flash Crowd?
  - Flash Crowd → Many clients using a service
    - Slashdot Effect

Smurf Attack

Ping to a broadcast IP from the (spoofed) source address of the victim



ICMP Ping
Dst: bcast addr of remote net
Src: Victim

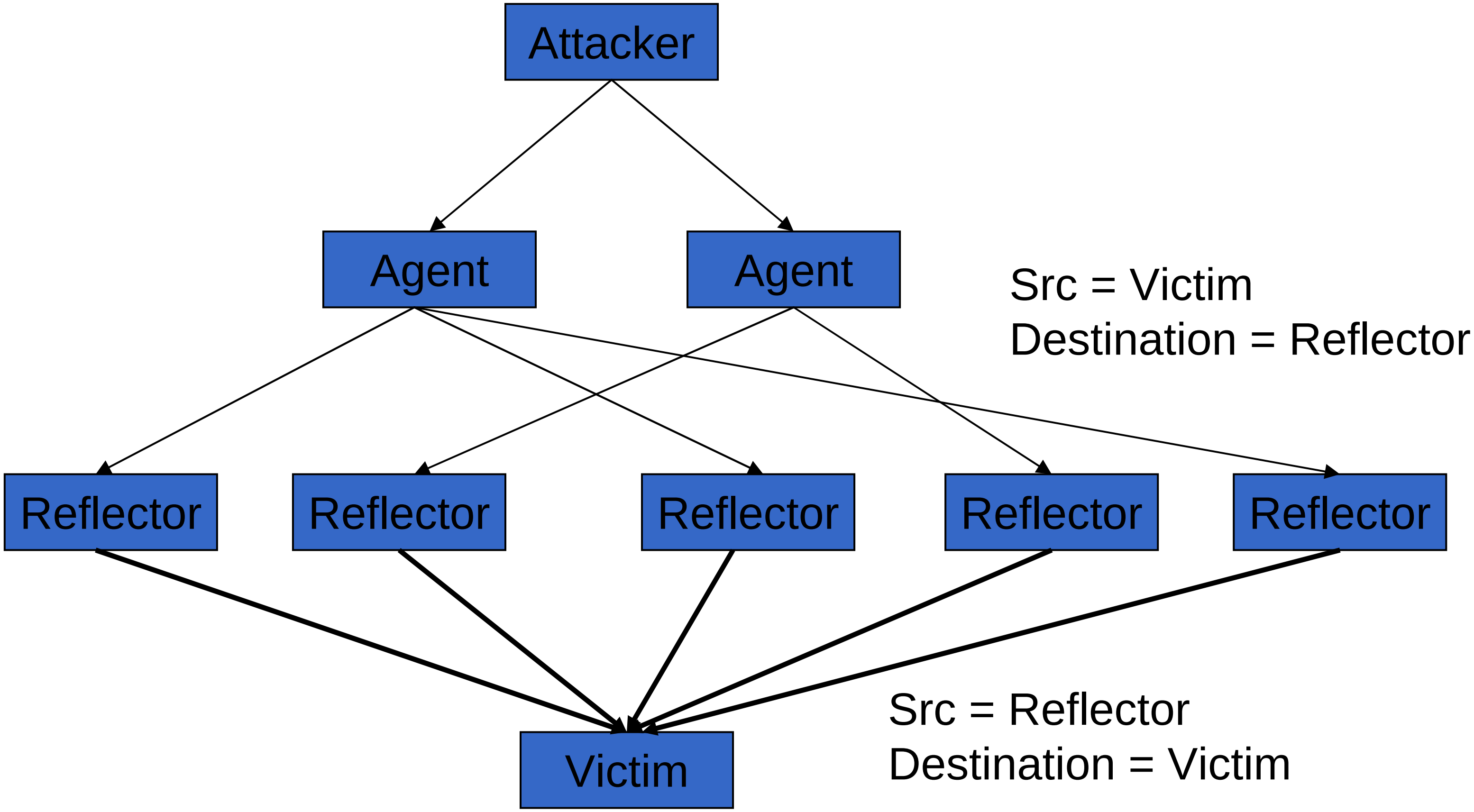
Internet

Attacking System

Broadcast
Enabled
Network

Victim System

Reflector Attack



Attacker

Agent

Agent

Reflector

Reflector

Reflector

Reflector

Reflector

Victim

Src = Victim
Destination = Reflector

Src = Reflector
Destination = Victim

Unsolicited traffic at victim from legitimate hosts

Outline

- Security Vulnerabilities
- Denial of Service
- ***Worms***
- Countermeasures: Firewalls/IDS

Worm Overview

- Self-propagate through network
- Typical Steps in Worm Propagation
  - Probe host for vulnerable software
  - Exploit the vulnerability
    - E.g., Sends bogus input (for buffer overflow – how does it work?)
    - Attacker can do anything that the privileges of the buggy program allow
  - Launches copy of itself on compromised host
- Spread at exponential rate
  - 10M hosts in < 5 minutes
  - Hard to deal with manual intervention

Worm or Virus?

Probing Techniques

- Random Scanning
- Local Subnet Scanning
- Routing Worm
- Pre-generated Hit List
- Topological

Random Scanning

- 32 bit number is randomly generated and used as the IP address
    - Aside: IPv6 worms will be different …
- E.g., Slammer and Code Red I
- Hits black-holed IP space frequently
    - Only 28.6% of IP space is allocated
    - Aside: can track worms by monitoring unused addresses
        - Honeypots

Subnet Scanning

- Generate last 1, 2, or 3 bytes of IP address randomly
- Code Red II and Blaster
- Some scans must be completely random to infect whole internet

Routing Worm

- BGP information can tell which IP address blocks are allocated
- This information is publicly available
  - http://www.routeviews.org/
  - http://www.ripe.net/ris/

Hit List

- Hit list of vulnerable machines is sent with payload
  - Determined before worm launch by scanning
- Gives the worm a boost in the starting phase
- Can avoid detection by the early detection systems

Topological

- Uses info on the infected host to find the next target
  - Morris Worm used /etc/hosts , .rhosts
  - Email address books
  - P2P software usually store info about peers that each host connects to

Some proposals for countermeasures

- Better software safeguards
  - Static analysis and array bounds checking (lint/e-fence)
  - Safe versions of library calls
    - gets(buf) -> fgets(buf, size, ...)
    - sprintf(buf, ...) -> snprintf(buf, size, ...)
- Host-level solutions
  - E.g., Memory randomization, Stack guard
- Host-diversity
  - Avoid same exploit on multiple machines
- Network-level: IP address space randomization
  - Make scanning ineffective
- Rate-limiting: Contain the rate of spread
- Dynamic quarantine: Isolate infected hosts
- Content-based filtering: signatures in packet payloads

Outline

- Security, Vulnerabilities
- Denial of Service
- Worms
- ***Countermeasures: Firewalls/IDS***

Countermeasure Overview

- High level basic approaches
  - Prevention
  - Detection
  - Resilience
- Requirements
  - Security: soundness / completeness (false positive / negative
  - Overhead
  - Usability

Design questions ..

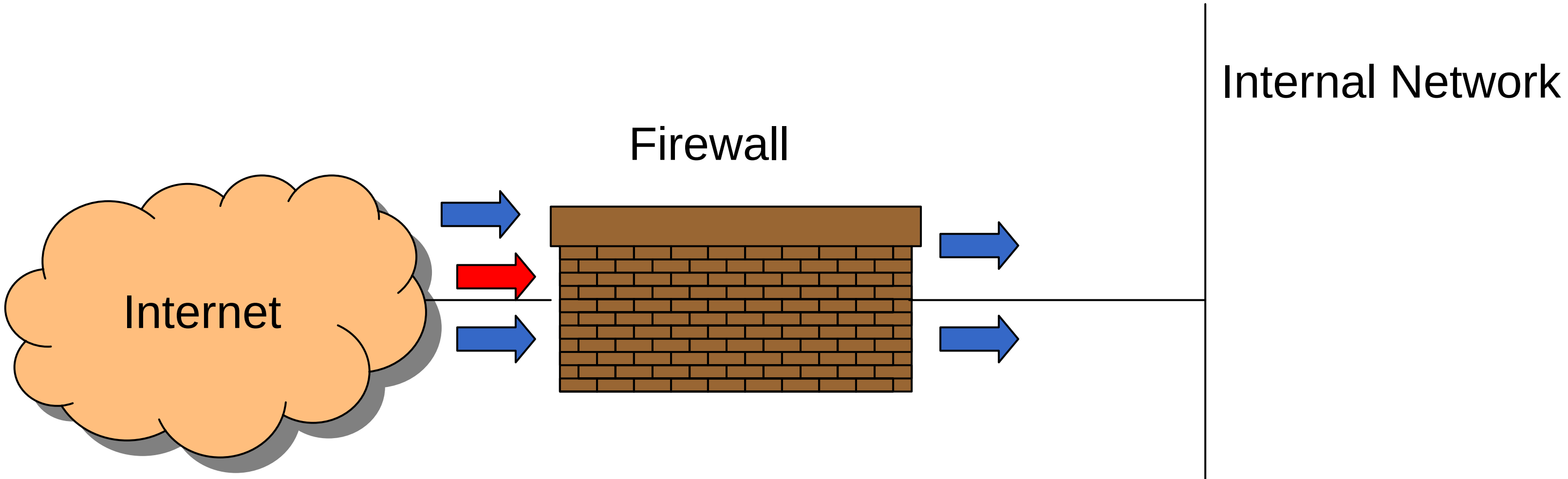
- Why is it so easy to send unwanted traffic?
  - Worm, DDoS, virus, spam, phishing etc
- Where to place functionality for stopping unwanted traffic?
  - Edge vs. Core
  - Routers vs. Middleboxes
- Redesign Internet architecture to detect and prevent unwanted traffic?

Firewalls

- Lots of vulnerabilities on hosts in network
- Users don’t keep systems up to date
  - Lots of patches
  - Zero-day exploits
- Solution
  - Limit access to the network
  - Put firewalls across the perimeter of the network

Firewalls (contd...)

- Firewall inspects traffic through it
- Allows traffic specified in the policy
- Drops everything else
- Two Types
  - Packet Filters, Proxies



Packet Filters

- Selectively passes packets from one network interface to another

- Usually done within a router between external and internal network

- What to filter based on?
  - Packet Header Fields
    - IP source and destination addresses
    - Application port numbers
    - ICMP message types/ Protocol options etc.
  - Packet contents (payloads)

Packet Filters: Possible Actions

- Allow the packet to go through
- Drop the packet (Notify Sender/Drop Silently)
- Alter the packet (NAT?)
- Log information about the packet

Some examples

- Block all packets from outside except for SMTP servers
- Block all traffic to/from a list of domains
- Ingress filtering
  - Drop all packets from outside with addresses inside the network
- Egress filtering
  - Drop all packets from inside with addresses outside the network

Firewall implementation

- Stateless packet filtering firewall
- Rule → (Condition, Action)
- Rules are processed in top-down order
    - If a condition satisfied – action is taken

Packet Filters

- Advantages
  - Transparent to application/user
  - Simple packet filters can be efficient
- Disadvantages
  - Security
  - Overhead (speed)
  - Usability

    - Very hard to configure the rules
    - Doesn’t have enough information to take actions (Does port 22 always mean SSH? Who is the user accessing the SSH?)

Alternatives

- Stateful packet filters
  - Keep the connection states
  - Easier to specify rules
  - Problems?
    - State explosion
    - State for UDP/ICMP?
- Proxy Firewalls
  - Two connections instead of one
  - Either at transport level
    - SOCKS proxy
  - Or at application level
    - HTTP proxy

Intrusion Detection Systems

- Firewalls allow traffic only to legitimate hosts and services
- Traffic to the legitimate hosts/services can have attacks
- Solution?
  - Intrusion Detection Systems
  - Monitor data and behavior
  - Report when identify attacks

Classes of IDS

- What type of analysis?
  - Signature-based
  - Anomaly-based
- Where is it operating?
  - Network-based
  - Host-based

Summary

- Security vulnerabilities are real!
  - Protocol or implementation or bad specs
  - Poor programming practices
  - At all layers in protocol stack
- DoS/DDoS
  - Resource utilization
- Worm
  - Exponential spread
  - Scanning strategies
- Firewall/IDS
  - Counter-measures to protect hosts
  - Fail-open vs. Fail-close?