COMP/ELEC 429/556
Introduction to Computer Networks

Network security

Some slides used with permissions from Edward W. Knightly, T. S. Eugene Ng, Ion Stoica, Hui Zhang

The 2004 A. M. Turing Award Goes to...

Bob Kahn Vint Cerf

- *"For pioneering work on internetworking, including the design and implementation of the Internet's basic communications protocols, TCP/IP, and for inspired leadership in networking."*

Design Philosophy of the DARPA Internet Protocols by David D. Clark (1988)

1. Internet c... of networks or gatewa...
2. The Intern... ...nications service
3. The Intern... ...ety of networks
4. The Intern... ...anagement of its resou...
5. The Intern...
6. The Intern... ...ent with a low level c...
7. The resou... ...t be accountabl...

Importance of Network Security

- Internet currently used for important services
  - Financial transactions, medical records
- Could be used in the future for *critical* services
  - 911, surgical operations, energy system control, transportation system control
- Networks more open than ever before
  - Global, ubiquitous Internet; wireless access
- Malicious Users
  - Selfish users: want more network resources
  - Malicious users: want to do harm

Network Security: Our Focus

- Host Compromise
  - Attacker gains control of a host
- Denial-of-Service
  - Attacker prevents legitimate users from gaining service

- Note: Attack can be both
  - E.g., host compromise that provides resources for denial-of-service

Host Compromise Method 1: User Exploitation

- Many applications rely on the user to decide if a potentially dangerous action should be taken, e.g.,
  - Run code downloaded from the Internet
    - "Do you accept content from Microsoft?"
  - Run code attached to email
    - "Subject: You've got to see this!"
  - Allow a macro in a data file to be run
    - "Here is the latest version of the document."

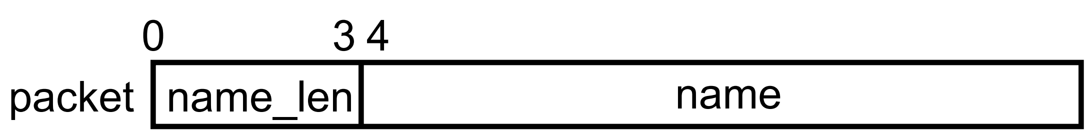
Host Compromise Method 2: Stack Based Buffer Overflow (User not involved!)

- Code can have many unknown bugs because those bugs are not triggered by common input
- Such bugs in network facing code can be triggered by inputs received from the network
- Network facing code that runs with high privileges (i.e., as root) is especially dangerous

Example

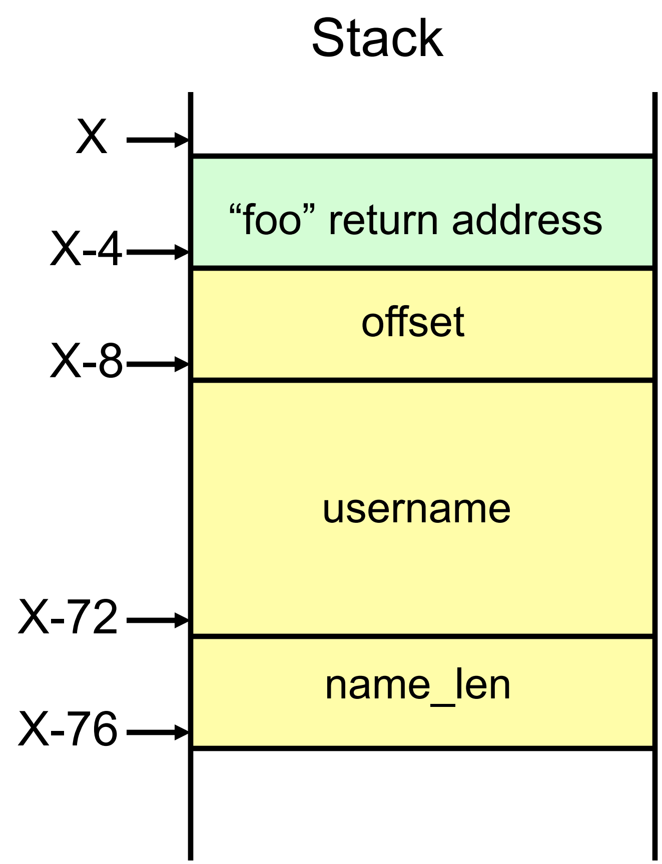
- What is wrong here?

```c
#define MAXNAMELEN 64
int offset = 4;
char username[MAXNAMELEN];
int name_len;

name_len = ntohl(*(int *)packet);
memcpy(&username, packet[offset], name_len);
```



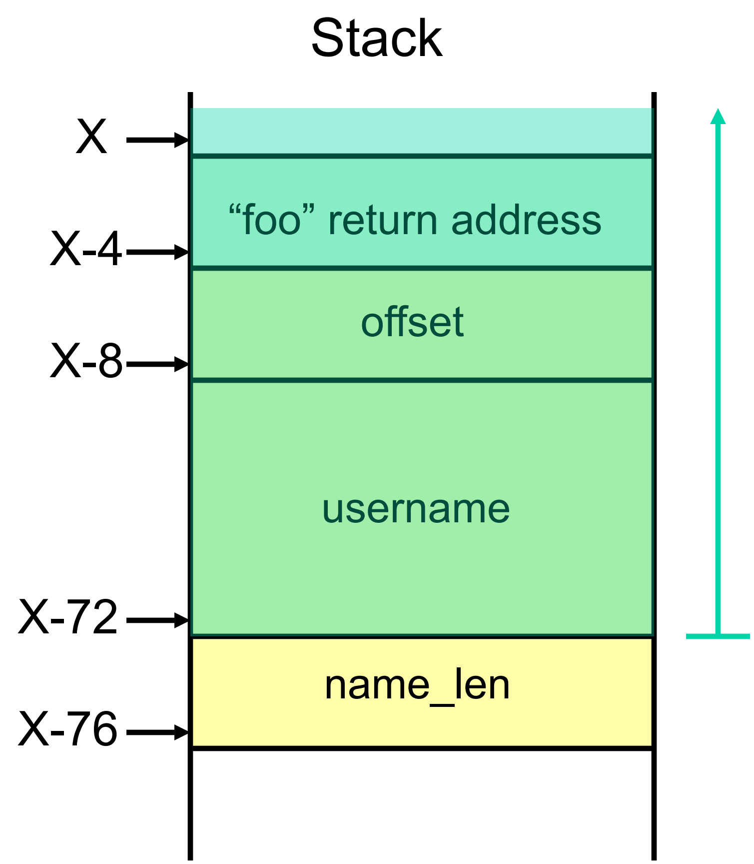
Example

```
void foo(packet) {
  #define MAXNAMELEN 64
  int offset = 4;
  char username[MAXNAMELEN];
  int name_len;

  name_len = ntohl(*(int*)packet);
  memcpy(&username,
         packet[offset],name_len);
  …
}
```



Example

```
void foo(packet) {
  #define MAXNAMELEN 64
  int offset = 4;
  char username[MAXNAMELEN];
  int name_len;

  name_len = ntohl(*(int *) packet);
  memcpy(&username,
         packet[offset],name_len);
  ...
}
```



Hall of Shame

- Software that have had many stack overflow bugs:
  - BIND (a popular DNS server)
  - RPC (Remote Procedure Call, used for Network File System)
  - Sendmail (a popular UNIX mail delivery software)
  - IIS (Windows web server)
  - SNMP (Simple Network Management Protocol, used to manage routers and other network devices)

Effect of Stack Based Buffer Overflow

- Write into part of the stack or heap
  - Write arbitrary code to part of memory
  - Cause program execution to jump to arbitrary code
- Worm
  - Sends bogus input to potential victims to spread the worm
  - Attacker can do anything that the privileges of the buggy program allows
    - Launches copy of itself on compromised host

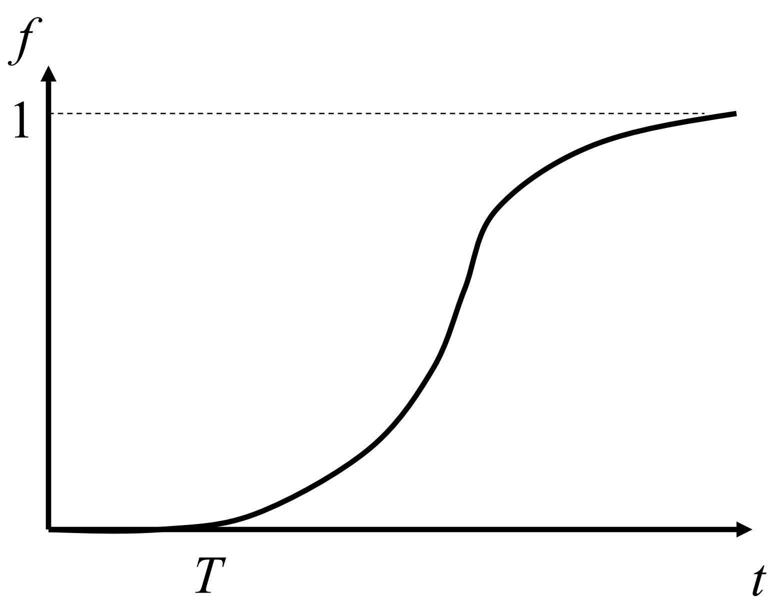
Internet Worm

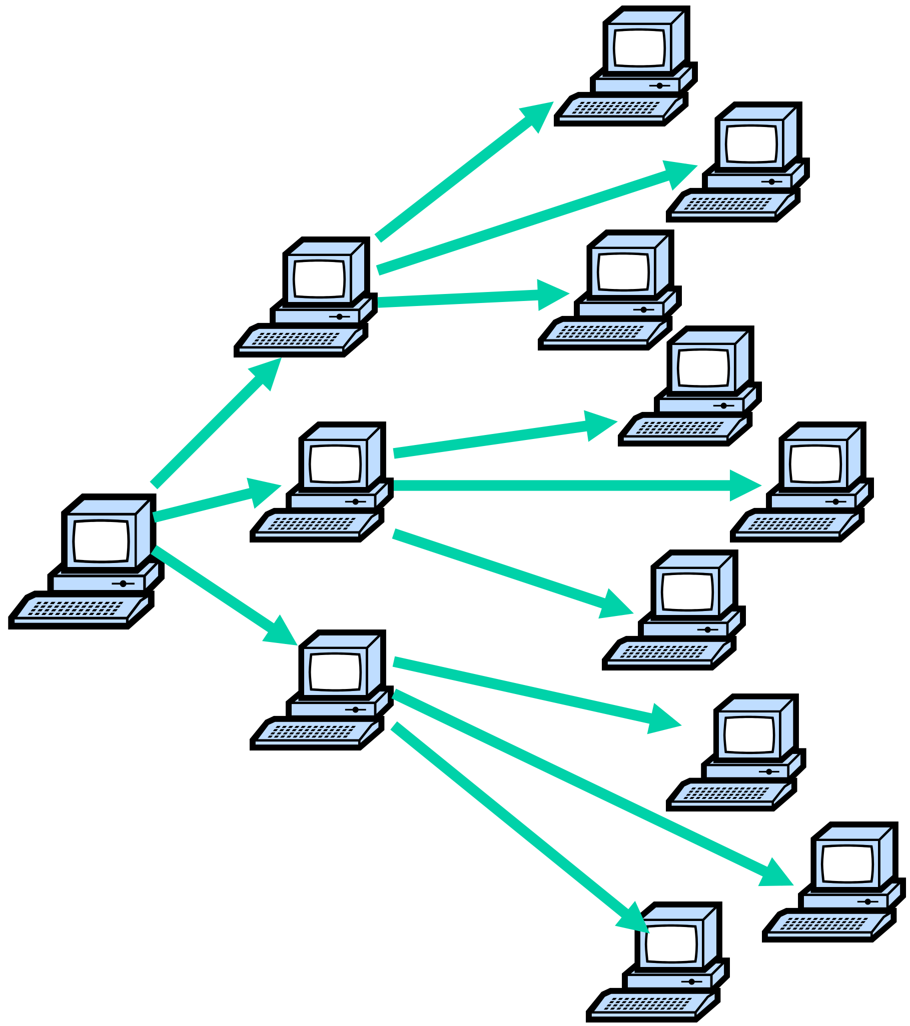
- One of earliest major Internet security incidents
  - Internet Worm (1988): compromised almost every BSD-derived machine on Internet
- Today: estimated that a single worm could compromise 10M hosts in < 5 min
- Attacker gains control of a host
  - Reads data
  - Erases data
  - Compromises another host
  - Launches denial-of-service attack on another host

Worm Spreading

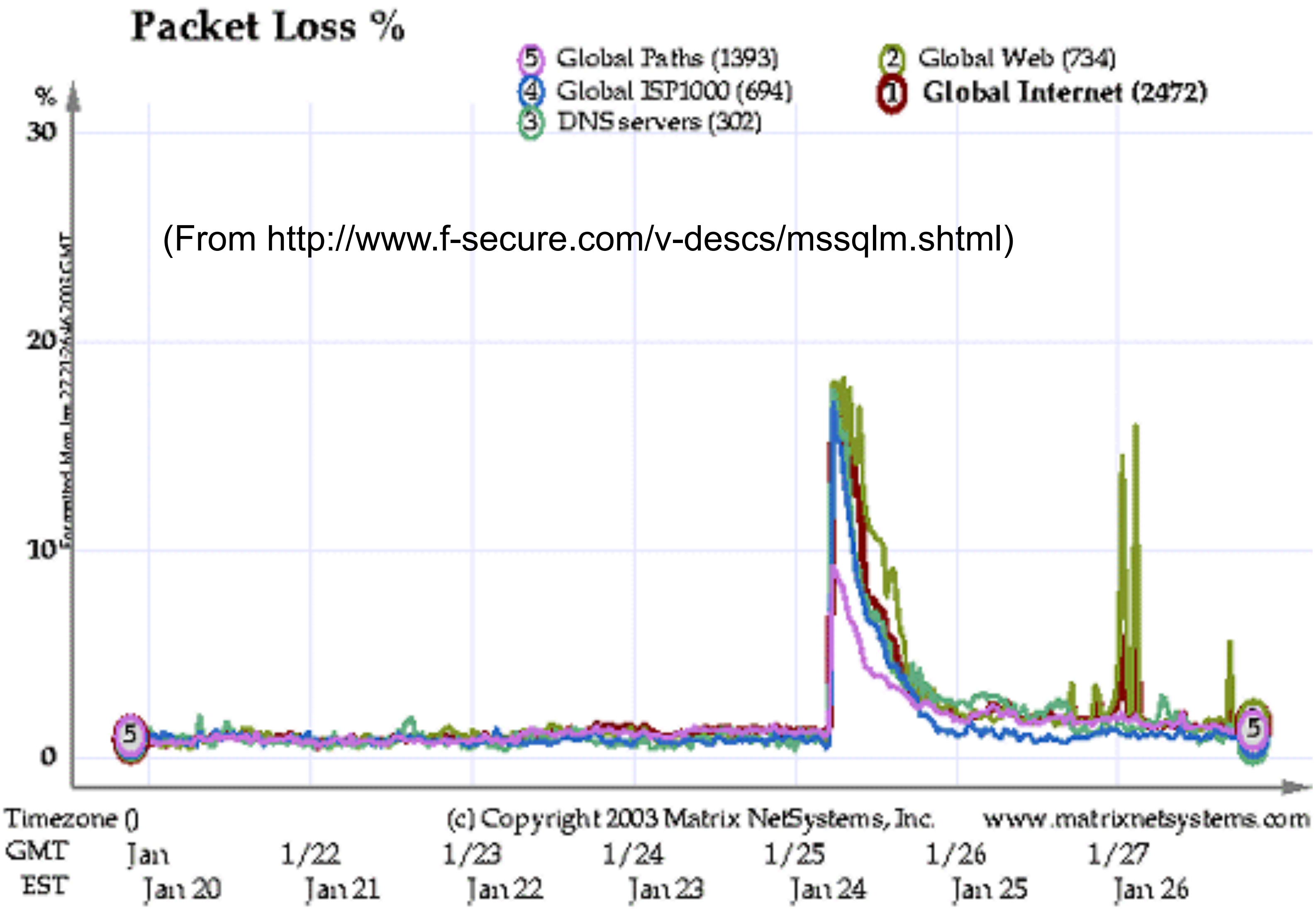
$$f = (e^{K(t-T)} - 1) / (1 + e^{K(t-T)})$$

- f – fraction of hosts infected
- K – rate at which one host can contact others
- T – start time of the attack





e.g. MS SQL Slammer (January 2003)



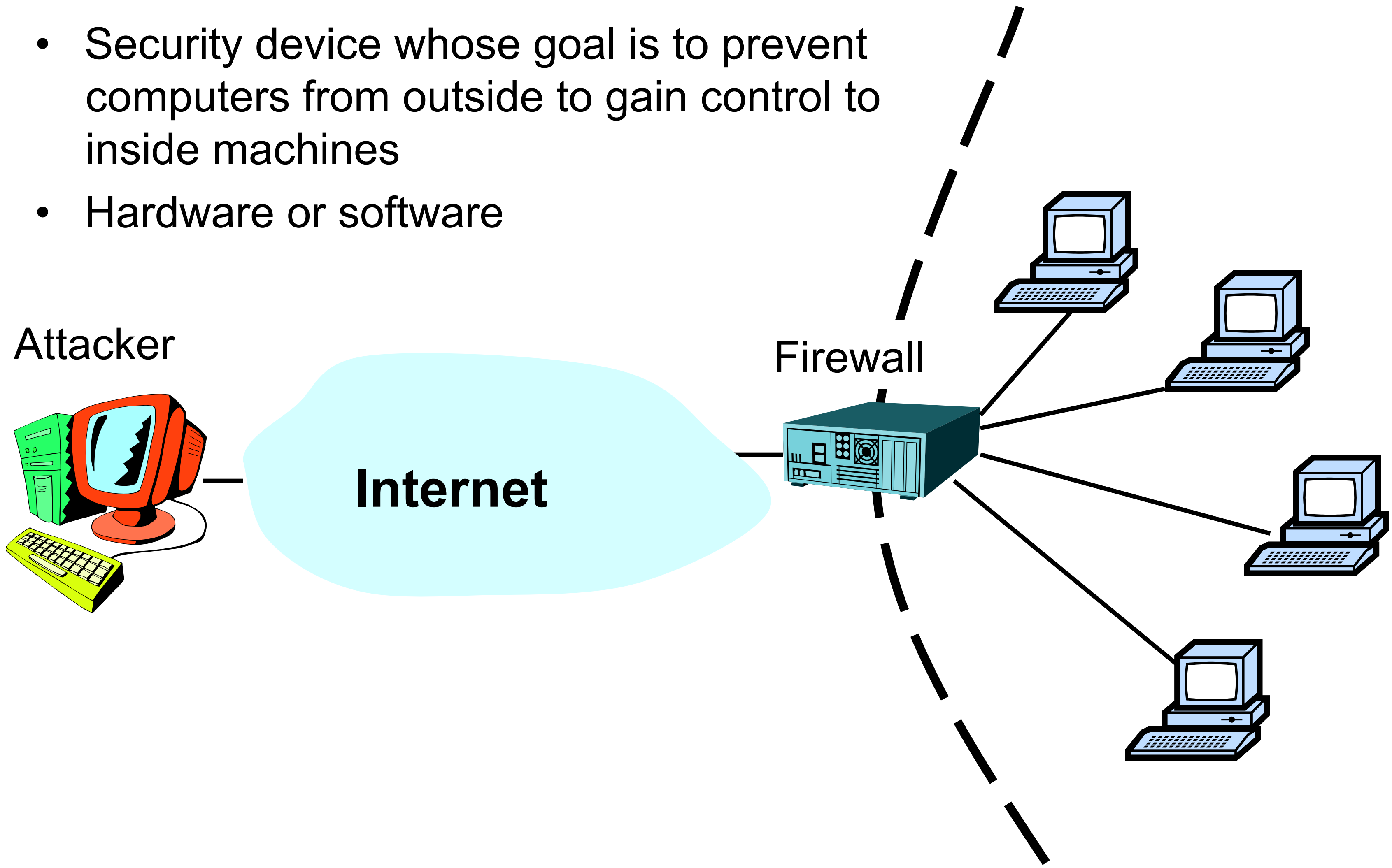
(From http://www.f-secure.com/v-descs/mssqlm.shtml)

Potential Solutions

- Don't write buggy software
- Type-safe Languages
  - Unrestricted memory access of C/C++ contributes to problem
  - Use Java, Python, etc. instead (but there is a performance cost)
- Operating system
  - Make stack memory pages non-executable
  - Add a guard next to the return address on stack, check guard value before executing jump instruction
  - Compartmentalize programs better, so one compromise doesn't compromise the entire system
    - E.g., DNS server doesn't need total system access
- Firewalls

Firewall

- Security device whose goal is to prevent computers from outside to gain control to inside machines
- Hardware or software



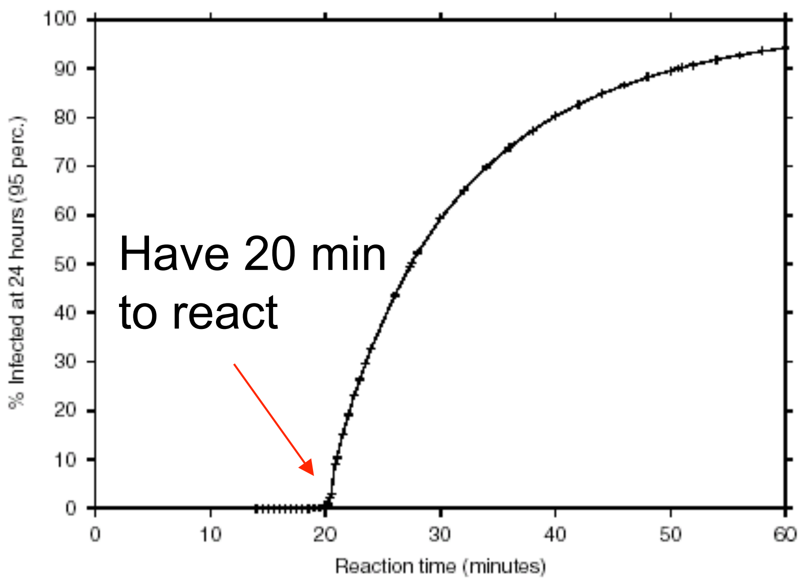
Firewall

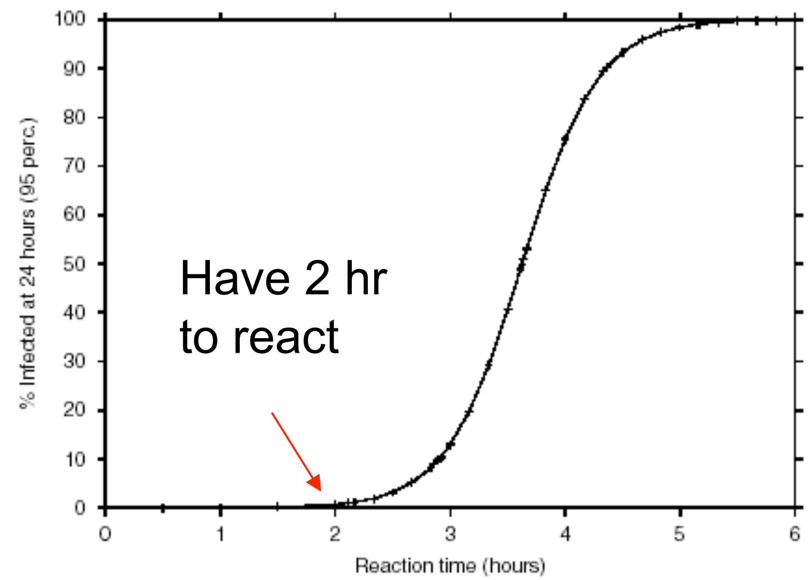
- Restrict traffic between Internet and devices (machines) behind it based on
  - Source address and port number
  - Payload
  - Stateful analysis of data
- Examples of rules
  - Block any external packets not for port 80
  - Block any email with an attachment
  - Block any external packets with an internal IP address as source

Firewall Properties

- Easier to deploy firewall than secure all internal hosts
- Doesn’t prevent user exploitation
- Can’t prevent problem from spreading from within
- Tradeoff between availability of services (firewall allows/ disallows ports) and security
  - If firewall is too restrictive, users will find way around it, thus compromising security
  - E.g., have all services use port 80, which is typically not blocked by firewalls

Address Blacklisting and Content Filtering at Firewalls: Code Red Worm Example





Address Blacklisting: Set firewalls to block an IP address after it has been detected as malicious

Content Filtering: Set firewalls to block packets with malicious contents after such contents are identified

Bottom line: Internet worms are very hard to contain

Compromised hosts are then used to launch Denial of Service attacks

Denial of Service

- Huge problem in current Internet
  - Major sites attacked: Google, Yahoo!, Amazon, eBay, CNN, Microsoft, ...
  - Almost all attacks launched from compromised hosts
    - So called Botnets
- Impact
  - Prevent legitimate users from gaining service by overloading or crashing the service

Exploit "Asymmetry"

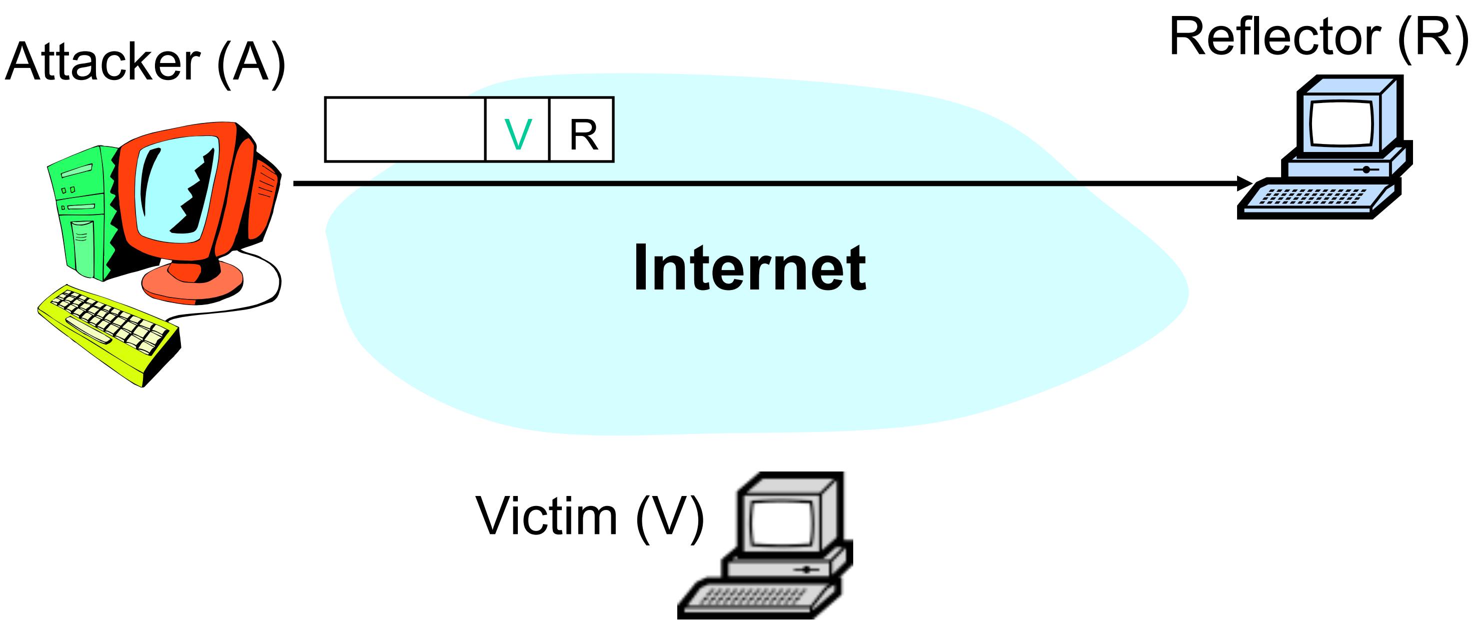
- Attacker possesses asymmetric amount of power to disrupt victim than the victim's power to defend itself
- E.g. Use a large number of compromised hosts
- E.g. Exploit "asymmetry" in protocol designs and in software implementations
  - Focus of our discussion

E.g. Ping Flood

- Attacker sends ICMP PING packets with IP subnet broadcast address as the destination address
  - e.g. 128.42.255.255
- Last hop router broadcasts the PING packet in the victim's Ethernet network
- Amplify attacker's power to congest the victim's Ethernet network

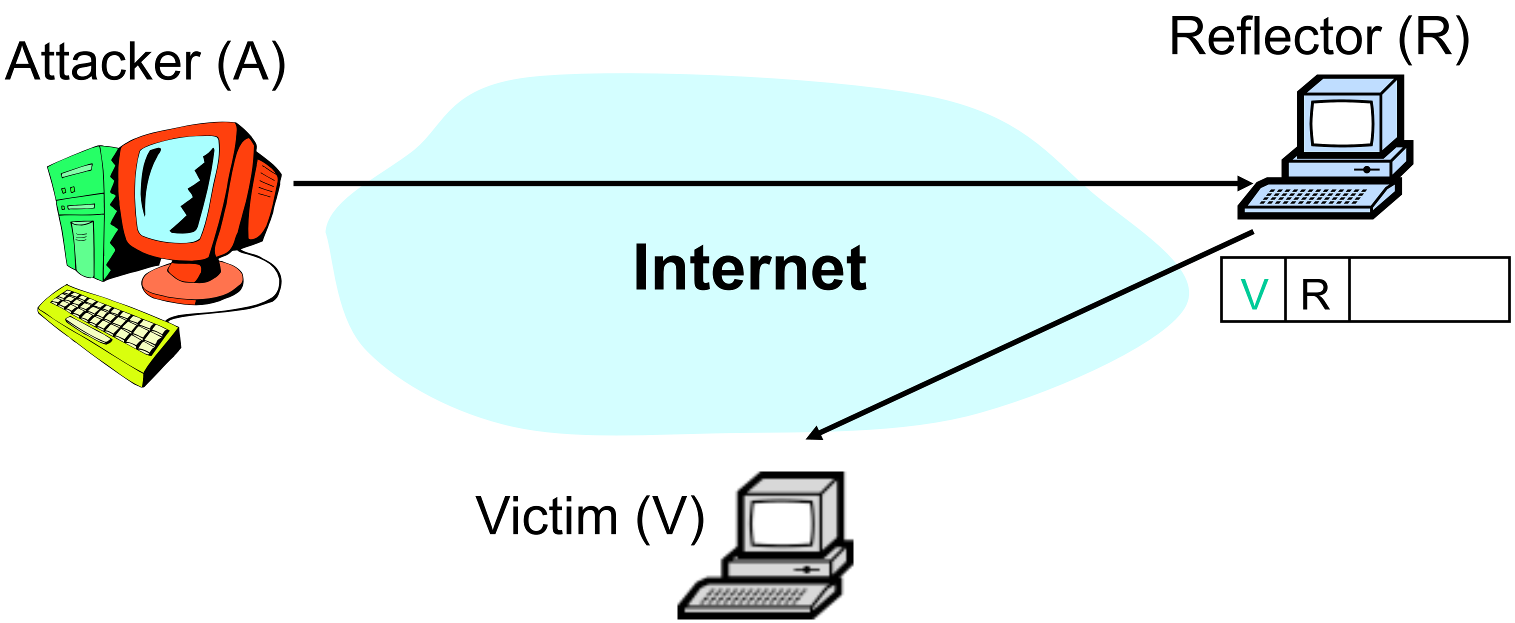
E.g. Reflection

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



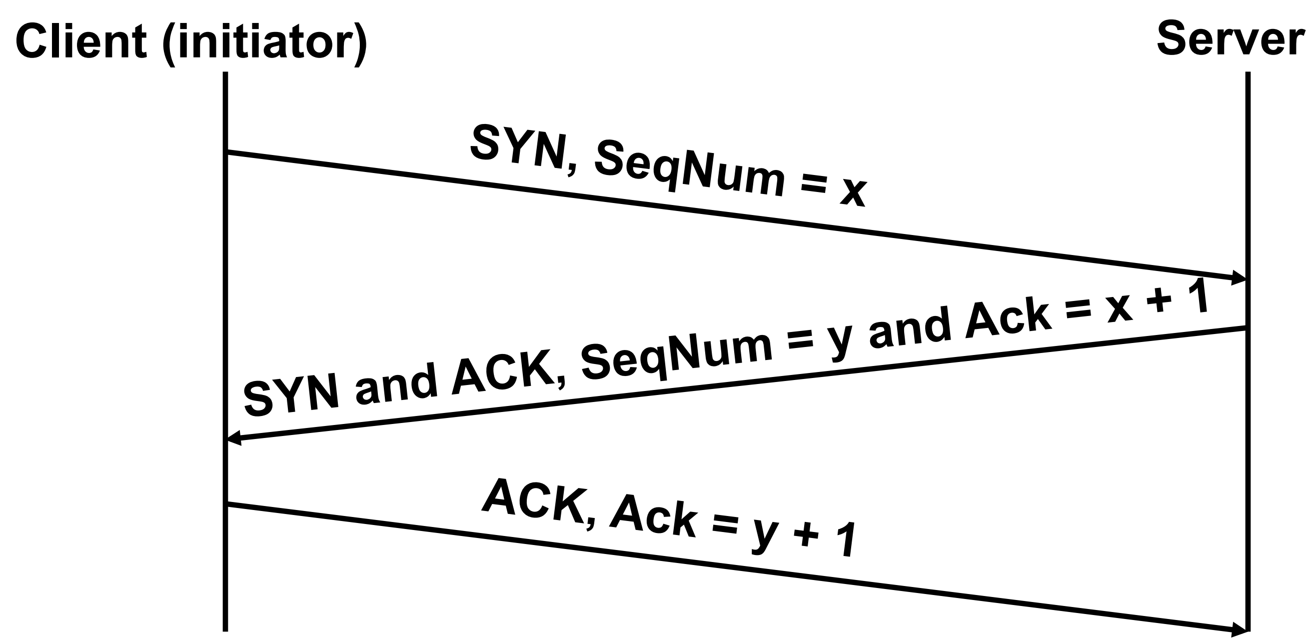
E.g. Reflection

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



E.g. SYN Attack (Recap: 3-Way Handshaking)

- Goal: agree on a set of parameters: the start sequence number for each side
  - Starting sequence numbers are random.



E.g. SYN Attack

- Attacker: send at max rate TCP SYN with random spoofed source address to victim
  - Spoofing: use a different source IP address than own
  - Random spoofing allows one host to pretend to be many
- Victim receives many SYN packets
  - Send SYN+ACK back to spoofed IP addresses
  - Holds some memory until 3-way handshake completes
    - Usually never, so victim times out after long period (e.g., 3 minutes)

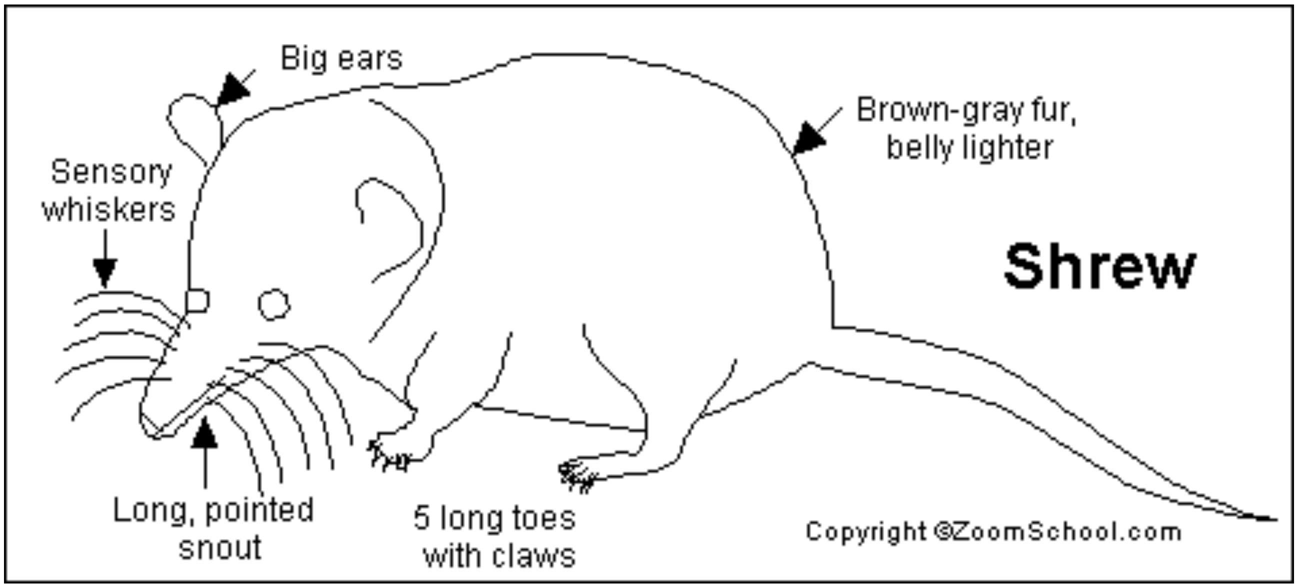
Effect on Victim

- Buggy implementations allow unfinished connections to eat all memory, leading to crash
- Better implementations limit the number of unfinished connections
  - Once limit reached, new SYNs are dropped
- Effect on victim's users
  - Users can't access the targeted service on the victim because the unfinished connection queue is full → DoS

Solution: SYN Cookies

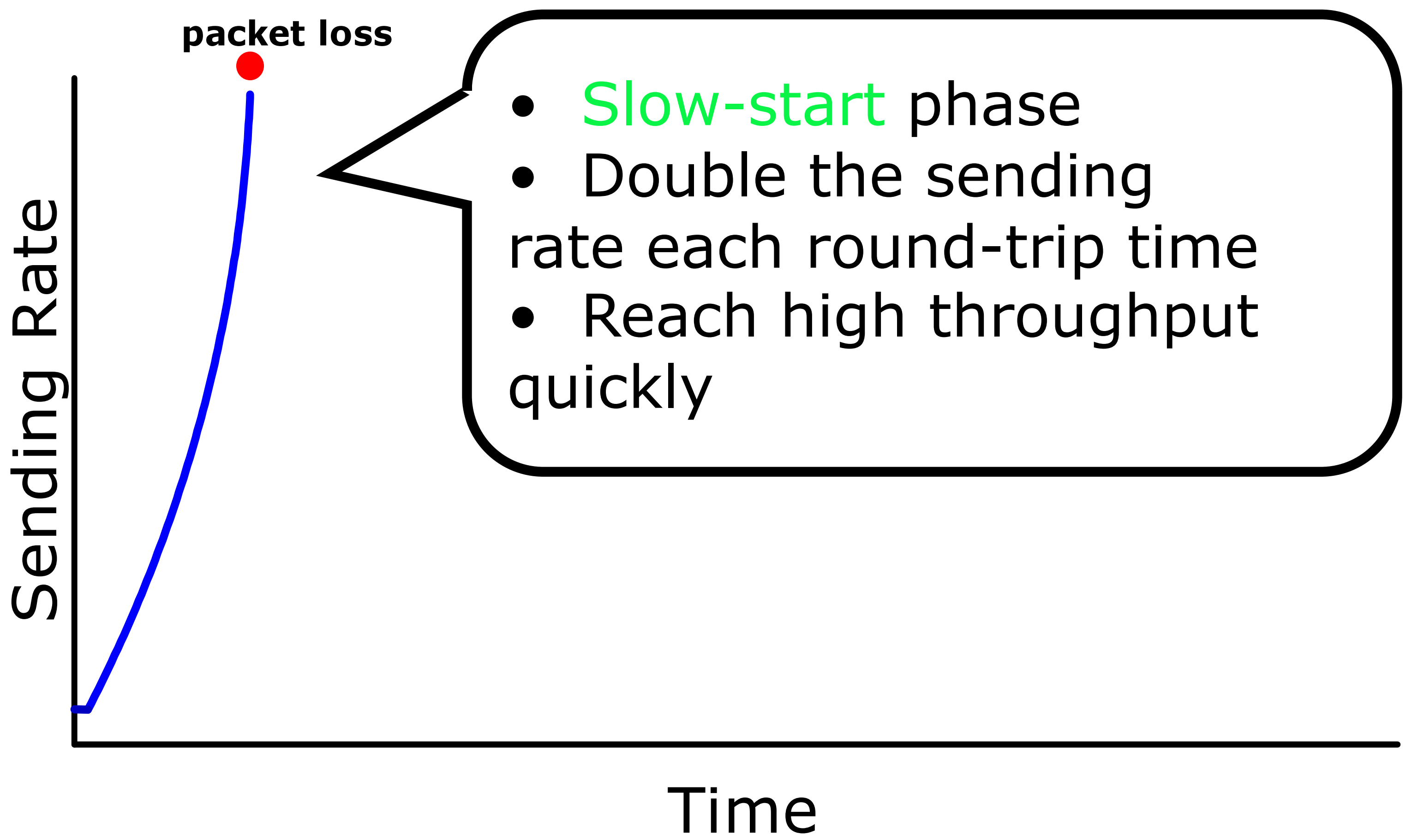
- Server: send SYN-ACK with sequence number y, where
  - y = H(client_IP_addr, client_port, server_secret)
  - H(): one-way hash function
- Client: send ACK containing y+1
- Sever:
  - verify if y = H(client_IP_addr, client_port, server_secret)
  - If verification passes, allocate memory
- Note: server doesn't allocate any memory if the client's address is spoofed

E.g. "Shrew attack" (Low rate TCP attack) by former ECE student Aleksander Kuzmanovic

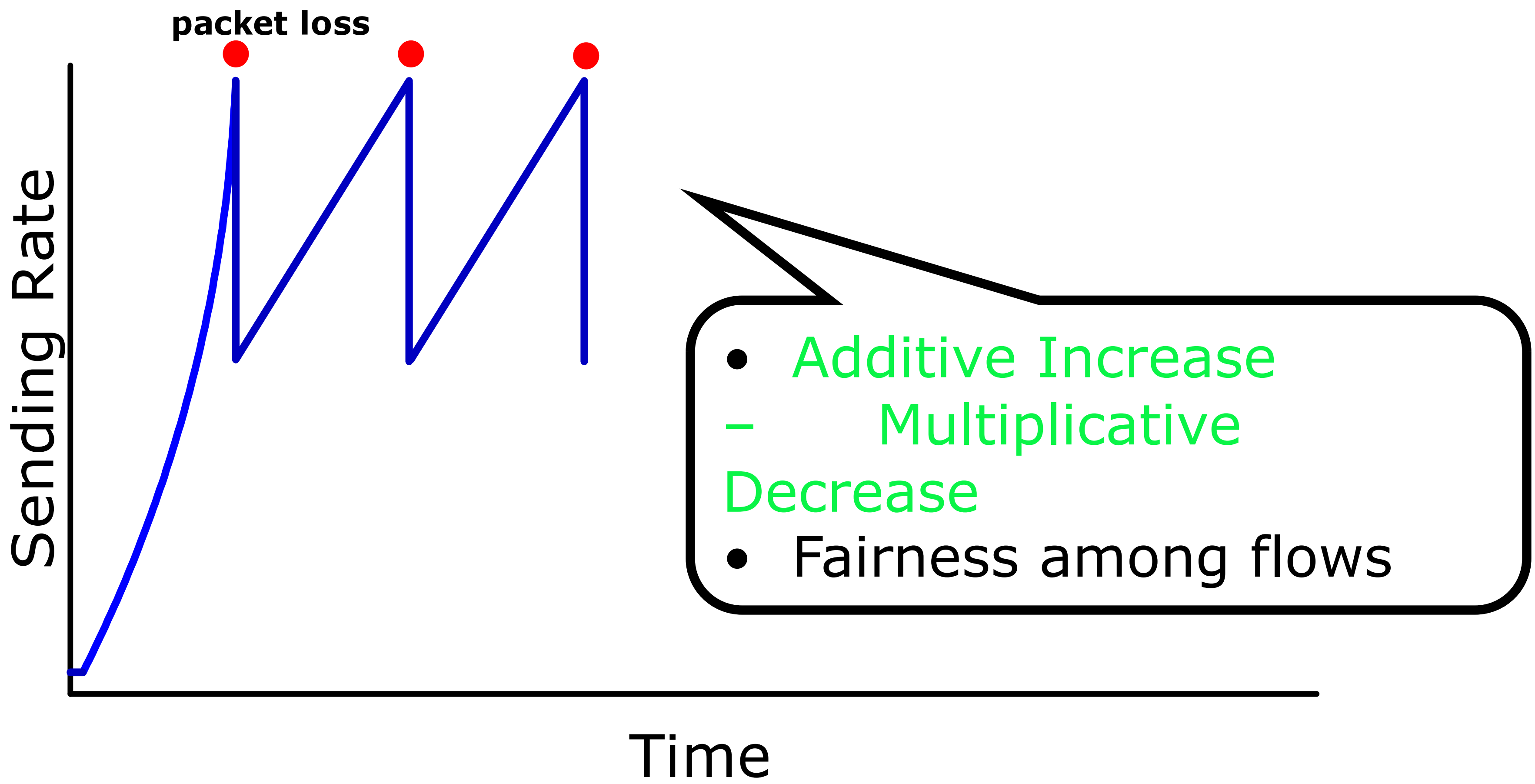


- Very small but aggressive mammal that ferociously attacks and kills much larger animals with a venomous bite

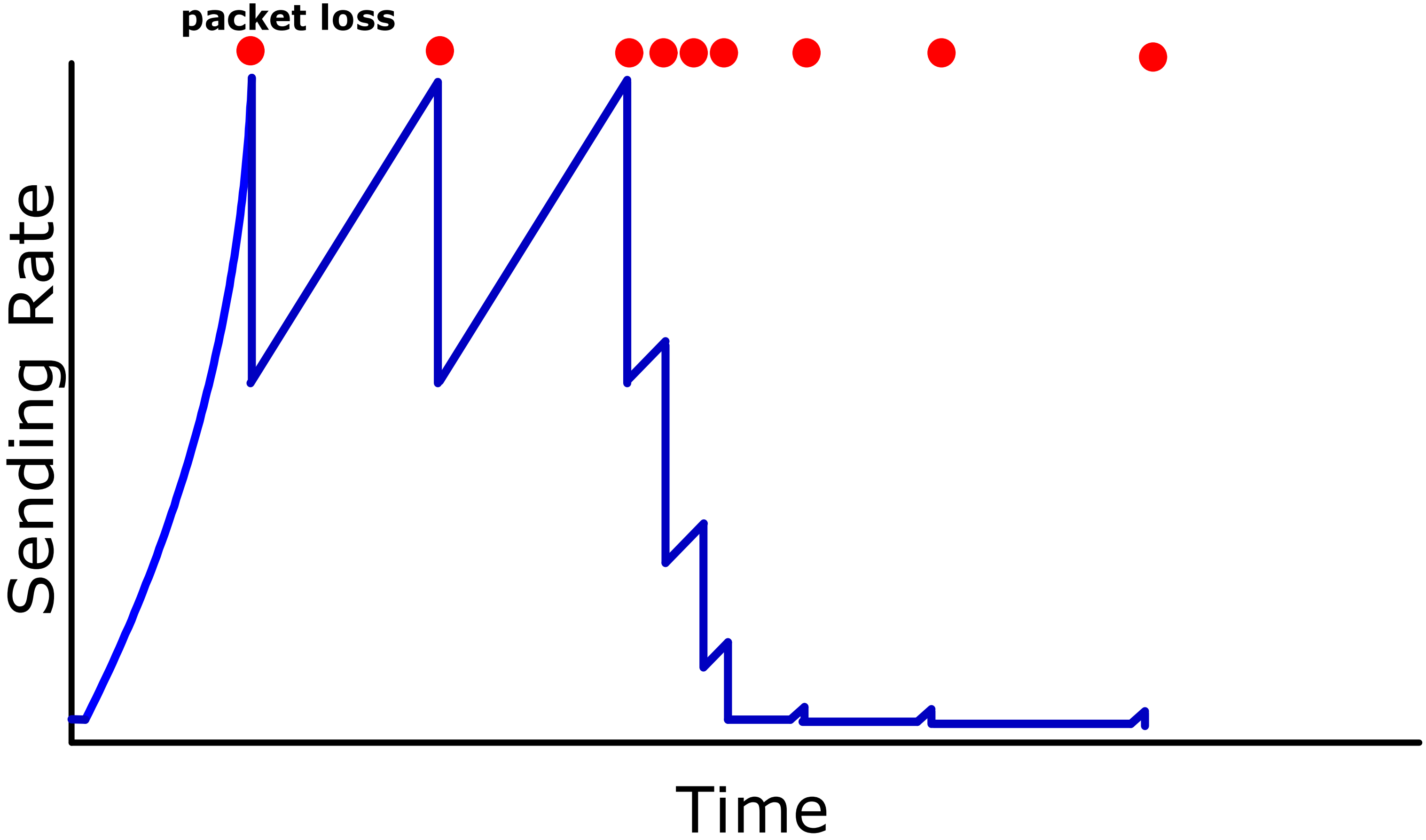
TCP Congestion Control



TCP Congestion Control



TCP Congestion Control

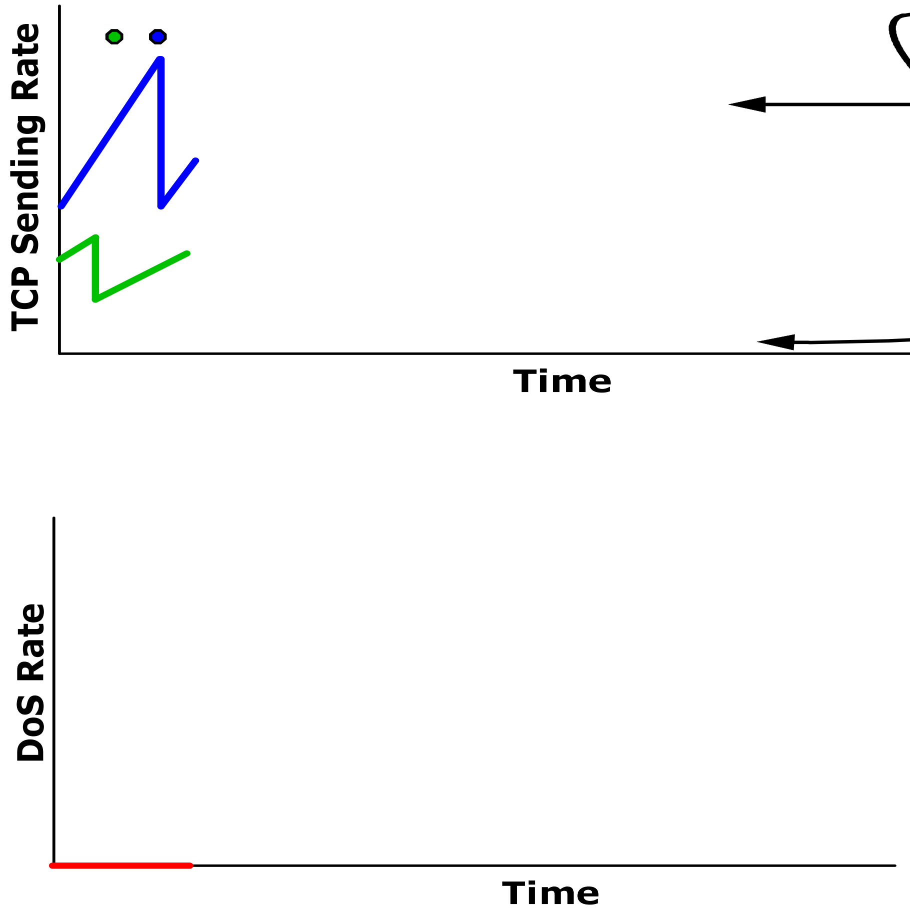


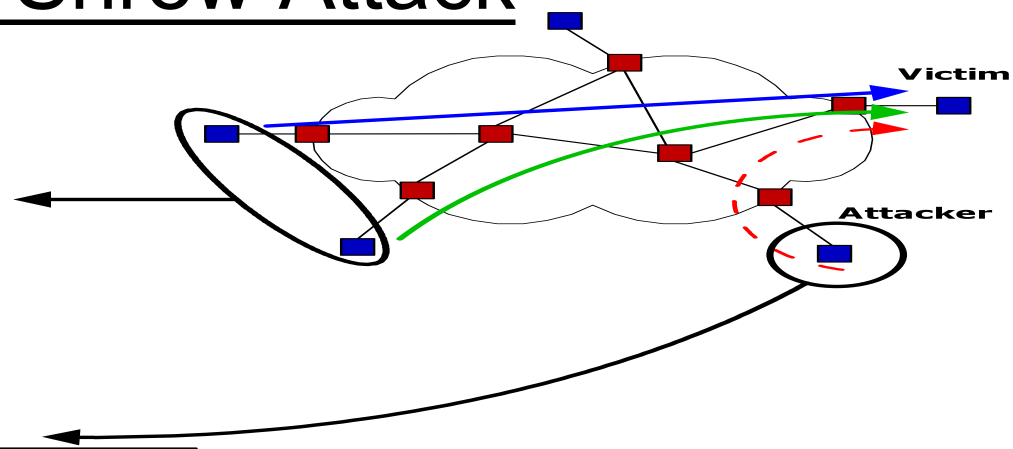
TCP Retransmssion Timeout (RTO)

- *RTO=SRTT+4*RTTVAR*
  - RTT ~ 10-100ms

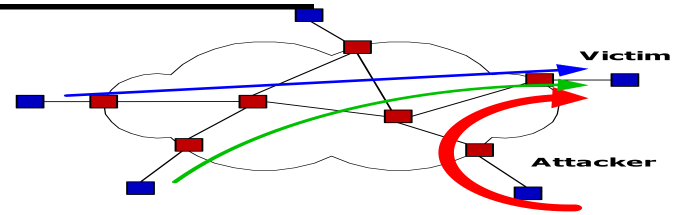
- Lower-bounding the RTO parameter:
  - [AllPax99]: minRTO = 1 sec
    - to avoid spurious retransmissions
  - RFC6298 recommends minRTO = 1 sec
    - Many implementations actually use 200ms

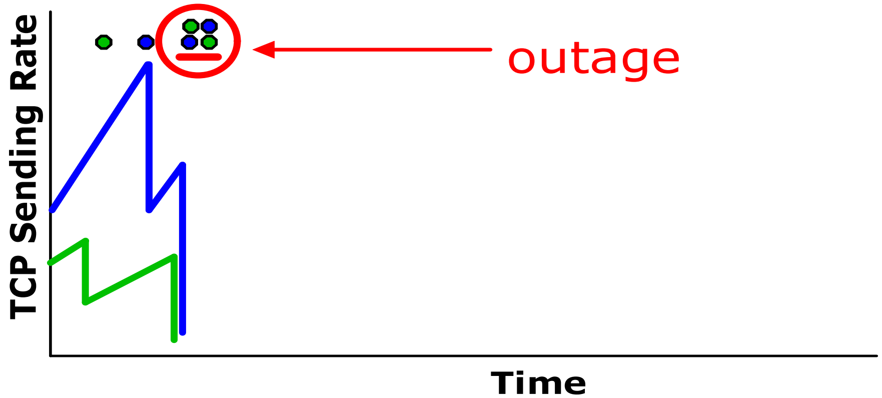
The Shrew Attack

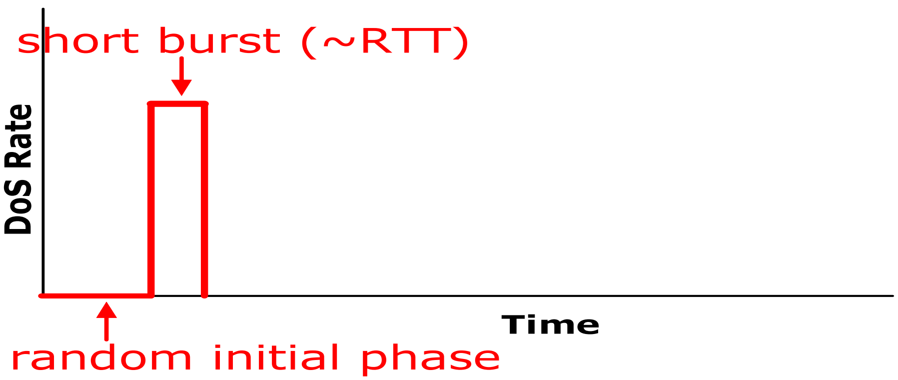




The Shrew Attack

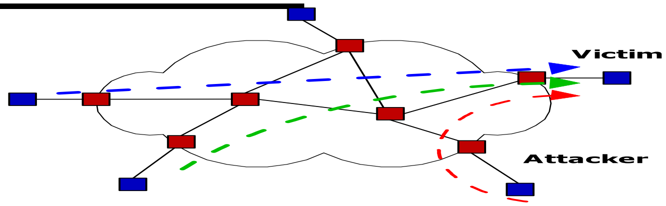


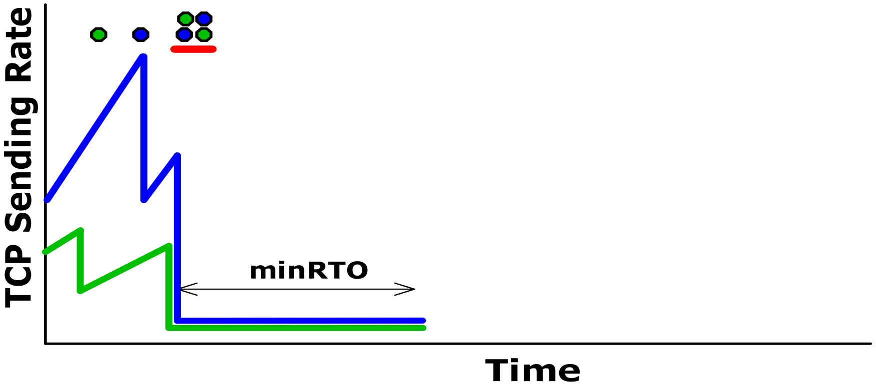


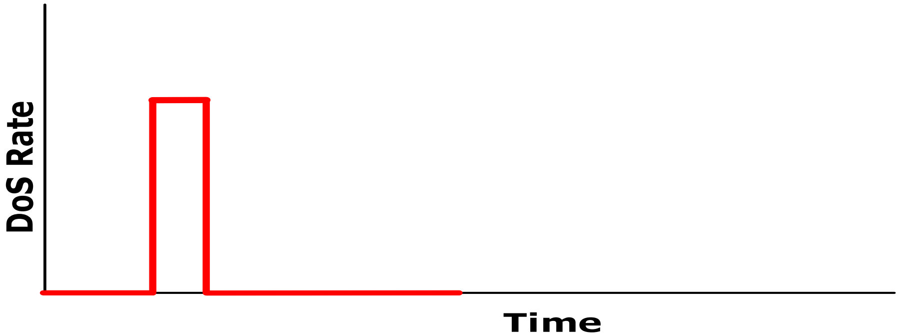


- A short burst (~RTT) sufficient to create outage
- Outage – event of correlated packet losses that forces TCP to enter RTO mechanism

The Shrew Attack

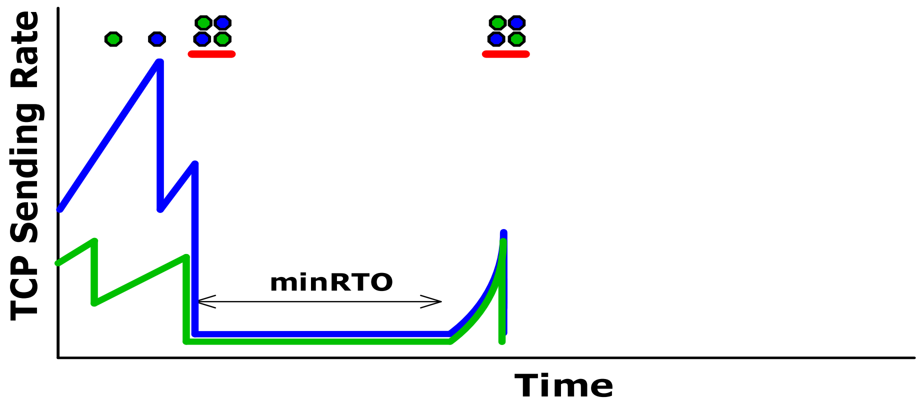


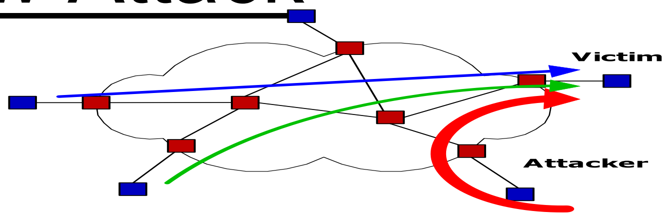


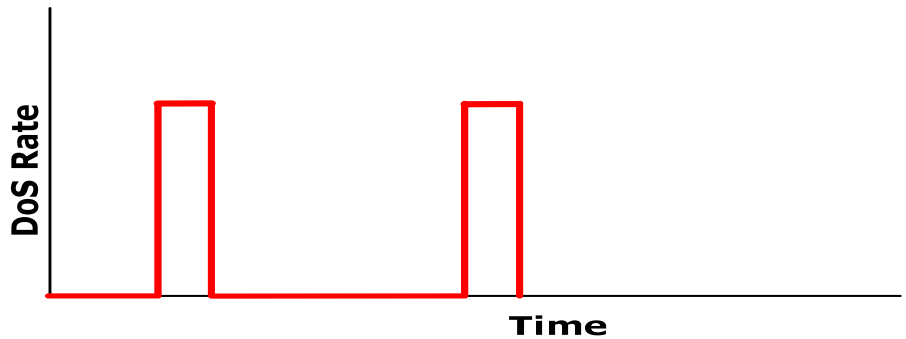


- The outage synchronizes all TCP flows
  - All flows react simultaneously and identically
    - backoff for minRTO

The Shrew Attack



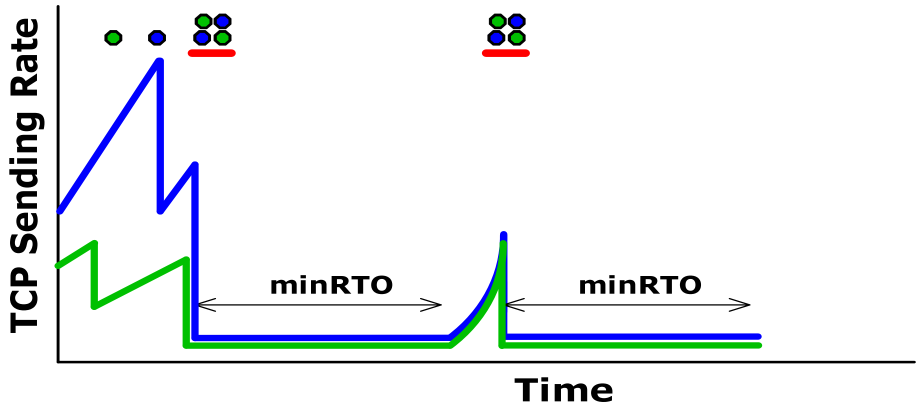


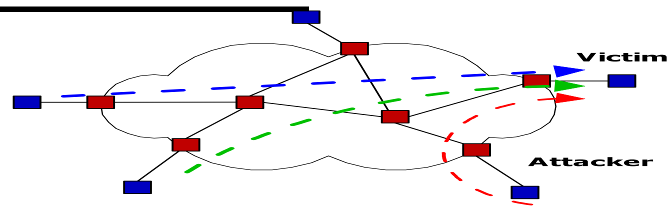


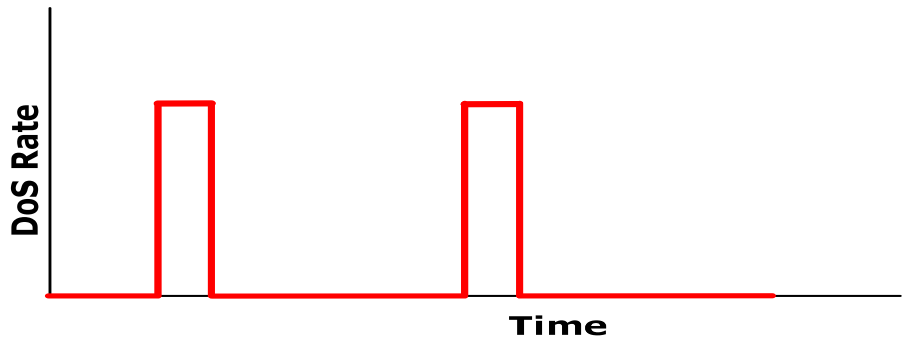
- Once the TCP flows try to recover – hit them again

- Exploit protocol determinism

The Shrew Attack







- And keep repeating…
- RTT-time-scale outages inter-spaced on minRTO periods can deny service to TCP traffic

Shrew Principles

- A single RTT-length outage forces all TCP flows to *simultaneously* enter timeout
  - All flows respond *identically* and stop for the minRTO period
  - Most invoke slow start afterwards
- Shrews exploit protocol *determinism*, and repeat the outage after each minRTO period
- Periodic outages *synchronize* TCP flows and deny their service
- Outages occur relatively slowly and can be induced with low average rate

Difficulties of dealing with DoS Attacks

- Distinguish DoS attack from flash crowd
  - Can be very hard if the attack is sophisticated
- Prevent damage
  - Distinguish (if possible) attack traffic from legitimate traffic
  - Rate limit attack traffic
- Stop attack
  - Identify attacking machines, not easy if source address is spoofed
  - Shutdown attacking machines
  - Usually done manually, requires cooperation of ISPs
- Identify persons responsible
  - Very difficult
  - Usually done manually, requires cooperation of ISPs

Prevent Damage

- Use fair queuing to limit damage if attack traffic is not distinguishable
- Prevent an attacker from sending at 10Mb/s and hurting a user sending at 1Mb/s
- Does not prevent 10 attackers from sending at 1Mb/s and hurting a user sending a 1Mb/s

Stop Attack

- Need to identify attacking machines, but not easy if source addresses are spoofed

Some ideas to defeat spoofed source addresses:

- Ingress filtering
  - A domain's border router drop outgoing packets which do not have a valid source address for that domain
  - If universally used, could abolish spoofing
- IP Traceback
  - Routers probabilistically tag packets with an identifier
  - Destination can infer path to true source after receiving enough packets

In Closing...

- You can now explain to your Mom and Dad how the Internet works!!
- Application level
  - Domain name/IP address conversion, socket programming (TCP and UDP, client/server), buffer overflow bug, worm, DoS attack
- Transport level
  - Reliability (sliding window, CRC, etc.), congestion control (AIMD)
- Network level
  - Hierarchical addressing, intra-domain routing (LS and DV), inter-domain routing (path vector, policies)
- Link level
  - Broadcast network access control (Aloha, CSMA/CD, WiFi), Ethernet spanning tree, weighted fair queuing
- Physical level
  - Bit encoding (NRZI, 4B/5B, etc), packet framing (bit and byte stuffing)