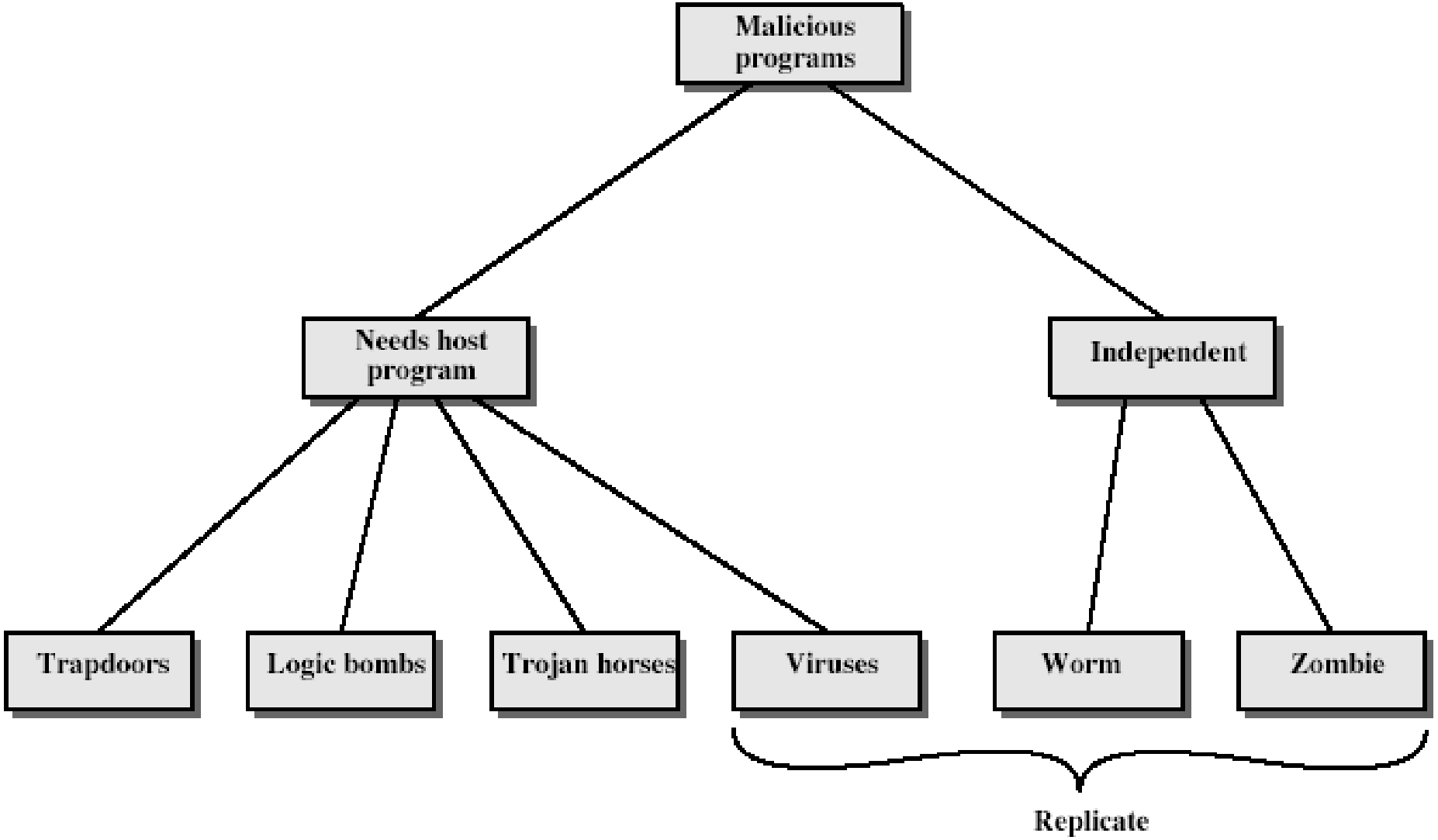
Outlines

- Mobile malcode Overview
- Viruses
- Worms

Mobile Malcode Overview

- Malicious programs which spread from machine to machine *without* the consent of the owners/operators/users
  - Windows Automatic Update is (effectively) consensual
- Many strains possible
  - Viruses
  - Worms
  - Compromised Auto-updates
    - No user action required, very dangerous

Malicious Software



Trapdoors (Back doors)

- Secret entry point into a program
- Allows those who know access bypassing usual security procedures
- Have been commonly used by developers
- A threat when left in production programs allowing exploited by attackers
- Very hard to block in O/S
- Requires good s/w development & update

Logic Bomb

- one of oldest types of malicious software
- code embedded in legitimate program
- activated when specified conditions met
  - eg presence/absence of some file
  - particular date/time
  - particular user
  - particular series of keystrokes
- when triggered typically damage system
  - modify/delete files/disks

Trojan Horse

- Programs that appear to have one function but actually perform another.
- Modern Trojan Horse: resemble a program that the user wishes to run - usually superficially attractive
  - eg game, s/w upgrade etc
- When run performs some additional tasks
  - allows attacker to indirectly gain access they do not have directly
- Often used to propagate a virus/worm or install a backdoor
- Or simply to destroy data

Zombie

- program which secretly takes over another networked computer
- then uses it to indirectly launch attacks
- often used to launch distributed denial of service (DDoS) attacks
- exploits known flaws in network systems

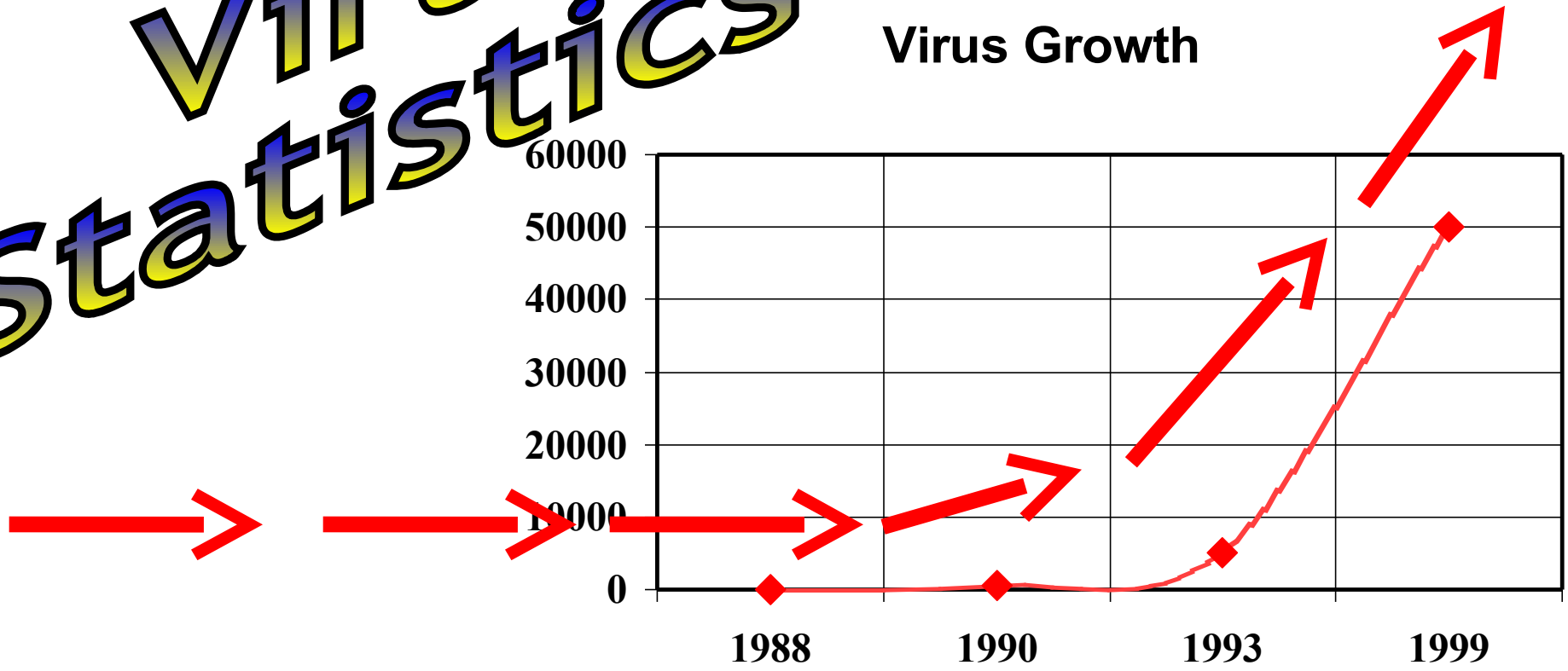
Outlines

- Mobile malcode Overview
- Viruses
- Worms

Viruses

- Definition from RFC 1135: *A virus* is a piece of code that inserts itself into a host, including operating systems, to propagate. It cannot run independently. It requires that its host program be run to activate it.
- On execution
  - Search for valid target files
    - Usually executable files
    - Often only infect uninfected files
  - Insert a copy into targeted files
    - When the target is executed, the virus starts running
- Only spread when contaminated files are moved from machine to machine
- Mature defenses available

Virus Statistics



- 1988: Less than 10 known viruses
- 1990: New virus found every day
- 1993: 10-30 new viruses per week
- 1999: 45,000 viruses and variants

Source: McAfee

Virus Operation

- virus phases:
  - dormant – waiting on trigger event
  - propagation – replicating to programs/disks
  - triggering – by event to execute payload
  - execution – of payload
- details usually machine/OS specific
  - exploiting features/weaknesses

Anatomy of a Virus

- Two primary components
  - Propagation mechanism
  - Payload
- Propagation
  - Method by which the virus spreads itself.
  - Old days: single PC, transferred to other hosts by ways of floppy diskettes.
  - Nowadays: Internet.

Structure of A Virus

```
virus() {
  infectExecutable();
  if (triggered()) {
     doDamage();
  }
  jump to main of infected program;
}

void infectExecutable() {
  file = choose an uninfected executable file;
  prepend V to file;
}

void doDamage() { ... }
int triggered() { return (some test? 1 : 0); }
```

Virus Infectables

- Executable files: .com, .exe, .bat
- Macros
  - With macro languages the line between pure data files and executable files is blurring
  - An infected file might be attached to an E-mail
  - E-mail programs may use other programs (e.g., word) with macros to display incoming mail
- System sector viruses
  - Infect control sectors on a disk
    - DOS boot sectors
    - Partition (MBR) sectors
  - System sector viruses spread easily via floppy disk infections

Variable Viruses

- Polymorphic viruses
  - Change with each infection
    - Executables virus code changing (macros: var name, line spacing, etc.)
    - Control flow permutations (rearrange code with goto's)
  - Attempt to defeat scanners
- Virus writing tool kits have been created to "simplify" creation of new viruses
  - Current tool kits create viruses that can be detected easily with existing scanner technology
  - But just a matter of time …

Virus Detection/Evasion

- Look for changes in size
- Check time stamp on file
- Look for bad behavior
  - False alarm prone
- Look for patterns (byte streams) in virus code that are unique
- Look for changes in file checksum
- Compression of virus and target code
- Modify time stamp to original
- Do bad thing insidiously
- Change patterns – polymorphism
- Rearrange data in the file
- Disable anti-virus programs

Internet checksum

Goal: detect "errors" (e.g., flipped bits) in transmitted segment (note: used at transport layer *only*)

Sender:

- treat segment contents as sequence of 16-bit integers
- checksum: addition (1's complement sum) of segment contents
- sender puts checksum value into UDP checksum field

Receiver:

- compute checksum of received segment
- check if computed checksum equals checksum field value:
  - NO - error detected
  - YES - no error detected. *But maybe errors nonetheless?* More later ….

More on Virus Detection

- Scanning
  - Depend on prior knowledge of a virus
  - Check programs before execution
  - Need to be regularly updated
- Integrity Checking
  - Read entire disk and record integrity data that acts as a signature for the files and system sectors
  - Use cryptographic computation technique instead of simple checksum

More on Virus Detection

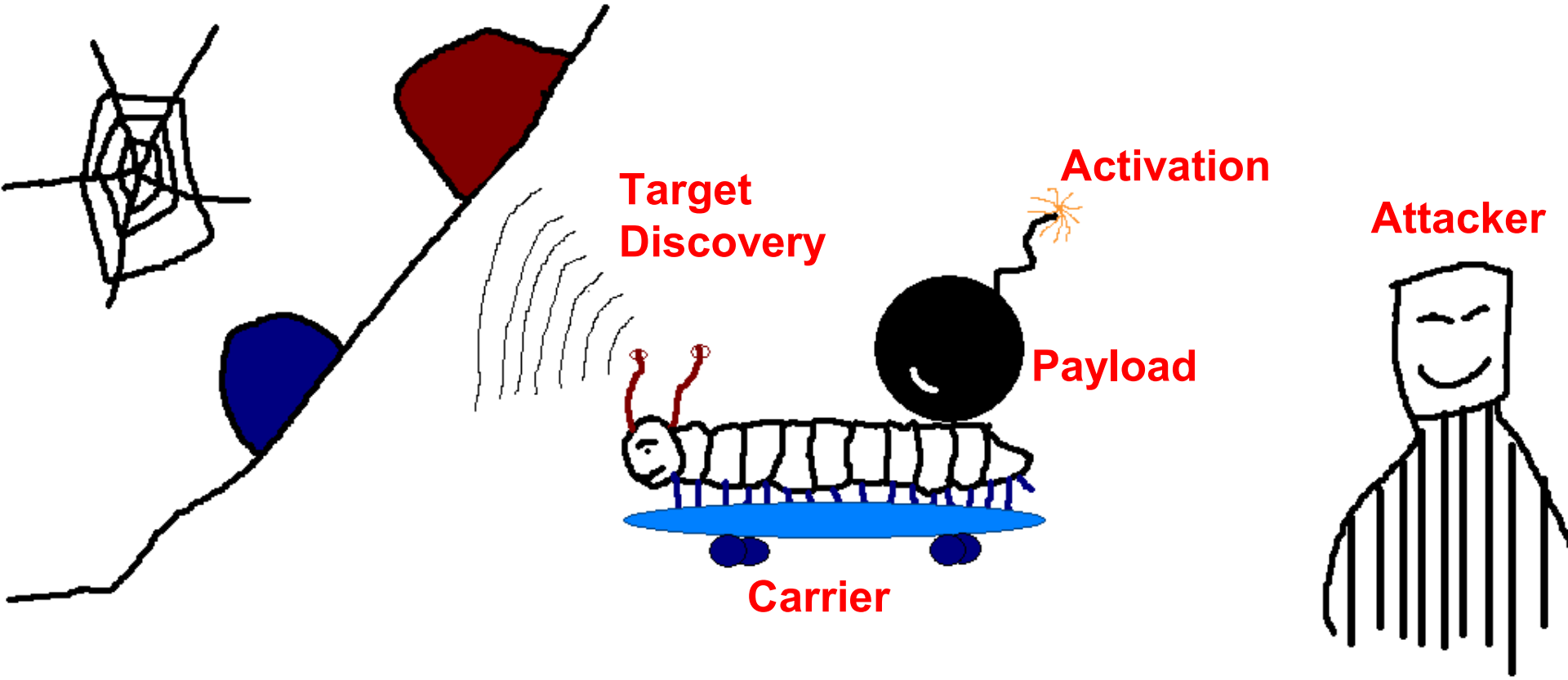
- Interception
  - Monitoring for system-level routines that perform destructive acts
  - Good for detecting logic bomb and Trojan horse
  - Cannot depend entirely upon behavior monitors as they are easily bypassed.
- Combination of all three techniques can detect most viruses

Virus Recovery

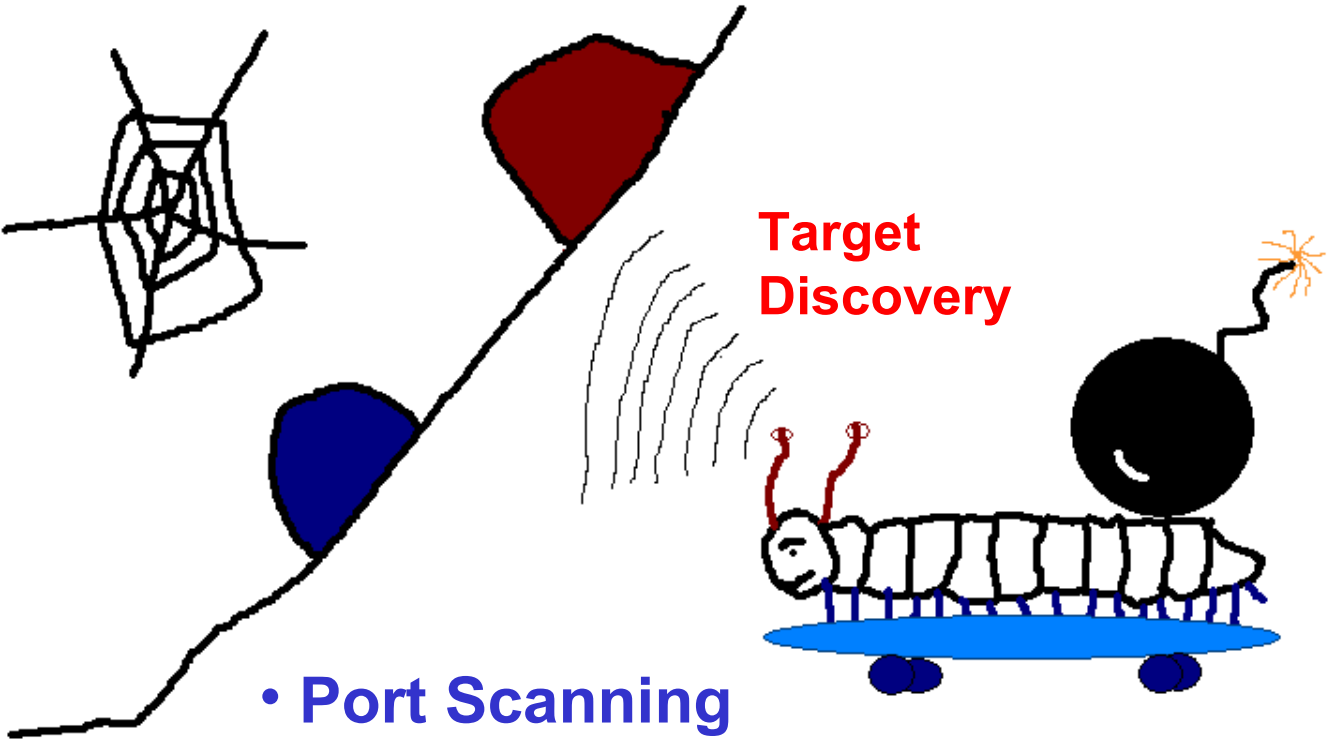
- Extricate the virus from the infected file to leave the original behind
- Remove the redirection to the virus code
- Recover the file from backup
- Delete the files and move on with life

Worms

- Autonomous, active code that can replicate to remote hosts without any triggering
  - Replicating but not infecting program
- Because they propagate autonomously, they can **spread much more quickly** than viruses!
- Speed and general lack of user interaction make them the most significant threats



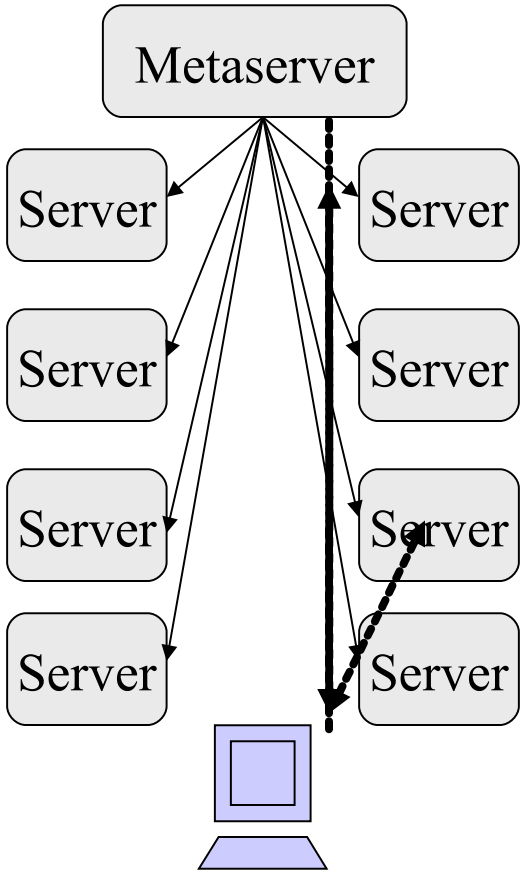
Worm Overview



- **Port Scanning**
  - Sequential: working through an address block
  - Random
- **Target Lists**
  - Externally generated through Meta servers
  - Internal target list
  - Passive worms

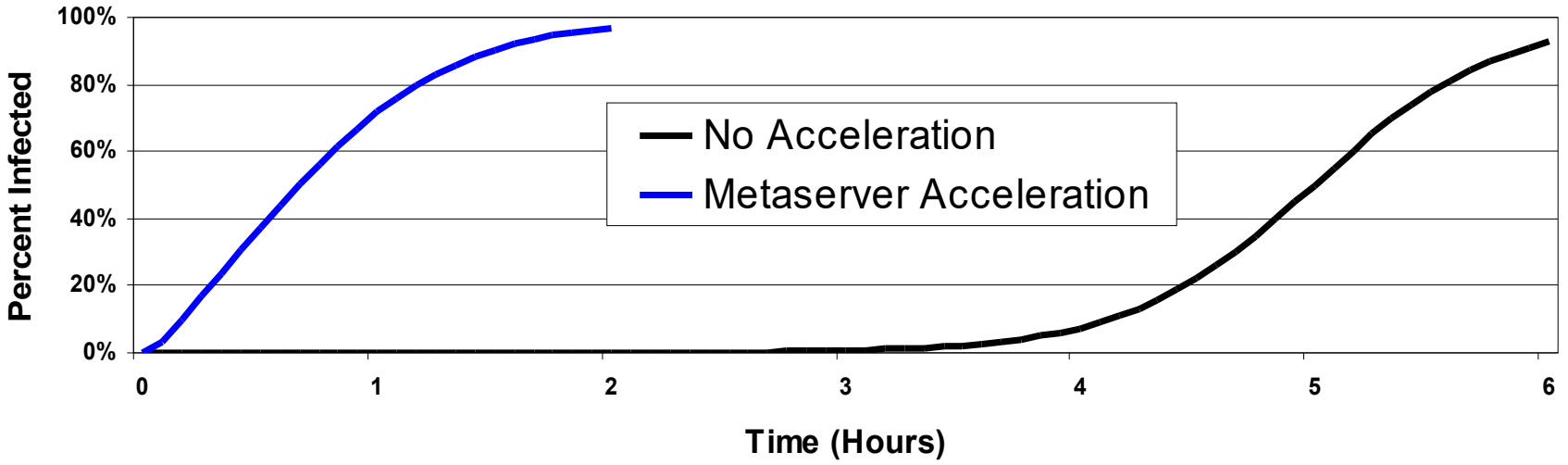
External Target Lists: Metaserver Worms

- Many systems use a "metaserver", a server for information about other servers
  - Games: Use as a matchmaker for local servers
  - Google: Query google to find web servers
  - Windows Active Directory: Maintains the "Network Neighborhood"
- Worm can leverage these services
  - Construct a query to find new targets
  - Each new victim also constructs queries
    - Creates a divide-and-conquer infection strategy
- Original strategy, not yet seen



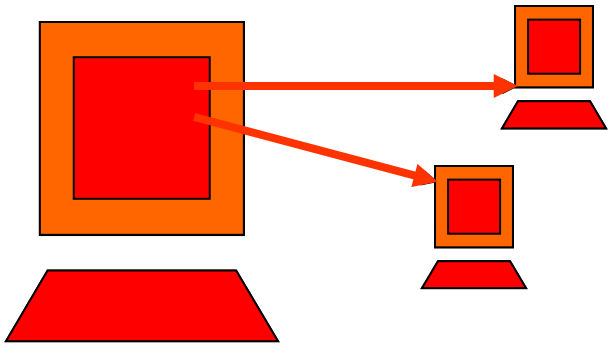
How Fast Are Metaserver Worms?

- Game Metaserver: Use to attack a small population (eg, all Half-Life servers)
  - ~1 minute to infect all targets
- Google: Use to enhance a scanning web worm
  - Each worm conducts initial queries to find URLs



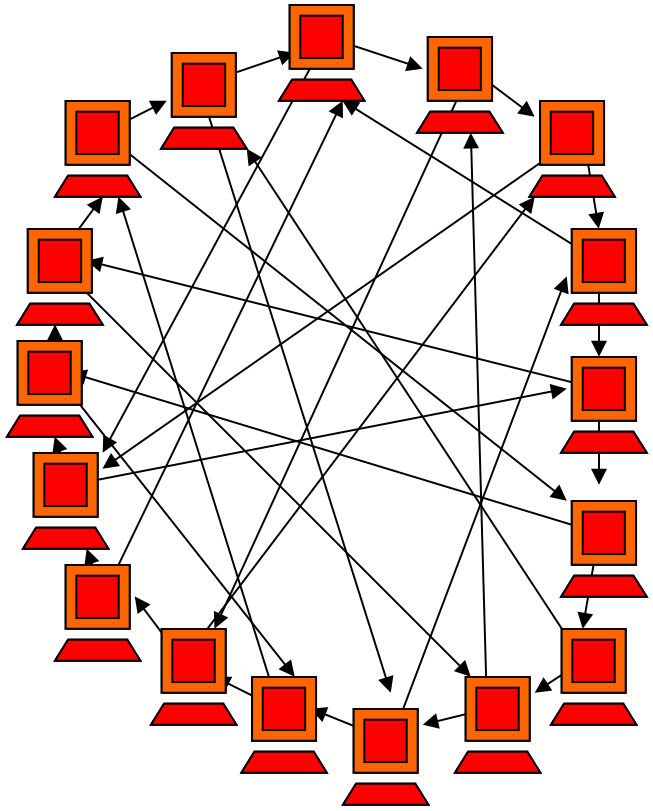
Internal Target Lists: Topological Information

- Look for local information to find new targets
  - URLs on disk and in caches
  - Mail addresses
  - .ssh/known_hosts
- Ubiquitous in mail worms
  - More recent mail worms are more aggressive at finding new addresses
- Basis of the Morris worm
  - Address space was too sparse for scanning to work



How Fast are Topological Worms?

- Depends on the topology *G = (V, E)*
  - Vulnerable machines are vertices, edges are local information
  - Time to infect is a function of the shortest paths from the initial point of infection
- Power law or similar graph (KaZaA)
  - Depends greatly on the parameters, but generally very, VERY fast



Some Major Worms

Worm	Year	Strategy	Victims	Other Notes
Morris	1988	Topological	6000	First major autonomous worm. Attacked multiple vulnerabilities.
Code Red	2001	Scanning	~300,000	First recent "fast" worm, 2nd wave infected 360,000 servers in 14 hours
CRClean	2001	Passive	none	Unreleased Anti-Code-Red worm.
Nimda	2001	Scanning IIS, Code Red 2 backdoor, etc	~200,000	Local subnet scanning. Effective mix of techniques
Scalper	2002	Scanning	<10,000	Released 10 days after vulnerability revealed