# Superworms

# (How to 0wn the Internet in 30 Seconds)

**Roger Dingledine**

## Overview (part one)

Background: worms in practice

Mechanisms for scanning

Flash worms (fast, obvious)

Contagion worms (slow, subtle)

## Overview (part two)

How to get a worm network without deploying a worm

How to control your worm network

What you can do with your worm network

Are there any defenses?

## Background: worms in history

Morris worm (November 1988)

Code Red V1 and V2 (July/August 2001)

Nimda (September 2001)

Sapphire/Slammer (January 2003)

## Morris worm: November 1988

Spread by buffer overflows in fingerd, sendmail, rexecd.

Uses local host tables (hosts.equiv, .rhosts) to learn about more targets.

Also guessed passwords based on /etc/passwd

60,000 Internet hosts at this time.

## Code Red: July 2001 / August 2001

Spread by buffer overflow on Microsoft IIS webserver

Added a root-level backdoor to the system

Code Red I: spreads by random scanning

Code Red II: by localized random scanning

## Nimda: Sept 2001

Spreads by:

- random active probing (like before)
- bulk emailing itself to victim's address book
- copying itself across network shares
- adding exploit code to webservers, to infect web clients.
- scanning for backdoors left behind by code red II and other worms

## Sapphire/slammer (January 2003)

400 byte UDP payload to Microsoft SQL server.

- (Code red was 4 kilobytes and TCP.)
- Sapphire didn't need to stop for a response!

Doubled in size every 8 seconds

- (code red doubled every 37 minutes)

It had 90% of vulnerable hosts (about 75000) in 10 minutes.

55 million scans per second after 3 minutes.

## Sapphire/slammer (January 2003)

Cancelled airline flights, interfered with elections, caused ATM failures.

- (Actually, no. That was just hype.)

Really trashed the Internet for an hour or so. Fortunately, it was a UDP packet to a mostly unused port, so it was easy to filter.

Filtering did not affect infection — just controlled the damage afterward.

## Mechanisms for scanning

Random (code red)

Localized random (code red II)

Hit-list scanning

Permutation scanning

Topology-based scanning

## Hit-list scanning

Build a hit-list by:

- stealthy scans
- distributed scanning
- dns searches
- spiders
- public surveys, e.g. netcraft
- just listen: vulnerable machines will come to you.

Then divide-and-conquer the list.

Moves from 8 hours to scan 300,000 hosts, to 15 minutes.

## Permutation scanning

Each instance comes with a pseudo-random 32-bit permutation

All nodes share the same seed

Nodes start at random places in the permutation. When you hit another compromised node, you know that fraction is done.

## Permutation scanning

This way you can stop scanning as soon as most victims are found.

You reseed the permutation periodically, to get a different order.

- Otherwise nodes can "protect" certain ranges of the permutation by pretending to be compromised.

## Topology-based scanning

Morris worm did this

- it needed to, because of the sparse address space

p2p networks: instant peer lists.

- (aim buffer-overflow story)

if you hit web clients, go through their cache for a list of web servers

## Dual-mode worms

Permutation scan, but instead of stopping, move to a rarer vulnerability

Much more efficient than multi-mode (e.g. going after both at once)

## Flash worms

Netcraft says there are 12.6 million webservers on the Internet.

...and it knows what software each of them runs.

That's just a 48M hitlist.

## Flash worms (cont)

Example: about 3 million of those are IIS servers.

With divide-and-conquer, the tree is only 7 layers deep

Assuming they're all on DSL (256kb) or better, we could flash them all in less than 30 seconds.

## Counterattacks have been seen

Some of us ran default.ida scripts of our own

When probed by code red II
- attacked back
- disabled the server
- and rebooted the machine.

Lesson: so close the exploit when you're in.

## Contagion worms

The worms so far are fast and loud

How about a worm that has no suspicious behavior?

## Contagion worms (cont)

First, get an exploit for common web browsers, and an exploit for common web servers.

Then set up a porn website, and wait. Infect browsers and send both exploits.

## Contagion worms (cont)

Ok, so that requires two exploits.
How about only one?

P2P overlay networks are perfect for this.

Gnutella, Kazaa, etc have tens of millions of machines.

## (Break)

Next: So you've got a worm network. What now?

## How to control it

Previous control approaches:
- gurer (mail worm) used irc to coordinate ddos
- w32/sonic had a website where the worm looked for updates
- stacheldraht used encrypted communication for direct control

All of these are central points of failure / inefficient

## How to control it

The trick is to use your worm overlay network to distribute updates.

Turns out that by default after a permutation scan each worm will know about 4 other worms.

Each time you rescan you'll learn two more.

## How to control it

There are efficient p2p lookup systems under design right now.

E.g., Chord here at MIT.

So have your nodes create a Chord network. Fast search, fast propagation.

## How to control it

Pay attention to the details:

- Code red targeted the IP of whitehouse.gov, rather than the DNS name

Payloads can (will) be analyzed

## Defenders must get good at analyzing binaries

Lesson: get good at obfuscating binaries.

- e.g., add extra unused payload sections, as decoys for your worm's secondary payloads.

## People research your worm

Most worms are identified and discussed on bugtraq and a few other critical mailing lists.

Lesson: read the list and see how your worm is doing.

or:

Lesson: DDoS that list when you launch the worm.

## Hijacking deployed networks

Deployed worm networks:
(Code Red1 vs 2 vs Nimda...)

Other widespread networks:

- Windowsupdate (hosted on Akamai!)
- Akamai
- Kazaa
- Blizzard's diablo
- Everquest
- Debian
- Ximian's redcarpet

## Hijacking deployed networks

If the autoupdate relies on dns, hijack dns?

SSL or just ip-based authentication?

Signed code? might need a private key.

Single point of failure gives you millions of machines. how secure?

## Uses for the worm network

One million hosts? ten million hosts? More?

Distributed denial of service.

E-commerce, news sites, milnet infrastructure, routers, root name servers

- (whispering / screaming)

## Uses for the worm network (2)

Use compromised cablemodem hosts for ultra-robust websites

...and advertise them via spam.

A company in Poland does this.

## Uses for the worm network (3)

Each of those machines has interesting files on it.

Passwords, credit card numbers, address books, archived email, patterns of user activity, mp3's.

Information warfare / terrorism:
Publish these documents?

## Uses for the worm network (4)

IP-based access controls:

- Research papers from publisher's websites
- Encyclopedias, dictionaries, ...
- MIT-only info
- Send bulk mail because MIT's mail server trusts your IP
- Print?

- Access other parts of a company's intranet?
- Authenticate to various servers?

## Uses for the worm network (5)

Anonymizer: use dozens of nodes as relays, like in Sneakers.

## Uses for the worm network (6)

Man in the middle:

Be any website you want to be!

Make the Internet look however you want!

## Uses for the worm network (7)

Perfect cracker network

Distribute scans.
Bootstrap to a second worm network.

Distributed computation -- brute force windowsupdate's key

You always have a zombie near your target, to

- monitor your victim
- attack from locally trusted network

## Uses for the worm network (8)

Beat the Chinese firewall.

Infect everybody in China.

"I was reading which news page? Gosh. Must have been the worm."

## Uses for the worm network (9)

Unstoppable spam delivery

Talk about personalized --
you own the target's computer.

## Uses for the worm network (9b)

Heck, you already have the guy's credit card number. Just go buy stuff for him directly!

## Uses for the worm network (10)

Gather cool stats.

Ever wanted to learn how the Internet works really?

## Uses for the worm network (11)

Crash the Internet

(over and over and over)

## Are there any defenses?

Better security design (rilight)

The Berkeley folk suggest a CDC analog for Internet worms.

Honeypots? Honeynets? Early warning systems?
- Lesson: disable the honeynets

## Are there any defenses?

Universally deployed new-hosts-per-second throttling.

Newer switching technology lets routers filter by content in the tcp stream. This is a good start.

Di-culture rather than monoculture for P2P systems.

---