The Worms Crawl In
The Worms Crawl Out

15-744
David Andersen

Credits

- Parts of these slides are heavily inspired by Stefan Savage's NDSS 2005 talk
- (Some bits are stolen verbatim)
- See
  - http://www.cs.ucsd.edu/~savage/papers/Intern

    for original, much prettier, slides

Threat Model

Traditional

- High-value targets
- Insider threats

Worms & Botnets

- Automated attack of *millions* of targets
- Value in aggregate, not individual systems
- Threats: Software vulnerabilities; naïve users

... and it's profitable

- Botnets used for
  - Spam (and more spam)
  - Credit card theft
  - DDoS extortion
- Flourishing Exchange market
  - Spam proxying: 3-10 cents/host/week
  - 25k botnets: $40k - $130k/year
  - Also for stolen accounts, compromised machines, credit cards, identities, etc. (be worried)

Why is this problem hard?

- **Monoculture**: little “genetic diversity” in hosts
- **Instantaneous transmission**: Almost entire network within 500ms
- **Slow immune response**: human scales (10x-1Mx slower!)
- **Poor hygiene**: Out of date / misconfigured systems; naïve users
- **Intelligent designer** ... of pathogens
- **Near-Anonymitity**

Example Outbreak: SQL Slammer (2003)

- Single, small UDP packet exploit (376 b)
- First ~1min: classic random scanning
  - Doubles # of infected hosts every ~8.5sec
  - (In comparison: Code Red doubled in 40min)
- After 1min, starts to saturate access b/w
  - Interferes with itself, so it slows down
  - By this point, was sending 20M pps
  - Peak of 55 million IP scans/sec @ 3min
- 90% of Internet scanned in < 10mins
- Infected ~100k or more hosts

Digression: Fast Worms

- How fast could a *really* fast worm spread?
- **Localized scanning**: Preferential scanning of “nearby” hosts
  - Host density not uniform
- **Multi-vector worms**: Can find more vulnerable hosts
- **Hit-list scanning**: Pre-identify many “seed” machines; divide & conquer
  - Scanning; DNS; spiders; surveys; passive

Fast Worms, Cont'd.

- **Permutation Scanning**
  - Don't scan purely randomly; divide scan space intelligently among worms
  - Simple permutation -> coordinated behavior
- How fast?
  - Easy: A couple of minutes for the entire 'net
  - Pre-scanning: 10s of seconds?
  - Pre-scanning, UDP, insane effort: < 2sec?
    - (follow-on paper to the one we're reading)
- Exponential growth is a pain...

An Ounce of Prevention?

- Get rid of the vulnerabilities (testing, modeling, proving, engineering, etc.)
  - Soundness, completeness, usability...
- Permute vulnerabilities (e.g., address space randomization) – makes it harder to compromise
- Block traffic (firewalls): helps, but many worms slipped inside firewalls. Only takes *one* vulnerable computer wandering between in & out or multi-homed, etc.

We keep trying, but worms keep worming

Hygiene

- Keep vulnerable hosts off network
  - Must scan / etc., before connecting
  - Some commercial products do this
- Helps, but not entire problem
  - 0-day worms
  - Incomplete vuln. databases
  - etc.

Containment

- Slow down scan rate
  - Allow hosts limited # of new contacts/sec.
  - Can slow worms down, but they do still spread
- Quarantine
  - Detect worm, block it

Reactive “Immune System”

- **Reaction time**: How long to detect & react?
- **Containment strategy**: How the behavior is (1) identified; and (2) stopped
- **Deployment strategy**: Who participates? End-hosts? Routers?

Strategies

- Reaction time: seconds?
- Containment:
  - Address blacklisting (more false positives make it harder to be aggressive)
  - Content filtering
- Deployment
  - Top 40 ISPs provide decent containment
  - But really, need lots and lots of nets

Detection

- Behavior: Contacting 1000s of hosts, etc.
- Honeypots: Hosts *nobody* should contact
  - Traffic assumed to be malicious
  - Replies to traffic, permits real/pretend infection
  - Virtual machines / honeyd / etc.
- After detection: *signature inference*

Signature Inference

- Content prevalence: Autograph, EarlyBird, etc.
  - Assumes *some* content invariance
  - Pretty reasonable for starters.
  -
  - Goal: Identify "attack" substrings
    - Maximize detection rate
    - Minimize false positive rate

Common strings

- Definition of substring:
  - Byte range, protocol, port (why?)
- First: identify common *packets*
  - Hash and count?
    - Saw from Snoeren – still has pretty large memory requirements
  - "heavy-hitter" identification: only need the common stuff, so sampling should work well
    - This paper uses "multi-stage" filters: basically a counting bloom filter like we talked about last time

Common Substrings

- Fix length as beta (small)
- Use Rabin Fingerprinting to efficiently hash
  - Shift values in & out of polynomial
  - O(N) computation for O(N) bytes
- Reduce the # by sampling
  - But must *deterministically* sample (why?)
  - Sample only values whose low-order hash bits are zero (or somehing else)
  - This trick is used for lots of things...

Finding the Guilty

- Address Dispersion
  - Scanning worms will cover more addresses than most “legitimate” content
  - How many distinct sources/dests
- EarlyBird technique: scaled bitmap
  - 1/(2^n)th of hash space -> bitmap
    - e.g., hash(src) -> [0, 63], bitmap [0,31]
  - When bitmap fills, double hash size
    - hash(src) -> [0, 127]; increment scale counter
  - Small tweak: Keep 2 older bitmaps, correct for double counting

False Negatives in EB

- False Negatives
  - Very hard to prove...
  - Earlybird detected all worm outbreaks reported on security lists over 8 months
  - EB detected all worms detected by Snort (signature-based IDS)
    - And some that weren't

False Positives in EB

- Common protocol headers
  - HTTP, SMTP headers
  - p2p protocol headers
- Non-worm epidemic activity
  - Spam
  - BitTorrent (!)
- Solution:
  - Small whitelist...

Distributing Signatures

- No time; see Dawn Song's work for some pointers on distributing verifiable signatures
  - Requires access to vulnerable binary
  - Creates signatures based on *actual* vulnerability, not content prevalence. Can be better – but slower – than prevalence metrics
- Have to get the signatures sent around fast
- Trust?

Unrelated: Presentations

- See David Patterson's “How to Give a Bad Talk” advice...
- Be neat
- Be concise! <= 7 bullets/slide, LARGE FONTS
  - Talk about the most important things
  - Your talk is an *advertisement* for your paper, not a complete summary. You MUST downsample, so do it well.
- Use pictures! Words + words == mental confict; words + pictures = reinforcement
- Use color, italics, bold to emphasize (and do it *consistently*)
- Make eye contact with audience
- Practice your talk! Even for this class