Security

Robert Grimm
New York University
Traditionally, security focuses on
  - Protection (authentication, authorization)
  - Privacy (encryption)

But, the Internet adds/amplifies several threats
  - (Distributed) Denial of service attacks
  - Viruses
  - Worms

For today, we are looking at
  - How to identify origin of denial of service attacks (or how to find those evil-doers)
  - How to build better worms (don’t try this at home…)
IP Traceback
In General

- Assumptions
- Goal
- Transformations
- Strategies
IP Traceback
Assumptions

- Packets may be addressed to more than one host
- Duplicate packets may exist in the network
- Routers may be subverted (though, not often)
- Attackers may be aware of being traced
- Routing behavior may be unstable
- Packet size should not grow b/c of tracing
- End hosts may be resource constrained
- Traceback is an infrequent operation
IP Traceback

Goal

- Identify source of any packet
  - Ingress point to traceback-enabled network
  - Actual host or network
  - Compromised router(s)

- Consequently, we are really looking for *path*!
IP Traceback
Transformations

- Even under normal operation, packets are not constant
  - Trivial transformations
    - Time-to-live
    - Checksum
  - Packet encapsulation
    - Packet fragmentation, NAT, IP-over-IP, IPsec
  - Packet generation
    - ICMP Echo Request → ICMP Echo Response
    - Multicast
IP Traceback Strategies

- Link testing
  - Flood candidate network links and watch for variations

- Auditing/logging
  - End-host based
    - Mark packets in network
    - Store and analyze on endhosts
  - Infrastructure based
    - Store packets in network
    - Analyze in network

- Single-Packet IP Traceback
  - Infrastructure based logging (with some clever twists…)

Single-Packet IP Traceback

- Basic idea: Log bloom-filtered packet digests
  - Why digests?
    - 0.00092% collision rate for 28-byte prefix
    - Need additional state for fragmentation, NAT, ICMP, IP-in-IP, IPsec
  - Why bloom-filtered?
Bloom-Filters and Hash Functions

- Need *family* of hash functions
  - Each function uniformly distributes hashes
  - Collisions in one function are independent of collisions in other functions
    - Known as universal hash families
  - Functions are efficient to compute
    - We are targeting routers, after all
Source Path Isolation Engine (SPIE)

- Per router data generation agent
  - Generates digests and stores them in time-stamped tables
- Per region SPIE collection and reduction agent
  - Generates attack graphs for region
- Per network SPIE traceback manager
  - Accepts and authenticates requests
  - Collects and coalesces per region attack graphs
Traceback Processing

- Provide packet, victim, time
  - Packet to reproduce digest
  - Victim to identify starting point
  - Time to identify table
- Complication: Transformations
  - Use transform lookup tables
    - Map digest to irrecoverable packet data
  - Typically processed on control and not fast path
  - Rely on transformations being infrequent
Single-Packet IP Traceback Analysis

- False positives
  - Based on analysis and simulation
- Time and memory utilization
  - Goal: collect 1 minute worth of digests
    - 0.5% of total link capacity
      - Four OC-3 (155.52 Mbps) links: 23MB
      - 32 OC-192 (9.952 Gbps) links: 12 GB
    - One 16 Mb SRAM for current table
    - Backed by additional SDRAM
Single-Packet IP Traceback Discussion

- Deployment
- Vulnerabilities
  - Denial of service
  - Flow amplification
  - Information leakage
  - Transformations
And Now for Something Completely Different…

Yeah, Right …
Worms, Worms, Worms

- Install payload on as many machines as possible and then...
  - Launch distributed denial of service attacks
  - Access sensitive information (financial, medical, …)
  - Corrupt information

30 minutes of Sapphire

Code Red I, II, Nimda
Analysis of Code Red I

- Attacks Microsoft IIS web servers
- Launches 99 threads, trying to compromise servers at random IP addresses
  - Version 1 has bug in random number generator
    - Fixed seed → Linear spread
  - Version 2
    - Fixes bug
    - Does not deface web sites
    - Launches DDOS attack against IP address of www.whitehouse.gov
Random Constant Spread Model (RCS)

- N: # of vulnerable machines
- K: initial compromise rate
- T: time of incident
- a: proportion of compromised machines

\[ a = \frac{e^{K(t-T)}}{1 + e^{K(t-T)}} \]
Better Worms Practice

- Code Red II
  - Same vulnerability, different payload (root backdoor)
  - Localized scanning strategy favoring local machines
- Nimda
  - Combines several techniques into one worm
    - Microsoft IIS web servers
    - Email attachments
    - Network shares
    - Web pages
    - Backdoors left behind by Code Red II and “sadmind”
Better Worms
Theory: Hit-List Scanning

- Helps with “getting off the ground”
- Collect list of potentially vulnerable machines, start with them, splitting list between worm copies
  - Stealthy scans
  - Distributed scans
  - DNS searches
  - Spiders
  - Public surveys
  - Just listen (think peer-to-peer networks)
Better Worms
Theory: Permutation Scanning

- Avoid repeatedly probing same machine
- Start scanning machines after your own address, using a common pseudo random permutation
  - Still looks like a random scan!
- Stop scanning when several machines already infected
  - Optionally, use new permutation after some waiting time
- As optimization, split permutation into ranges
Better Worms
Theory: Simulation of Spread

- “Warhol worm”
  - Uses hit-list and permutation scanning

Comparison
Calibration
Closeup
Better Worms
Theory: Alternatives to Hit-Lists

- Topological scanning
  - Use local information
    - Peers, email addresses, URLs

- Flash worms
  - Create exhaustive list of all vulnerable hosts
  - Distribute list
    - By dividing list repeatedly
    - Storing chunks on well-connected servers
  - Spread may initially be limited by size of list
    - Start across high-bandwidth links
Better Worms
Back to Reality: Sapphire

- We don’t necessarily need all these techniques
- Microsoft SQL Server + one 404-byte UDP packet are good enough!!!
  - Even though random number generator is broken
  - Even though port is atypical and thus easily blocked
More Bad News: Stealth Worms

- Quickly spreading worms are easily detected
- Alternatively, spread real slowoooooooooooowly
  - Almost no peculiar communication patterns
- Strategy
  - Pair of exploits E(server) and E(client) for popular web server and client
    - Hmmmm, which company comes to mind…
  - Alternatively, target peer-to-peer systems
    - Only need one exploit as all machines run the same software
Stealth Worms
Why Attack Peer-to-Peer Systems?

- Interconnect to many peers
- Often transfer large files
- Less likely to be monitored
- Typically run on less secure user machines
- Transfer “grey” content
- Are pretty popular
  - KaZaA: 9 to 30 million users…
What to Do?
Cyber-center for Disease Control!

- Identify outbreaks
- Rapidly analyze pathogens
- Fight infections
- Anticipate new vectors
- Proactively devise detectors
- Resist future threats
What Do You Think?