

CSci 5271
Introduction to Computer Security
Day 22: Malware and Denial of Service

Stephen McCamant
University of Minnesota, Computer Science & Engineering

Outline

Intrusion detection systems
Malware and the network
Announcements intermission
Denial of service and the network
Bonus: anonymity overlays

Signature matching

- *Signature* is a pattern that matches known bad behavior
- Typically human-curated to ensure specificity
- See also: anti-virus scanners

Anomaly detection

- Learn pattern of normal behavior
- “Not normal” is a sign of a potential attack
- Has possibility of finding novel attacks
- Performance depends on normal behavior too

Recall: FPs and FNs

- False positive: detector goes off without real attack
- False negative: attack happens without detection
- Any detector design is a tradeoff between these (ROC curve)

Signature and anomaly weaknesses

- Signatures
 - Won't exist for novel attacks
 - Often easy to attack around
- Anomaly detection
 - Hard to avoid false positives
 - Adversary can train over time

Base rate problems

- If the true incidence is small (low base rate), most positives will be false
 - Example: screening test for rare disease
- Easy for false positives to overwhelm admins
- E.g., 100 attacks out of 10 million packets, 0.01% FP rate
 - How many false alarms?

Adversarial challenges

- FP/FN statistics based on a fixed set of attacks
- But attackers won't keep using techniques that are detected
- Instead, will look for:
 - Existing attacks that are not detected
 - Minimal changes to attacks
 - Truly novel attacks

Wagner and Soto mimicry attack

- Host-based IDS based on sequence of syscalls
- Compute $A \cap M$, where:
 - A models allowed sequences
 - M models sequences achieving attacker's goals
- Further techniques required:
 - Many syscalls made into NOPs
 - Replacement subsequences with similar effect

Outline

Intrusion detection systems

Malware and the network

Announcements intermission

Denial of service and the network

Bonus: anonymity overlays

Malicious software

- Shortened to Mal...ware
- Software whose inherent goal is malicious
 - Not just used for bad purposes
- Strong adversary
- High visibility
- Many types

Trojan (horse)

- Looks benign, has secret malicious functionality
- Key technique: fool users into installing/running
- Concern dates back to 1970s, MLS

(Computer) viruses

- Attaches itself to other software
- Propagates when that program runs
- Once upon a time: floppy disks
- More modern: macro viruses
- Have declined in relative importance

Worms

- Completely automatic self-propagation
- Requires remote security holes
- Classic example: 1988 Morris worm
- "Golden age" in early 2000s
- Internet-level threat *seems* to have declined

Fast worm propagation

- Initial hit-list
 - Pre-scan list of likely targets
 - Accelerate cold-start phase
- Permutation-based sampling
 - Systematic but not obviously patterned
 - Pseudorandom permutation
- Approximate time: 15 minutes
 - "Warhol worm"
 - Too fast for human-in-the-loop response

Getting underneath

- Lower-level/higher-privilege code can deceive normal code
- Rootkit: hide malware by changing kernel behavior
- MBR virus: take control early in boot
- Blue-pill attack: malware is a VMM running your system

Malware motivation

- Once upon a time: curiosity, fame
- Now predominates: money
 - Modest-size industry
 - Competition and specialization
- Also significant: nation-states
 - Industrial espionage
 - Stuxnet (not officially acknowledged)

User-based monetization

- Adware, mild spyware
- Keyloggers, stealing financial credentials
- Ransomware
 - Application of public-key encryption
 - Malware encrypts user files
 - Only \$300 for decryption key

Bots and botnets

- Bot: program under control of remote attacker
- Botnet: large group of bot-infected computers with common "master"
- Command & control network protocol
 - Once upon a time: IRC
 - Now more likely custom and obfuscated
 - Centralized → peer-to-peer
 - Gradually learning crypto and protocol lessons

Bot monetization

- Click (ad) fraud
- Distributed DoS (next section)
- Bitcoin mining
- Pay-per-install (subcontracting)
- Spam sending

Malware/anti-virus arms race

- "Anti-virus" (AV) systems are really general anti-malware
- Clear need, but hard to do well
- No clear distinction between benign and malicious
- Endless possibilities for deception

Signature-based AV

- Similar idea to signature-based IDS
- Would work well if malware were static
- In reality:
 - Large, changing database
 - Frequent updated from analysts
 - Not just software, a subscription
 - Malware stays enough ahead to survive

Emulation and AV

- Simple idea: run sample, see if it does something evil
- Obvious limitation: how long do you wait?
- Simple version can be applied online
- More sophisticated emulators/VMs used in backend analysis

Polymorphism

- Attacker makes many variants of starting malware
- Different code sequences, same behavior
- One estimate: 30 million samples observed in 2012
- But could create more if needed

Packing

- ▣ Sounds like compression, but real goal is obfuscation
- ▣ Static code creates real code on the fly
- ▣ Or, obfuscated bytecode interpreter
- ▣ Outsourced to independent "protection" tools

Fake anti-virus

- ▣ Major monetization strategy recently
- ▣ Your system is infected, pay \$19.95 for cleanup tool
- ▣ For user, not fundamentally distinguishable from real AV

Outline

Intrusion detection systems

Malware and the network

Announcements intermission

Denial of service and the network

Bonus: anonymity overlays

Note to early readers

- ▣ This is the section of the slides most likely to change in the final version
- ▣ If class has already happened, make sure you have the latest slides for announcements

Outline

Intrusion detection systems

Malware and the network

Announcements intermission

Denial of service and the network

Bonus: anonymity overlays

DoS versus other vulnerabilities

- ▣ Effect: normal operations merely become impossible
- ▣ Software example: crash as opposed to code injection
- ▣ Less power than complete compromise, but practical severity can vary widely
 - Airplane control DoS, etc.

When is it DoS?

- Very common for users to affect others' performance
- Focus is on unexpected and unintended effects
- Unexpected channel or magnitude

Algorithmic complexity attacks

- Can an adversary make your algorithm have worst-case behavior?
- $O(n^2)$ quicksort
- Hash table with all entries in one bucket
- Exponential backtracking in regex matching

XML entity expansion

- XML entities (HTML `<t`) are like C macros

```
#define B (A+A+A+A+A)
#define C (B+B+B+B+B)
#define D (C+C+C+C+C)
#define E (D+D+D+D+D)
#define F (E+E+E+E+E)
```

Compression DoS

- Some formats allow very high compression ratios
 - Simple attack: compress very large input
- More powerful: nested archives
- Also possible: "zip file quine" decompresses to itself

DoS against network services

- Common example: keep legitimate users from viewing a web site
- Easy case: pre-forked server supports 100 simultaneous connections
- Fill them with very very slow downloads

Tiny bit of queueing theory

- Mathematical theory of waiting in line
- Simple case: random arrival, sequential fixed-time service
 - M/D/1
- If arrival rate \geq service rate, expected queue length grows without bound

SYN flooding

- SYN is first of three packets to set up new connection
- Traditional implementation allocates space for control data
- However much you allow, attacker fills with unfinished connections
- Early limits were very low (10-100)

SYN cookies

- Change server behavior to stateless approach
- Embed small amount of needed information in fields that will be echoed in third packet
 - MAC-like construction
- Other disadvantages, so usual implementations used only under attack

DoS against network links

- Try to use all available bandwidth, crowd out real traffic
- Brute force but still potentially effective
- Baseline attacker power measured by packet sending rate

Traffic multipliers

- Third party networks (not attacker or victim)
- One input packet causes n output packets
- Commonly, victim's address is forged source, multiply replies
- Misuse of debugging features

"Smurf" broadcast ping

- ICMP echo request with forged source
- Sent to a network broadcast address
- Every recipient sends reply
- Now mostly fixed by disabling this feature

Distributed DoS

- Many attacker machines, one victim
- Easy if you own a botnet
- Impractical to stop bots one-by-one
- May prefer legitimate-looking traffic over weird attacks
 - Main consideration is difficulty to filter

Outline

Intrusion detection systems

Malware and the network

Announcements intermission

Denial of service and the network

Bonus: anonymity overlays

Traffic analysis

- What can you learn from encrypted data? A lot
- Content size, timing
- Who's talking to who
 - countermeasure: anonymity

Anonymous remailers

- Anonymizing intermediaries for email
 - First cuts had single points of failure
- Mix and forward messages after receiving a sufficiently-large batch
- Chain together mixes with multiple layers of encryption
- Fancy systems didn't get critical mass of users

Tor: an overlay network

- Tor (originally from "the onion router")
 - <https://www.torproject.org/>
- An anonymous network built on top of the non-anonymous Internet
- Designed to support a wide variety of anonymity use cases

Low-latency TCP applications

- Tor works by proxying TCP streams
 - (And DNS lookups)
- Focuses on achieving interactive latency
 - WWW, but potentially also chat, SSH, etc.
 - Anonymity tradeoffs compared to remailers

Tor Onion routing

- Stream from sender to D forwarded via A, B, and C
 - One Tor circuit made of four TCP hops
- Encrypt packets (512-byte "cells") as $E_A(B, E_B(C, E_C(D, P)))$
- TLS-like hybrid encryption with "telescoping" path setup

Client perspective

- Install Tor client running in background
- Configure browser to use Tor as proxy
 - Or complete Tor+Proxy+Browser bundle
- Browse web as normal, but a lot slower
 - Also, sometimes `google.com` is in Swedish

Anonymity loves company

- Diverse user pool needed for anonymity to be meaningful
 - Hypothetical Department of Defense Anonymity Network
- Tor aims to be helpful to a broad range of (sympathetic sounding) potential users

Anti-censorship

- As a web proxy, Tor is useful for getting around blocking
- Unless Tor itself is blocked, as it often is
- *Bridges* are special less-public entry points
- Also, protocol obfuscation arms race (currently behind)

Hidden services

- Tor can be used by servers as well as clients
- Identified by cryptographic key, use special rendezvous protocol
- Servers often present easier attack surface

Intersection attacks

- Suppose you use Tor to update a pseudonymous blog, reveal you live in Minneapolis
- Comcast can tell who in the city was sending to Tor at the moment you post an entry
 - Anonymity set of 1000 → reasonable protection
- But if you keep posting, adversary can keep narrowing down the set

Exit sniffing

- Easy mistake to make: log in to an HTTP web site over Tor
- A malicious exit node could now steal your password
- Another reason to always use HTTPS for logins

Browser bundle JS attack

- Tor's Browser Bundle disables many features try to stop tracking
- But, JavaScript defaults to on
 - Usability for non-expert users
 - Fingerprinting via NoScript settings
- Was incompatible with Firefox auto-updating
- Many Tor users de-anonymized in August'13 by JS vulnerability patched in June'13

Next time

- Usability and security