CSci 5271
Introduction to Computer Security
Day 21: Malware and Denial of Service

Stephen McCamant
University of Minnesota, Computer Science & Engineering

Outline

Firewalls and NAT boxes

Intrusion detection systems

Malware and the network

Denial of service and the network

What a firewall is

- Basically, a router that chooses not to forward some traffic
  - Based on an a-priori policy
- More complex architectures have multiple layers
  - *DMZ*: area between outer and inner layers, for outward-facing services

Inbound and outbound control

- Most obvious firewall use: prevent attacks from the outside
- Often also some control of insiders
  - Block malware-infected hosts
  - Employees wasting time on Facebook
  - Selling sensitive info to competitors
  - Nation-state Internet management
- May want to log or rate-limit, not block

Default: deny

- Usual whitelist approach: first, block everything
- Then allow certain traffic
- Basic: filter packets based on headers
- More sophisticated: *proxy* traffic at a higher level

IPv4 address scarcity

- Design limit of 2^{32} hosts
  - Actually less for many reasons
- Addresses becoming gradually more scarce over a many-year scale
- Some high-profile exhaustions in 2011
- IPv6 adoption still very low, occasional signs of progress

Network address translation (NAT)

- Middlebox that rewrites addresses in packets
- Main use: allow inside network to use non-unique IP addresses
  - RFC 1918: 10.*, 192.168.*, etc.
  - While sharing one outside IP address
- Inside hosts not addressable from outside
  - De-facto firewall

Packet filtering rules

- Match based on:
  - Source IP address
  - Source port
  - Destination IP address
  - Destination port
  - Packet flags: TCP vs. UDP, TCP ACK, etc.
- Action, e.g. allow or block
- Obviously limited in specificity

Client and server ports

- TCP servers listen on well-known port numbers
  - Often $<$ 1024, e.g. 22 for SSH or 80 for HTTP
- Clients use a kernel-assigned random high port
- Plain packet filter would need to allow all high-port incoming traffic

Stateful filtering

- In general: firewall rules depend on previously-seen traffic
- Key instance: allow replies to an outbound connection
- See: port 23746 to port 80
- Allow incoming port 23746
  - To same inside host
- Needed to make a NAT practical

Circuit-level proxying

- Firewall forwards TCP connections for inside client
- Standard protocol: SOCKS
  - Supported by most web browsers
  - Wrapper approaches for non-aware apps
- Not much more powerful that packet-level filtering

Application-level proxying

- Knows about higher-level semantics
- Long history for, e.g., email, now HTTP most important
- More knowledge allows better filtering decisions
  - But, more effort to set up
- Newer: "transparent proxy"
  - Pretty much a man-in-the-middle

Tunneling

- Any data can be transmitted on any channel, if both sides agree
- E.g., encapsulate IP packets over SSH connection
  - Compare covert channels, steganography
- Powerful way to subvert firewall
  - Some legitimate uses

Outline

Firewalls and NAT boxes

Intrusion detection systems

Malware and the network

Denial of service and the network

Basic idea: detect attacks

- The worst attacks are the ones you don't even know about
- Best case: stop before damage occurs
  - Marketed as "prevention"
- Still good: prompt response
- Challenge: what is an attack?

Network and host-based IDSes

- Network IDS: watch packets similar to firewall
  - But don't know what's bad until you see it
  - More often implemented offline
- Host-based IDS: look for compromised process or user from within machine

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 $\mathbb{A} \cap \mathbb{M}$, where:
  - $\mathbb{A}$ models allowed sequences
  - $\mathbb{M}$ models sequences achieving attacker's goals
- Further techniques required:
  - Many syscalls made into NOPs
  - Replacement subsequences with similar effect

Outline

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

Outline

Firewalls and NAT boxes

Intrusion detection systems

Malware and the network

Denial of service and the network

DoS versus other vulnerabilities

- Effect: normal operations merely become impossible
- Software example: crash as opposed to code injection
- Less power that 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 `<`) 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

Next time

- Networking layers to protect privacy