Lecture 28
December 6, 2023

Counterattacking

- Use legal procedures
  - Collect chain of evidence so legal authorities can establish attack was real
  - Check with lawyers for this
    - Rules of evidence very specific and detailed
    - If you don’t follow them, expect case to be dropped
- Technical attack
  - Goal is to damage attacker seriously enough to stop current attack and deter future attacks

Consequences

1. May harm innocent party
   - Attacker may have broken into source of attack or may be impersonating innocent party
2. May have side effects
   - If counterattack is flooding, may block legitimate use of network
3. Antithetical to shared use of network
   - Counterattack absorbs network resources and makes threats more immediate
4. May be legally actionable

Example: Counterworm

- Counterworm given signature of real worm
  - Counterworm spreads rapidly, deleting all occurrences of original worm
- Some issues
  - How can counterworm be set up to delete *only* targeted worm?
  - What if infected system is gathering worms for research?
  - How do originators of counterworm know it will not cause problems for any system?
    - And are they legally liable if it does?

Incident Response Groups

- *Computer security incident response team* (CSIRT): team established to assist and coordinate responses to a security incident among a defined constituency
  - "Constituency" defined broadly; may be vendor, company, sector such as financial or academic, nation, etc.
- Mission depends in large part on constituency
  - Critical part: keep constituency informed of services CSIRT provides, how to communicate with CSIRT

Example: CERT/CC

- Grew out of Internet worm, when many groups dealt with it and had to communicate with one another
  - In some cases, they did not know about other groups, what they are doing
  - Sometimes trusted third party did introduction
- Raised concerns of how to communicate and coordinate responses to future events
- Led to development of Computer Emergency Response Team (CERT, later CERT/CC)

CSIRT Missions

1. *Publication*: publish policies, procedures about what it can do, how it will communicate with constituency, how constituency can communicate it
2. *Collaboration*: collaborate with other CSIRTs to gather, disseminate information about attacks, respond to attacks
3. *Secure communication*: preserve credibility; ensure constituency they are communicating with CSIRT and not masquerader; and CSIRT must be sure it is dealing with affected members of constituency and other CSIRTs, not masqueraders

How a CSIRT Functions

- Policy defines what it will, will not do
- Plan how to respond to incidents, driven by needs and constraints of constituents
  - Avoid solely technical approach
  - Couple that with strategic analysis to find organizational issues contributing to attack or hindering appropriate responses
  - Understanding incident involves non-technical aspects of organization such as people, resources, economics, laws and regulations
- Disseminate information to prevent, limit attacks
  - Include vulnerability reports

Digital Forensics

The science of identifying and analyzing entities, states, state transitions of events that have occurred or are occurring

- Also called *computer forensics*
- Usually done to figure out what caused an anomaly or understand nature of attack: how did attackers (try to) enter system, what they did, and how defenses failed
- *Legal forensics* may include digital forensics
    - Here, analysts must acquire information and perform analysis in such a way that what is uncovered can be admitted into a legal proceeding

Goals of Forensics Principles

- *Locard's Exchange Principle*: every contact leaves a trace
- Forensics principles create environment in which Locard's Exchange Principle holds
- Must consider entire system
  - Attack on one component may affect other components
  - Multistage attacks leverage compromise of a component to compromise another
  - Attack may have effects that analyst does not expect

Principle 1: Consider the Entire System

- Analyst needs access to information the intruder had before, after attack
  - Includes changes to memory, kernel, file systems, files
- Rarely recorded continuously, so information incomplete
- Logs also often omit useful information
  - Record connections, states of connections, services, programs executed
  - Omit directories searched to find dynamically loaded libraries, or which ones are loaded; also omit memory contents during program execution
  - Application logging also may not log security-relevant events

Principle 2: Assumptions Should Not Control What Is Logged

- Analysts work from logs capturing information before, during, after incident being analyzed
  - If assumptions guide what is being logged, information may be incomplete
- Record enough information to reconstruct system state at any time
  - Virtual machine introspection great for this

Example: ExecRecorder

Architecture to enable replay of events with minimal overhead and no changes to operating system

- Hypervisor Bochs contains checkpoint, logging, replay mechanisms
  - These are invisible to operating system running in Bochs
- Checkpoint component takes snapshots of system state
- Logging component records nondeterministic events to enable them to be reproduced *exactly*
- Replay component reconstructs and restores state of system, and system activity occurs from that point on

Principle 3: Consider the Effects of Actions As Well As the Actions

- Aim is to establish what system did as well as what attacker did
- Logs record actions, sometimes effects, but almost never causes allowing actions to occur
- Example: remote attacker gains enough access to execute commands on other systems
  - Logs show which server she went to, commands issued
  - Logs do not show vulnerability that enabled attacker to succeed, so others may exploit the same vulnerability

Principle 4: Context Assists in Understanding Meaning

- Same action may cause 2 different effects when executed in 2 different contexts
- Example: LINUX command typed at keyboard (not full path name of command)
  - What gets executed depends on search path, contents of file system
- Example: file system monitoring tool logging access to files by file name
  - The same name may refer to 2 different files (refers to file X, then file X deleted and a new file X created)

Principle 5: Information Must Be Processed, Presented in an Understandable Way

- Those who need to understand the forensic analysis can do so
- First audience: analysts
  - Interfaces to forensic tools must be designed with usability in mind, and indicate where gaps in data, analysis are
  - Presentation of results must also be clear to a technical audience
- Second audience: non-technical audience
  - Provide information in a way that the audience can understand what happened, how it happened, what the effects of the attack were, the level of assurance that the data, analysis is correct
  - May need to present evidence in a way appropriate to a particular audience, such as legal audiences

Practice

Typically 4 steps to reconstruct state of system and sequence of actions of interest

1. Capture, preserve current state of system, network data
2. Extract information about that state and prior states
    - Reverse these steps if system is active; in this case, state will be approximate as gathering data takes time and state may change during that process
3. Analyze data to determine sequence of actions, objects affected, and how they are affected
4. Prepare, report results of analysis to intended audience

Gathering Data

- Get a complete image of all components
- If infeasible (because compromise discovered after it is done, or system is active), get as complete an image as possible
  - May include disk images, backups, stored network or IDS data
- Be sure to make cryptographic hash of all data
  - That way, you and others can verify data is unaltered after being checksummed

Example: Gathering Data

- Disk is full, but space used by files much less than size of disk
- Sysadmin removes disk, mounts it read-only on another system
- Sysadmin creates image of it on some other media
  - On a second, previously wiped, disk
- Sysadmin creates cryptographic checksum of image
  - Can be used to show image was not changed since its creation
- Sysadmin uses a different program to recompute checksum and verifies it matches previously computed checksum
  - Used to ensure cryptographic checksum is correct

Persistent vs. Volatile Data

- Persistent data: remains when system or data storage is powered off
  - Data on hard drive or secondary storage
- Volatile data: transient, disappearing at some point in time (like when system is powered off)
  - Data in memory
  - More difficult to capture than persistent data

Capturing Volatile Data

- Problem: using software to capture memory contents alters memory
- One approach: use specialized hardware
  - Carrier and Grand built custom PCI card; attached to bus
    - When computer boots, card configures itself, disables its controller so it is invisible to programs scanning PCI bus
    - Throw switch, card re-enables controller, suspends CPU, dumps memory to a non-volatile storage medium
    - When done, disables its controller and restart CPU

Capturing Volatile Data

- Second approach: store memory-reading software in trusted location
  - Attacker cannot alter it
  - Software freezes operating system and all associated processes, captures and dumps memory contents, unfreezes operating system and all associated processes
  - Intel IA-32 platforms have System Management Mode to provide such an area
    - SMM has software drivers for standard network PCI card
    - SMM grabs contents of CPU registers, and PCI grabs contents of memory; these transmitted to waiting server
    - Using SMM suspends operating system so memory contents in consistent state

Capturing Volatile Data

- Third approach: put acquisition software between operating system, hardware
  - Virtual machine introspection does this; to capture memory contents, virtual machine monitor stops VM, copies contents of memory
- Fourth approach: remanence effect
  - Memory retains contents for very short time after power lost
  - Cooling memory increases this time significantly
  - This used for forensics on Android phones

Extracting Information

- Analyze to produce a timeline
- Example for the disk mentioned earlier; work done from disk image
  1. Analysts obtain list of files on disk
  2. They check for deleted files; find several corresponding to undeleted files
  3. They examine free space; find large number of files there

Analyze the Data

- Goal is to answer specific questions that depend on nature of attack, resources involved, and the data
- Example for the disk mentioned earlier; information gathered from disk image
- Analysts examine files stored in free space as they are hidden; turn out to be copies of recently released movies
- Key question: how did they get there?
- Analysts extract log files of network server, user actions; find a login name with control characters in it, and no corresponding logout; possible buffer overflow
    - Validation: run login program, give it user name of 1000 characters; it crashes

Analyze the Data

How did attackers gain access to system (to run login program)?

- Analysts examine server logs, server configuration files; nothing suspicious
- Analysts look through other network log files, find an entry made by a program starting the *telnet* service
    - This is a remote terminal interface and should never run
    - Find the program in a sysadmin's directory
- Analysts look at network logs
    - IDS captures packets, stored for 30 days
    - After that, deletes packet bodies and saves headers for 5 months

Analyze the Data

- Analysts look for *telnet* packets; find several, including one containing the user name matching the one with control characters
- Analysts copy these packets to separate file, create a textual representation in another file
    - And these are checksummed and saved on read-only media
- How did movies get put into free space?
    - Obvious answer: attackers simply deleted them or wrote them directly to free space
        - But then disk would not have been full as deleted blocks would simply be overwritten
    - More probable answer: attacker created file, opened it, deleted file from file system
        - Program checking disk space by traversing file hierarchy will miss it; looking at disk map won't; this also explains discrepancy

Report the Findings

Must take into account the audience (principle of presenting information in an understandable way)

- If non-technical audience, report should say movie files stored in unused disk space, and give data on number of movies found, titles, and so forth
- If technical audience, also describe how movies stored, how they were found

This suggests preparing a detailed technical report for reference, then use that as basis for writing other reports as needed

Anti-Forensics

- *Anti-forensics*: the attempt to compromise the availability or usefulness of evidence to forensics process
- Goals:
  - Interfere with forensic analysis tools gathering information, by hiding data or obscuring type, sequence of evidence
  - Hinder the validation of authenticity of digital image
  - Exploit weaknesses in forensic analysis tools
  - Attacking users of forensic analysis tools, for example by crashing analyst’s system or increasing time needed to analyze data
  - Cast doubt on results of forensic analysis; will diminish its credibility in court, for example

Examples

- *timestomp*: enables user to change file access times
- *event_manager*: enables user to delete entries from log files
- JPEG image data compresses digital representation of image into multiple bands of transform coefficients, which generally follow a smooth distribution; altering image perturbs coefficients, so distribution different; anti-forensic tools add dithering to change coefficients back to approximate original one
- Forensic tool determines if Windows files are executable by looking at file extension (“.exe”) and first 2 bytes of file (“MZ”), so anti-forensics tools can just change the extension

Intrusion Detection

- Detect wide variety of intrusions
  - Previously known and unknown attacks
  - Suggests need to learn/adapt to new attacks or changes in behavior
- Detect intrusions in timely fashion
  - May need to be be real-time, especially when system responds to intrusion
    - Problem: analyzing commands may impact response time of system
  - May suffice to report intrusion occurred a few minutes or hours ago

Intrusion Detection Systems

- Present analysis in simple, easy-to-understand format
  - Ideally a binary indicator
  - Usually more complex, allowing analyst to examine suspected attack
  - User interface critical, especially when monitoring many systems
- Be accurate
  - Minimize false positives, false negatives
  - Minimize time spent verifying attacks, looking for them

Principles of Intrusion Detection

- Characteristics of systems not under attack
  - User, process actions conform to statistically predictable pattern
  - User, process actions do not include sequences of actions that subvert the security policy
  - Process actions correspond to a set of specifications describing what the processes are allowed to do
- Systems under attack do not meet at least one of these

Example

- Goal: insert a back door into a system
  - Intruder will modify system configuration file or program
  - Requires privilege; attacker enters system as an unprivileged user and must acquire privilege
    - Nonprivileged user may not normally acquire privilege (violates #1)
    - Attacker may break in using sequence of commands that violate security policy (violates #2)
    - Attacker may cause program to act in ways that violate program's specification

Basic Intrusion Detection

- *Attack tool* is automated script designed to violate a security policy
- Example: *rootkit*
  - Includes password sniffer
  - Designed to hide itself using Trojaned versions of various programs (*ps*, *ls*, *find*, *netstat*, etc.)
  - Adds back doors (*login*, *telnetd*, etc.)
  - Has tools to clean up log entries (*zapper, etc.)*

Detection

- *Rootkit* configuration files cause *ls*, *du*, etc. to hide information
  - *ls* lists all files in a directory
    - Except those hidden by configuration file
  - *dirdump* (local program to list directory entries) lists them too
    - Run both and compare counts
    - If they differ, *ls* is doctored
- Other approaches possible

Key Point

- *Rootkit* does *not* alter kernel or file structures to conceal files, processes, and network connections
  - It alters the programs or system calls that *interpret* those structures
  - Find some entry point for interpretation that *rootkit* did not alter
  - The inconsistency is an anomaly (violates #1)

Denning’s Model

- Hypothesis: exploiting vulnerabilities requires abnormal use of normal commands or instructions
  - Includes deviation from usual actions
  - Includes execution of actions leading to break-ins
  - Includes actions inconsistent with specifications of privileged programs

Models of Intrusion Detection

- Anomaly detection
  - What is usual, is known
  - What is unusual, is bad
- Misuse detection
  - What is bad, is known
  - What is not bad, is good
- Specification-based detection
  - What is good, is known
  - What is not good, is bad

Anomaly Detection

- Analyzes a set of characteristics of system, and compares their values with expected values; report when computed statistics do not match expected statistics
  - Threshold metrics
  - Statistical moments
  - Markov model

Misuse Detection

- Determines whether a sequence of instructions being executed is known to violate the site security policy
  - Descriptions of known or potential exploits grouped into *rule sets*
  - IDS matches data against rule sets; on success, potential attack found
- Cannot detect attacks unknown to developers of rule sets
  - No rules to cover them

Types of Learning

- *Supervised learning methods*: begin with data that has already been classified, split it into “training data”, “test data”; use first to train classifier, second to see how good the classifier is
- *Unsupervised learning methods*: no pre-classified data, so learn by working on real data; implicit assumption that anomalous data is small part of data
- Measures used to evaluate methods based on:
  - TP: true positives (correctly identify anomalous data)
  - TN: true negatives (correctly identify non-anomalous data)
  - FP: false positives (identify non-anomalous data as anomalous)
  - FN: false negatives (identify anomalous data as non-anomalous)

Specification Modeling

- Determines whether execution of sequence of instructions violates specification
- Only need to check programs that alter protection state of system
- System traces, or sequences of events t_1, … t_i, t_{i+1}, …, are basis of this
  - Event t_i occurs at time $C(t_i)$
  - Events in a system trace are totally ordered

Comparison and Contrast

- Misuse detection: if all policy rules known, easy to construct rulesets to detect violations
  - Usual case is that much of policy is unspecified, so rulesets describe attacks, and are not complete
- Anomaly detection: detects unusual events, but these are not necessarily security problems
- Specification-based vs. misuse: spec assumes if specifications followed, policy not violated; misuse assumes if policy as embodied in rulesets followed, policy not violated

Measuring Effectiveness

- *Accuracy*: percentage (or fraction) of events classified correctly
  - ((*TP* + *TN*) / (*TP* + *TN* + *FP* + *FN*)) * 100%
- *Detection rate*: percentage (or fraction) of reported attack events that are real attack events
  - (*TP* / (*TP* + *FN*)) * 100%
  - Also called the *true positive rate*
- *False alarm rate*: percentage (or fraction) of non-attack events reported as attack events
  - (*FP* / (*FP* + *TN*)) * 100%
  - Also called the *false positive rate*

Usefulness of Measurement

- Data at installation should be similar to that used to measure effectiveness
- Example: military, academic network traffic different
  - KDD-CUP-99 dataset derived from unclassified and classified network traffic on an Air Force Base
  - Network data captured at Florida Institute of Technology
- FIT data showed anomalies not in KDD-CUP-99
  - FIT data: TCP ACK field nonzero when ACK flag not set
  - KDD-CUP-99 data: HTTP requests all regular, all used GET, version 1.0; in FIT data, HTTP requests showed inconsistencies, some commands not GET, versions 1.0, 1.1
- Conclusion: using KDD-CUP-99 data would show some techniques performing better than they would on the FIT data