Announcements

- Two articles you can use for "good faith" paper (which is due next Tuesday)
  - no extensions!

- Lab 5 is posted
  - reading due by Tuesday
  - check point on Wed. the 20th
  - final version due Wed. the 27th

- Check finals schedule & notify me of any conflicts
Computer Forensics

The branch of forensics that specializes in recovering digital evidence from a sequence of unknown events usually for eventual use in a court of law.

We've all seen CSI, but as more crimes “go digital”, different expertise is needed.

Fingerprints are hard to fake, but what about digital evidence?
Digital DNA

- Username (typically in log files)
- Network Address
- CPU Serial Numbers - on Pentium III's until ~2000
- Hardware/Software artifacts
- Software watermarks
  Ex: GUID in MS Office (used to track Melissa Worm)
- Encryption keys
Tools

1. Disk imaging + hashing
2. Text or binary editors
   Ex: Unix strings
3. System Logs ← more today
4. Network Scanners
5. Software Scanners
6. Data recovery
Issues in Forensics

- Chain of Custody
  First thing is to image + then never touch original
- Cryptographic hashes
  Periodically checked for tampering
A word of caution

Computer forensics is delicate; without legal authorization, illegal!

(So stay on DETER or your own private machine!)

Even in legal investigations, care must be taken not to exceed the warrant.
Main Element: Logging

2 issues:

1. How to configure ahead of time

2. How to use effectively after an issue has occurred.
Auditing:

1. Event discriminator
2. Alarm processor
3. Audit recorder
4. Audit analyzer
5. Security audit trail
6. Audit trail examiner
7. Audit provider
8. Audit archiver
9. Security reports
10. Archives
11. Action
Auditing: What to collect

Issues:
- Amount of data: quantity versus efficiency/space

Possibilities:
- Events relating to audit software
- Events relating to security mechanisms
- Intrusion detection & firewall events
- System management events
- System calls to OS
- Remote access or logins
- Access to some applications
Separation of audits:
① System level
② Application level
③ User level
System Level

- Includes login attempts, devices used, OS functions performed
- Also useful for monitoring system performance

Ex:

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Host</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jan 27</td>
<td>17:14:04</td>
<td>host1</td>
<td>login: ROOT LOGIN console</td>
</tr>
<tr>
<td>Jan 27</td>
<td>17:15:04</td>
<td>host1</td>
<td>shutdown: reboot by root</td>
</tr>
<tr>
<td>Jan 27</td>
<td>17:18:38</td>
<td>host1</td>
<td>login: ROOT LOGIN console</td>
</tr>
<tr>
<td>Jan 27</td>
<td>17:19:37</td>
<td>host1</td>
<td>reboot: rebooted by root</td>
</tr>
<tr>
<td>Jan 28</td>
<td>09:46:53</td>
<td>host1</td>
<td>su: 'su root' succeeded for user1 on /dev/tty0</td>
</tr>
<tr>
<td>Jan 28</td>
<td>09:47:35</td>
<td>host1</td>
<td>shutdown: reboot by user1</td>
</tr>
<tr>
<td>Jan 28</td>
<td>09:53:24</td>
<td>host1</td>
<td>su: 'su root' succeeded for user1 on /dev/tty1</td>
</tr>
<tr>
<td>Feb 12</td>
<td>08:53:22</td>
<td>host1</td>
<td>su: 'su root' succeeded for user1 on /dev/tty1</td>
</tr>
<tr>
<td>Feb 17</td>
<td>08:57:50</td>
<td>host1</td>
<td>date: set by user1</td>
</tr>
<tr>
<td>Feb 17</td>
<td>13:22:52</td>
<td>host1</td>
<td>su: 'su root' succeeded for user1 on /dev/tty0</td>
</tr>
</tbody>
</table>

2 Application - Level

- detect security violations within an application or flaws in application

Ex: mail delivery system:

<table>
<thead>
<tr>
<th>Date</th>
<th>Time</th>
<th>Host</th>
<th>Time</th>
<th>Size</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apr 9</td>
<td>11:20:22</td>
<td>host1</td>
<td>AA06370</td>
<td>from=<a href="mailto:user1@host1">user1@host1</a>, size=3355, class=0</td>
<td></td>
</tr>
<tr>
<td>Apr 9</td>
<td>11:20:23</td>
<td>host1</td>
<td>AA06370</td>
<td>to=<a href="mailto:user2@host2">user2@host2</a>, delay=00:00:02, stat=Sent</td>
<td></td>
</tr>
<tr>
<td>Apr 9</td>
<td>11:59:51</td>
<td>host1</td>
<td>AA06436</td>
<td>from=<a href="mailto:user4@host3">user4@host3</a>, size=1424, class=0</td>
<td></td>
</tr>
<tr>
<td>Apr 9</td>
<td>11:59:52</td>
<td>host1</td>
<td>AA06436</td>
<td>to=<a href="mailto:user1@host1">user1@host1</a>, delay=00:00:02, stat=Sent</td>
<td></td>
</tr>
<tr>
<td>Apr 9</td>
<td>12:43:52</td>
<td>host1</td>
<td>AA06441</td>
<td>from=<a href="mailto:user2@host2">user2@host2</a>, size=2077, class=0</td>
<td></td>
</tr>
<tr>
<td>Apr 9</td>
<td>12:43:53</td>
<td>host1</td>
<td>AA06441</td>
<td>to=<a href="mailto:user1@host1">user1@host1</a>, delay=00:00:01, stat=Sent</td>
<td></td>
</tr>
</tbody>
</table>

vary greatly depending on app
User-level

- holds users accountable
- can define "normal" behavior over time

Ex: Commands executed by users (on UNIX system)

<table>
<thead>
<tr>
<th>Command</th>
<th>User</th>
<th>Terminal</th>
<th>Time</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>rcp</td>
<td>user1</td>
<td>ttty0</td>
<td>0.02 secs</td>
<td>Fri Apr 8 16:02</td>
</tr>
<tr>
<td>ls</td>
<td>user1</td>
<td>ttty0</td>
<td>0.14 secs</td>
<td>Fri Apr 8 16:01</td>
</tr>
<tr>
<td>clear</td>
<td>user1</td>
<td>ttty0</td>
<td>0.05 secs</td>
<td>Fri Apr 8 16:01</td>
</tr>
<tr>
<td>rpcinfo</td>
<td>user1</td>
<td>ttty0</td>
<td>0.20 secs</td>
<td>Fri Apr 8 16:01</td>
</tr>
<tr>
<td>nroff</td>
<td>user2</td>
<td>ttty2</td>
<td>0.75 secs</td>
<td>Fri Apr 8 16:00</td>
</tr>
<tr>
<td>sh</td>
<td>user2</td>
<td>ttty2</td>
<td>0.02 secs</td>
<td>Fri Apr 8 16:00</td>
</tr>
<tr>
<td>mv</td>
<td>user2</td>
<td>ttty2</td>
<td>0.02 secs</td>
<td>Fri Apr 8 16:00</td>
</tr>
<tr>
<td>sh</td>
<td>user2</td>
<td>ttty2</td>
<td>0.03 secs</td>
<td>Fri Apr 8 16:00</td>
</tr>
<tr>
<td>col</td>
<td>user2</td>
<td>ttty2</td>
<td>0.09 secs</td>
<td>Fri Apr 8 16:00</td>
</tr>
<tr>
<td>man</td>
<td>user2</td>
<td>ttty2</td>
<td>0.14 secs</td>
<td>Fri Apr 8 15:57</td>
</tr>
</tbody>
</table>
Physical Access

Any critical system will be kept in a secured location.

Why?
If you have physical access, you can break in!

So door access, modification of access privileges, etc., is also relevant log information.
Protecting log data

1. Read/write on a host
   - log files
   - separate server, encryption

2. Write-once, read-many device
   - CD-ROM
   - magnetic tapes

3. Write only device
   - printer
Logging in Windows

Window Event Log:

Each event gets a numeric ID code, set of attributes (such as task, opcode, version, keywords), plus optional user supplied data.

3 types of logs:

- System event log
- Application event log
- Security event log
Windows (cont)

Auditing can be enabled in 9 categories:

- Account logon events
- Account management
- Directory service access
- Logon events (local)
- Object access
- Policy changes
- Privilege use
- Process tracking
- System events
Windows example

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Success Audit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event Source</td>
<td>Security</td>
</tr>
<tr>
<td>Event Category</td>
<td>(1)</td>
</tr>
<tr>
<td>Event ID</td>
<td>517</td>
</tr>
<tr>
<td>Date</td>
<td>3/6/2006</td>
</tr>
<tr>
<td>Time</td>
<td>2:56:40 PM</td>
</tr>
<tr>
<td>User</td>
<td>NT AUTHORITY\SYSTEM</td>
</tr>
<tr>
<td>Computer</td>
<td>KENT</td>
</tr>
<tr>
<td>Description</td>
<td>The audit log was cleared</td>
</tr>
<tr>
<td>Primary User Name</td>
<td>SYSTEM</td>
</tr>
<tr>
<td>Primary Domain</td>
<td>NT AUTHORITY</td>
</tr>
<tr>
<td>Primary Logon ID</td>
<td>(0x0,0x3F7)</td>
</tr>
<tr>
<td>Client User Name</td>
<td>userk</td>
</tr>
<tr>
<td>Client Domain</td>
<td>KENT</td>
</tr>
<tr>
<td>Client Logon ID</td>
<td>(0x0,0x28BFD)</td>
</tr>
</tbody>
</table>
UNIX logging

Syslog is the default found on all UNIX systems

Elements:
- syslog(): API referenced by several standard utilities, available to applications
  - logger -command to add entries to system log
  - /etc/syslog.conf
  - syslogd

Not uniform across UNIX systems!
Basic services:
- capture relevant events
- store them
- transmit to central machine, a syslog server

Other functions:
- robust filtering: basic is only facility
  + priority, but adds host or program / source or other filters
- log analysis: originally, no analysis
- event response
- log file encryption
- database storage
- rate limiting (to resist DDOS)
Unix example

Mar 1 06:25:43 server1 sshd[23170]: Accepted publickey for server2 from 172.30.128.115 port 21011 ssh2

Mar 1 07:16:42 server1 sshd[9326]: Accepted password for murugiah from 10.20.30.108 port 1070 ssh2

Mar 1 07:16:53 server1 sshd[22938]: reverse mapping checking getaddrinfo for ip10.165.nist.gov failed - POSSIBLE BREAKIN ATTEMPT!

Mar 1 07:26:28 server1 sshd[22572]: Accepted publickey for server2 from 172.30.128.115 port 30606 ssh2

Mar 1 07:28:33 server1 su: BAD SU kkent to root on /dev/ttyp2

Mar 1 07:28:41 server1 su: kkent to root on /dev/ttyp2