ISA 673
Operating Systems’ Security
HoneyPots & HoneyClients

Angelos Stavrou, George Mason University
Honeypots

- Honeypots are real or emulated vulnerable systems ready to be attacked.

- Definition: 
  
  "Honeypot is an information system resource whose value lies in unauthorized or illicit use of that resource"

- Primary value of honeypots is to collect information.
- This information is used to better identify, understand and protect against threats.
- Honeypots add little direct value to protecting your network.
Why use HoneyPots

- A great deal of the security profession and the IT world depend on honeypots.
- Honeypots are used to
  - Build anti-virus signatures
  - Build SPAM signatures and filters
  - Identify compromised systems
  - Assist law-enforcement to track criminals
  - Hunt and shutdown botnets
  - Malware collection and analysis
Advantages and Disadvantages

- **Advantages**
  - Collect only small data sets (only when interacted), which is valuable and easier to analyze.
  - Reduce false positives – because any activity with the honeypot is *unauthorized* by definition.
  - Reduce false negatives – honeypots are designed to identify and capture new attacks.
  - Capture encrypted activity – because honeypots act as endpoints, where the activity is decrypted.
  - Work with IPv6.
  - Highly flexible – extremely adaptable and can be used in a variety of environments.
  - Require minimal resources.
Disadvantages

- Honeypots have a limited field of view – see only what interacts with them. Can’t be used to detect attacks on other systems.
- However, there are some techniques to redirect attackers’ activities to honeypots.
- Risk – attacker may take over the honeypot and use it to attack other systems.
Types of Honeypots

- Server: Put the honeypot on the Internet and let the bad guys come to you.
- Client: Honeypot initiates and interacts with servers
- Other: Proxies
Types of Honeypots

- Low-interaction
  - Emulates services, applications, and OS’s
  - Low risk and easy to deploy/maintain
  - But capture limited information – attackers’ activities are contained to what the emulated systems allow

- High-interaction
  - Real services, applications, and OS’s
  - Capture extensive information, but high risk and time intensive to maintain
  - Can capture new, unknown, or unexpected behavior
Examples of Honeypots

- BackOfficer Friendly
- KFSensor
- Honeyd
- Honeynets

Low Interaction

High Interaction
Uses of Honeypots

- Preventing attacks
  - Automated attacks – (e.g. worms)
    - Attacker randomly scan entire network and find vulnerable systems
    - “Sticky honeypots” monitor unused IP spaces, and slows down the attacker when probed
    - Use a variety of TCP tricks, such as using 0 window size
  - Human attacks
    - Use deception/deterrence
    - Confuse the attackers, making them waste their time and resources
    - If the attacker knows your network has honeypot, he may not attack the network
Uses of Honeypots

Detecting attacks

- Traditional IDSs generate too much logs, large percentage of false positives and false negatives
- Honeypots generate small data, reduce both false positives and false negatives
- Traditional IDSs fail to detect new kind of attacks, honeypots can detect new attacks
- Traditional IDSs may be ineffective in IPv6 or encrypted environment
Uses of Honeypots

- Responding to attacks
  - Responding to a failure/attack requires in-depth information about the attacker.
  - If a production system is hacked (e.g. mail server) it can’t be brought offline to analyze.
  - Besides, there may be too much data to analyze, which will be difficult and time-consuming.
  - Honeypots can be easily brought offline for analysis.
  - Besides, the only information captured by the honeypot is related to the attack – so easy to analyze.
Uses of Honeypots

- **Research purposes**
  - How can you defend yourself against an enemy when you don’t know who your enemy is?
  - Research honeypots collect information on threats.
  - Then researchers can
    - Analyze trends
    - Identify new tools or methods
    - Identify attackers and their communities
    - Ensure early warning and prediction
    - Understand attackers’ motivations
Honeynets

- High-interaction honeypot designed to capture in-depth information.
- Information has different value to different organizations.
- It's an architecture you populate with live systems, not a product or software.
- Any traffic entering or leaving is a suspect.
Honeynet Architecture
How It Works

- A highly controlled network
  - where every packet entering or leaving is monitored, captured, and analyzed.

- Should satisfy two critical requirements:
  - *Data Control*: defines how activity is contained within the honeynet, without an attacker knowing it
  - *Data Capture*: logging all of the attacker’s activity without the attacker knowing it

- Data control has priority over data capture
Data Control

- Mitigate risk of honeynet
  - being used to harm non-honeynet systems
- Tradeoff
  - need to provide freedom to attacker to learn about him
  - More freedom – greater risk that the system will be compromised
- Some controlling mechanisms
  - Restrict outbound connections (e.g. limit to 1)
  - IDS (Snort-Inline)
  - Bandwidth Throttling
No Data Control
Data Control: Issues

- Must have both automated and manual control
- System failure should leave the system in a closed state (fail-close)
- Admin should be able to maintain state of all inbound and outbound connections
- Must be configurable by the admin at any time
- Activity must be controlled so that attackers can’t detect
- Automated alerting when honeypots compromised
Data Capture

- Capture all activity at a variety of levels.
  - Network activity.
  - Application activity.
  - System activity.

- Issues
  - No captured data should be stored locally on the honeypot
  - No data pollution should contaminate
  - Admin should be able to remotely view honeynet activity in real time
  - Must use GMT time zone
Risks

- **Harm**
  - Compromised honeynet can be used to attack other honeynets or non-honeynet systems

- **Detection**
  - Its value will dramatically decreased if detected by hacker
  - Hacker may ignore or bypass it
  - Hacker may inject false information to mislead

- **Disabling honeynet functionality**
  - Attacker disables the data control & capture

- **Violation**
  - Using the compromised system for criminal activity
Types of honeynets

- Gen-I
- Gen-II
- Virtual
- Distributed
Gen-II Honeynet Architecture
Virtual Honeynet

Diagram of a single, physical computer with multiple operating systems. The Host operating system is first installed, then the virtualization software is installed upon it. Last, the Guest OS's are installed, controlled by the virtualization software. The Host operating system acts as the Data Control and Data Capture of the Honeynet. It is the gateway to each GuestOS (honeyos). The Honeynet Project demonstrated this at Blackhat 2002, running a virtual Honeynet on a laptop.
Hybrid Virtual Honeynet

Diagram of two physical computers. The first computer, the Honeynet gateway, acts as the Data Control and Data Capture mechanism for the Honeynet. On a second physical computer we install multiple operating systems, GuestOS's, each one of which is a separate honeypot. There is also a HostOS that is not shown. The HostOS is not a honeypot, it's just a platform to install and manage the GuestOS's within the virtual software.

Source: http://his.sourceforge.net/honeynet/papers/virtual/virt2.jpg
What are Honeyclients?

High-interaction (active) 
client-side 
honeypots

used for detecting and characterizing malicious sites by driving a system in a way that mimics human users
<table>
<thead>
<tr>
<th>Low</th>
<th>Transport layer virtualization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Medium</td>
<td>Application layer virtualization</td>
</tr>
<tr>
<td>High</td>
<td>Real, vulnerable systems</td>
</tr>
</tbody>
</table>
Trade-offs

- Speed
- Ease of
  - Implementation
  - Maintenance
  - Reuse
  - Detection (fingerprinting)
- Depth of information gathered
- Reality vs. simulation
- Resources required
Server-side vs. Client-side
Why client-side is so important

- Threats triggered by end-user behavior
- Security is fundamentally a human problem
- Criminal focus on soft-targets
Honeyclient Uses

- Evaluating/characterizing web sites
- Testing endpoint security
- Detecting zero-day browser exploits
- Mapping malicious neighborhoods
- Obtaining unique malware and exploit samples
Examples

- Drive-by downloads
- Adware/spyware
- Exploitation websites
- Phishing?
- Typo-squatting
- Zero-day exploits against browsers
Honeyclients vs. Crawlers

**Honeyclients**
- Vulnerable to attack
- Utilize a mechanized browser when surfing
- Must be monitored to detect compromise
  - Blackbox (MS Strider)
  - Integrity checks
  - Scans (AV, AS, etc)
  - Intrusion Detection
  - Sandbox (Sandboxie)

**Crawlers**
- Not supposed to be compromised
- Crawlers programmatically surf websites to retrieve content
- Simulation can be used to determine if content is malicious (sandbox)
Issues with Crawlers

- Simulation
  - Exploit may not trigger
  - Active/dynamic content
  - Chain reactions
  - Secondary vulnerabilities

- Ease of detection
  - Fingerprinting

- Maliciousness detection
  - Signatures
  - Interpretation
Issues with Honeyclients

- Speed
  - More complexity = slower

- Stability
  - Infected systems are slow

- Maintenance
  - Reset after infection

- Maliciousness detection
  - Sandbox, IDS, scanners
Projects Utilizing Honeyclient or Crawler Technology

- MS Strider HoneyMonkey (Microsoft Research)
- Honeyclient.org (Kathy Wang)
- Mitre Honeyclient Project (Mitre)
- Client-side Honeypots (Univ. of Mannheim)
- Collapsar/Reverse Honeyfarm (Purdue Univ.)
- Phileas (Webroot)
- Websense (Hubbard)
- SiteAdvisor (McAfee)
Projects Utilizing Honeyclient or Crawler Technology Cont’d

- StillSecure / Pezzonavante (Danford)
- SPECTRE (Sunbelt)
- Shadow Honeypots (Anagnostakis)
- Email quarantine systems (Columbia Univ.)
- Spycrawler (Univ. of Washington)
- XPLIntel (Exploit Prevention Labs)
- Irish Honeynet Project (Espion)
When we browsed this site, it made unauthorized changes to our test PC.

Are you the owner of this site? Leave a comment

AUTOMATED WEB SAFETY TESTING RESULTS FOR HIGHCONVERT.COM

BROWSER EXPLOIT: Breaches browser security
When we browsed this site, it made unauthorized changes to our test PC

E-MAIL TESTS FOR HIGHCONVERT.COM: We have not found any e-mail sign-up forms on this site.

DOWNLOAD TESTS FOR HIGHCONVERT.COM: Testing 3 downloads
We found 3 downloads here, which we are still testing.
Submit a download for analysis

ONLINE AFFILIATIONS FOR HIGHCONVERT.COM: Links to red sites
When we tested this site we found links to super-reality-porn-sites.com, which we found to be a distributor of downloads. Some people consider adware, spyware or other unwanted programs.
Honeyclient.org Honeyclient
Issues to Overcome

- Constant supply of URLs
- Preventing infected clients from infecting the planet (honeywall)
- Surf tracking (URL Server)
  - Results from each visit
  - Coordinate across clients
  - Limited retries
- Correlating infections
- Avoid being blacklisted
Pezzonavante Honeyclient
Deployment Experience

October 2005 – March 2006

- 200,000 URLs surfed
- 7 million links harvested
- 600+ virus infections
- 750+ spyware-related events
- 1,500 malware samples
- 500+ malicious URLs submitted for takedown
Issues Found

- Speed
- Coordination
- Correlation
- Information Overload
- Candidate URLs
- Anti-VMware techniques

*Infected PCs are slow and unstable. Duh!*
Characterizing URLs

- Potentially malicious websites need to be identified in advance (guided search)
- Avoid surfing .mil, .gov, and froogle all day
Methods for Determining Candidate URLs

1. Compare IP/hostname against blacklists
2. Filename ends in an executable suffix (*.scr, *.exe, *.pif)
3. Known-bad strings (ie0502.htm, cartao, cmd.txt)
4. Obfuscated URLs
5. Known redirectors (from previous squid logs)
6. McAfee SiteAdvisor ranking
7. Site logged in the Norman Sandbox
8. Site or URL substring shows up in virus descriptions
URL Sources

- URLs harvested from unsolicited email
- Google API
- Harvested links
- SANS ISC URL list
- Blacklists
Detecting Malicious Activity

Pezzonavante used a hybrid, asynchronous approach to detection

- Osiris integrity checking
- Security tool scans
- Snort network IDS alerts
- Traffic analysis
- Snapshot comparisons
Sandboxes and Integrity Checking

- CWSandbox –
- Sandboxie –
  http://www.sandboxie.com/
Integrity Checking

Osiris

compare time: Tue Oct 25 22:57:40 2005
host: victim5
scan config: master (9ad9b7e7)
log file: 173
base database: 19
compare database: 20

[203][victim5][new][c:\windows\dlgb.exe]
[203][victim5][new][c:\windows\extract.exe]
[203][victim5][new][c:\windows\id.exe]
[203][victim5][new][c:\windows\ieupdate.dat]
[203][victim5][new][c:\windows\system32\appwiz.dll]
[203][victim5][new][c:\windows\ts.exe]
[203][victim5][new][c:\windows\v010101.exe]
[203][victim5][new][c:\windows\wupdt.exe]
Running an anti-virus product after the fact will produce some results

<table>
<thead>
<tr>
<th>Date</th>
<th>Filename</th>
<th>Virus Name</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>11/17/2005 17:04</td>
<td>yes[1].htm</td>
<td>Trojan Horse</td>
<td>Infected</td>
</tr>
</tbody>
</table>

But...
Intrusion Detection (Snort)

NIDS was most helpful in monitoring for post-infection behavior.

However, occasional gems were found…..

[**] [1:2436:5]  <eth5> WEB-CLIENT Microsoft wmf metafile access [**]
192.168.1.205:51372 -> 85.255.115.196:80 TCP TTL:64 TOS:0x0
ID:49261 IpLen:20 DgmLen:609 DF
***AP*** Seq: 0xBC5B3A5B Ack: 0xDB156A2C Win: 0x5B4 TcpLen: 32
[Xref => http://www.securityfocus.com/bid/10120]
Squid is Your Friend

Log entry for site access referenced in previous slide

113402032.379 376 192.168.1.205 TCP_MISS/200 1315
GET http://196.regvista.com/ie0601e.wmf
DIRECT/85.255.115.196 video/unknown
[HTTP/1.1 200 OK
Date: Thu, 02 Feb 2006 11:21:12 GMT
Server: Apache/2.0.53 (Fedora)
Accept-Ranges: bytes
Content-Length: 1024
Connection: close
Content-Type: video/unknown]
Network Traffic Analysis (IPTables)

- Basic visualization needs similar to other honeynet projects
- New visualization tools needed to observe near real-time activity on the client

Oct 20 19:36:41 localhost kernel: OUTBOUND TCP: IN=br0 OUT=br0
SRC=192.168.1.205 DST=1.1.1.1 LEN=48 TOS=0x00 PREC=0x00 TTL=128
PROTO=TCP SPT=1229 DPT=6667 WINDOW=64240 RES=0x00 SYN
Anti-honeyclient Methods

1. Blacklisting
   - Try to “look” normal and not get blacklisted
   - Distributed honeyclient farms

2. Dialog boxes
   - GUI automation needed (ex. Windpysend)

3. Anti-crawler techniques

4. Time-bombs
   - Wait 10 sec in case of delayed exploit

5. Page-close events
   - Load a blank page to trigger event (delayed exploit)
6. Non-deterministic URL behavior
   - Pool stats with other farms. Overlap surfing
7. Links no human would click
   - Background color hyperlinks
   - IMG links with “don’t click” on them
8. Timing analysis
9. Surf behavior
   - Timing analysis
   - Paths through a site
     - Depth-first vs. breadth-first
     - Referer information (deep linking)
10. Dynamic and relative URLs
   - JavaScript $*&#*
12. Cookies
13. Session IDs
14. Encoded URLs, foreign character sets
15. URL redirection
Malware Analysis Evasion

- Current trend in certain malware code-bases for detecting debugger or virtual machine environments
- More study required to determine what percentage of infections virtual honeyclients may miss
- Physical machines plus a disk imager like Ghost may be needed
Anti-VMware and VMware Detection Methods

1. Nopill (Smith)
2. Vmdetect (Lallous)
3. Redpill (Rutkowska)
4. Scoopy Doo (Klein)
5. Jerry (Klein)
6. Vmtools (Kato)
Malware Analysis Frameworks

- Analysis requires automation
- Sandboxes and fully instrumented lab networks
- Tools for building your own
The Future

- Data aggregation
- Data sharing
- Distributed Honeyclient Farms
- Correlate honeyclient and honeynet data
- Analysis (SANS ISC, CastleCops PIRT)
- Coordinated take-downs