

Implementing the Honeypot

Anton Chuvakin, Ph.D, GCIA

November 2002

Implementing the Honeypot

- What do you want? (10%)
 - High to low level view of honeypots
- Honeypots overview (20%)
- Detailed need and available technology assessment (20%)
- Building the honeypot (50%)
 - Unix/Linux research honeypot

Miscellaneous Comments

- **Audience:** security, system network administrators, technical management
- **Prerequisites:** TCP/IP, network and host security, UNIX

“Galaxy” View

- What do you want?
- Goals of your security program
 - Existing company policies
 - Available resources
 - Threat landscape
- What is a honeypot?
- What honeypots are good for?
- What can you get away with as a result?

“Solar System” View

- Three main reason to have a honeypot
- **Catch vs study vs defend**
- Catch
 - Entrapment, dealing with LE, etc
- Study
 - Why do you need it? Do you, really? ROI? Benefits?
- Defend
 - Maybe look elsewhere? Good IDS, SIM, etc

Sky-high View

- Research vs Production Honeypots
 - Honeynets – of course
- Low Interaction vs High Interaction
 - Maybe medium-interaction is for you
- Passive and active honeypot
 - Honeypots attack!
- Dangers and illusions
 - Your skills vs attackers – battleground “high interaction honeypot”

Mid-level View: Honeypot Policy I

- Honeypot security policy
 - Network placement
 - Production vs dedicated link
 - DMZ vs internal
 - Hardened box vs softened box
 - As tight as you can make it – not vulnerable to the best of your knowledge
 - Soft – potentially vulnerable
 - Known vulnerable
 - Known commonly exploitable

Mid-level View: Honeypot Policy II

- Services to offer
 - Network daemons and local applications
 - Web services? Databases? Proxies?
 - Simulated hacked environment?
- User activity
 - LAN inbound/outbound connections
 - Web surfing
 - Email / mailing lists
 - IRC
- User stored data

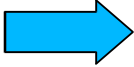
Mid-level View: Honeypot Policy III

- Inbound connections
 - All remote allowed
 - Limited by service / source location
 - Wireless
- Outbound connections
 - Unlimited – a **bad** idea!
 - Limited by number, location, protocol, service
 - Attack dropping
 - Packet mangling
 - None

Low-Mid Level: Fitting In

- Everything else in security should be cool!
- Honeypot placement
 - Dedicated connection
 - External net
 - Screened subnet (DMZ)
 - Dedicated subnet
 - Internal LAN
- No conflicts with current security policy?

Low Level: Platform and Topology

- Data Capture and Data Control – overview 
- Gen I vs Gen II honeynets – overview
- Typical Gen I: firewall, IDS, bash trojan
- Typical Gen II: bridge (“stealth” firewall), inline IDS, kernel trojan
- OS platform choices

Low Level: Software Controls

Data Capture and Data Control

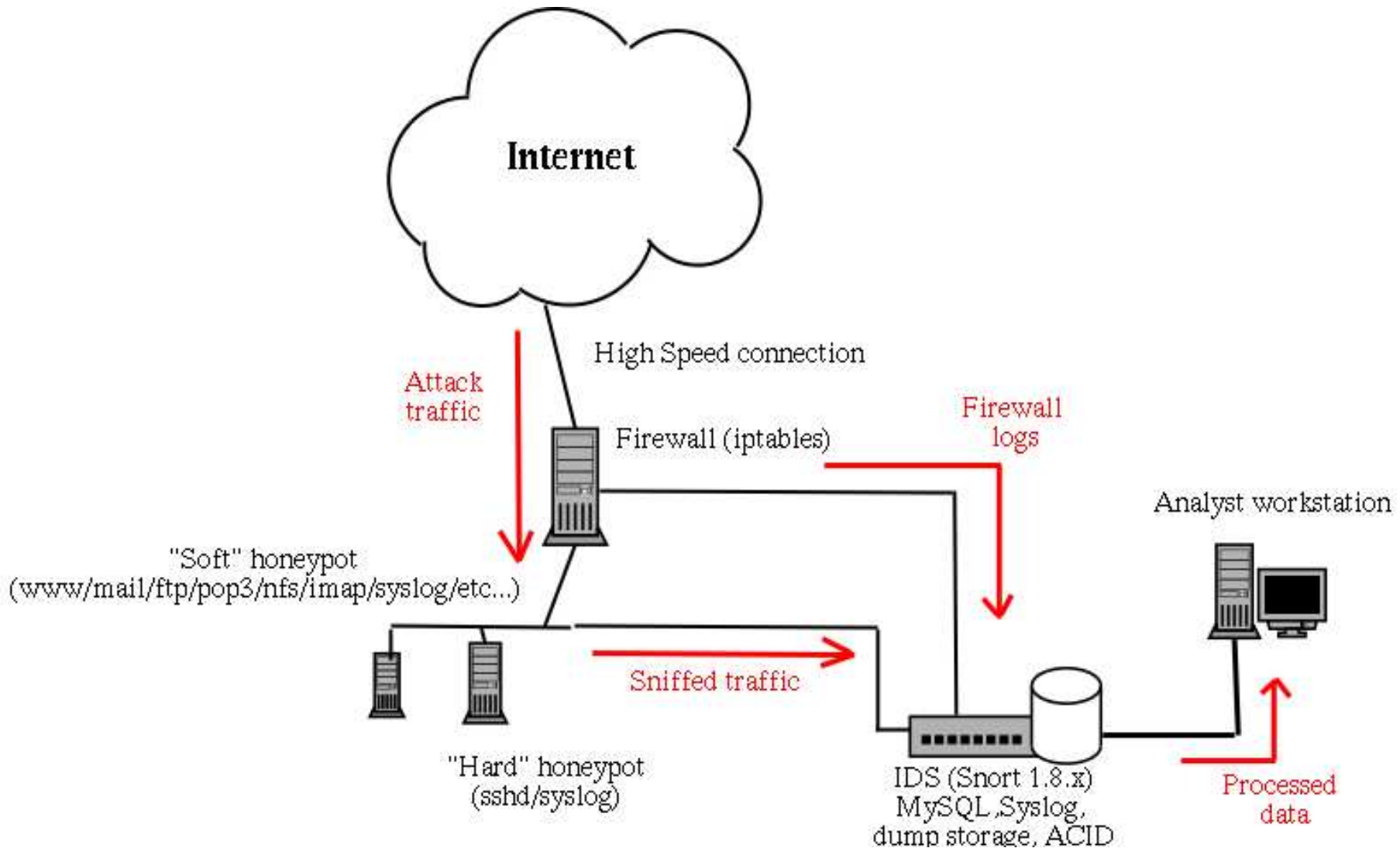
- Data Control – prevent attack escape:
 - Firewall / router
 - Bandwidth Throttle
- Data Capture – collect evidence
 - IDS
 - Tcpdump
- Data Collection – centralized reporting

Low Level: Other Resources

- Advanced security skills needed
- Time requirements
 - Honeypots **HAVE TO** be maintained!
 - Data analysis is time consuming
 - Incident response
- Legal support

Case Study: UNIX Research Honeypot

- Honeypot platform: UNIX
 - Transparency
 - Flexibility
 - “Securability”
 - Familiarity
- Victim platform: UNIX
 - Commonly deployed on servers – often attacked
- Options: Gen I and Gen II setups
- Options: normal and paranoid setup



Setup: Hardware (GI and GII)

- Four Intel-based machines - firewall/bridge, IDS, victim
 - Performance machine: IDS
 - Old junk: firewall
 - Whatever is left: victim
- Have just one or two machine? Virtual honeynets!
 - Risky business
- Linux, Solaris, Free/OpenBSD – take your pick

Wipe the pots!

- Prepare the hardware - “sterilize”
 - Helps lots with forensics
- Build the network
 - Defence in-depth, strict firewall rules
- Connect all to the management LAN
 - But not to outside
- Install and harden the chosen OS for firewall, IDS and victim

Software I : Firewall

- Linux iptables script available for GI and GII setups
 - <http://project.honeynet.org/papers/honeynet/>
- Free/OpenBSD ipf/pf developed for GI
- CheckPoint FW1 (GI)
- Control rules:
 - Counting outbound connections per protocol (GI, II)
 - Passing packets to snort-inline (GII)
 - Block spoofing (GI, II)
 - Block connections to the firewall (GI)
- Rate limiting (Linux *tc* and BSD patched *pf*)

Software II: IDS

- Snort (GI, II)
 - Ruleset developed: log all to binary dumps and SQL database, alert to syslog
 - *log ip any any <> any any (msg: "Snort Unmatched"; session: printable;)*
- Snort-inline (GII)
 - More Data Control than Capture
- TCPdump (GI,II)
 - Just a *tcpdump -i eth1 -s 1600 -w tcpdump_Oct12.dump*
 - Serves as backup data capture

Software III: Other

- NTP sync between firewall/bridge and IDS
 - UDP 123 with stringent access controls
- Remote logging from firewall to IDS
- **netForensics** agent on the IDS
- Alerting scripts available (*swatch*)
 - <http://project.honeynet.org/papers/honeynet/>
- *Logsentry* is good for daily reporting
- ACID/ Demarc for visual SQL database analysis
- **netForensics** for in-depth analysis and correlation

Host Hardening: Soft/Regular

- Minimized UNIX with few network services
- SSH RSA access only (no passwords)
 - With stringent access controls and TCP wrappers
- HTTP SSL for console
 - Access controls
- Host firewalls (only specific management hosts)
- AIDE/Tripwire
- Covert remote shutdown (email)

Host Hardening: Paranoid I

- Do you believe in any of this?
 - Stealth sniffer attacks (via libpcap, tcpdump bugs)
 - Bugs were discovered
 - Remote *syslog* bugs
 - iptables/pf/ipf/CheckPoint bugs
 - Are those real? Can firewall protect you?
 - TCP/IP stack bugs
 - Kernel remote exploits. Vile rumors or ...?
 - Network driver bugs and level II stuff
- Well, if you do...

Host Hardening: Paranoid II

- Chroot non-root syslog
 - Remote exploits will yield nothing (*syslog-ng*)
- Kernel hardening (MAC: LIDS, BSD call tracing)
 - E.g. bind the sniffer to only write a single file
- IP-less bridge (kernel with no networking)
- *Samhain* – covert HIDS
 - Invisible steg-protected tripwire clone with remote reporting
- Automated response scripts
 - Shutdown if something happens

Basic Victim Setup

- Default install of:
 - RedHat Linux 7.x
 - Solaris Intel/SPARC
 - Free/OpenBSD
- Network services
 - www, ftp, named, pop3, telnet, ssh, sendmail, etc
- User accounts
- “Tripwire” and AIDE

Covert Victim Monitoring

- Simple
 - Bash UDP trojan
- Medium
 - LKM local keylogger (Phrack)
 - Modified *script*
- Medium-high
 - *Sebek* (LKM, UDP, encrypted, spoofed log transfer)
- High

System Testing

- Outbound/inbound connectivity testing
- Flood testing
 - “Fail open” or “fail close”?
 - Both inbound and outbound floods
- Test data capture (snort, tcpdump)
- Test covert monitoring (generation, capture)
- Attack drop testing (GII)
- Test automated response

Honeypot Maintenance Brief

- Watch ACID/**netForensics** console
- Look at daily reports (*logsentry*)
 - Watch for floods, exploits and compromises
- Update IDS signature sets
- Administer victim machine
 - Selective patching might be in order
 - Clean up management traces on victim (and zero disks!)

Data Analysis Brief

- Compromise:
 - Honeynet Project compromise write-up v.0.2
 - Provides a template for compromise analysis write-up
 - Study hacker tools
 - Rootkits, scanners, backdoors, exploits, IRC bots
 - Analyze IRC conversations
 - Investigate sites used by hackers
 - Tool storage sites are of great interest
 - *Tag* hackers

Some Lessons and Cases

- DoS attacks
- Massive scans
- Targeted attacks
- IRC wars
- Unusual rootkits
- Covert backdoors
- New exploits
- Worm spreading

Conclusion

“ To learn the tools, tactics, and motives of the blackhat community, and share the lessons learned.”

Honeypots can successfully achieve that!

Also:

- Great IR and IDS training
- First hand knowledge of new attacks

The End

For more information:

Anton Chuvakin, Ph.D., GCIA

Senior Security Analyst

netForensics

anton@chuvakin.org

anton@netForensics.com

My honeypot papers: <http://www.info-secure.org>