Accenture Security

HONEYPOTS: SETTING THE PERFECT TRAP

Alice Silde

alise.silde@accenture.com

October, 2018

AGENDA:

- INTRODUCTION – DEFINITION AND TYPES
- PROS
- INTERLUDE – CYBERCRIMINAL PROFILING
- CONS
- BUILDING YOUR OWN – DESIGN CONSIDERATIONS AND PRACTICAL IMPLEMENTATION
- NEXT STEP – SELF-ADAPTIVE HONEYPOTS

DEFINITION

A honeypot is a system designed to appear vulnerable and valuable to attackers.

- What happens after a machine is compromised?

PROPERTIES

Available

- Poor security
- Weak passwords
- No encryption
- Many active services

Promising

- Usernames and passwords
- Useful services
- Sensitive information
- Sensitive functionality
- Access to other systems

TYPES

Purpose

- Research
- Detection
- Defense

Interactivity

- High-interaction (honeyservers, honeynets)
- Medium-interaction (host/service emultation)
- Low-interaction (honey-tokens, null listeners)
- (Adaptive?)

BENEFITS

- Fewer false-positives
- No issue with IPv6 or traffic encryption
- Research: Better quality of gathered information
- Defense: Better quality (depth) of gathered information
- Research: Allows to study attacker behavior, popular tools and methods
- Research: Malware analysis and attack appropriation
- Detection: Early warning
- Defense: Distracts attackers from real assets
- Defense: Improve response tools and procedures
- Active Defense: Fine-tune other defensive systems
- Active Defense: Test the effectiveness of standard security controls

TOPICAL INTERLUDE

CYBERCRIMINAL PROFILING

Profiling the enemy is the first step to building the right kind of defenses to stop them (Rossi, B. 2014)

- Attacker profiling increases the reliability of [log] analysis results as the separation of infrastructure properties and attacker properties allows to update these values independently from each other and reflect the ever-changing risk landscape in a more [realistic] way. (Lenin, A., Willemson, J., Sari, D. P. 2014)
- Vulnerability used for entry; skill and experience; stolen/targeted assets; nicknames; language; preferred tools; anti-forensic measures
- Honeypot logs => motive, means, opportunity, environment (victimology), personality (MO and execution style)
- Common tools and activities map to common goals or origins/training
- Focus defense and reduce response/investigation time

ATTACKER TYPES
(NUNES, S., CORREIA, M. 2010)

- Script Kiddies – younger and less skilled, testing a new vulnerability, launching automated scripts
- Botnet Owners – initial motivation was personal power, shifting towards financial gain later. Aiming to compromise maximum nr of computers. Medium level of skills - requires some expertise
- Online Group - searching for unknown vulnerabilities, constructing hacking toolkits for fame and recognition, proud to be a part of a notorious online social community
- Hacker - acts alone, self-studied and skillful, evades detection and covers tracks
- Hired Intruder – commonly corporate espionage, targeted attacks launched at the right time
- Organized Crime - maximize illicit gain, steal identities to commit fraud, ask for ransom
- Terrorists - high skill entry level, mass denial of service or theft of classified data
- Intelligence Services – information warfare

The most popular course of action was to check the software configuration, change the password, check the hardware and/or software configuration (again), download a file, install the downloaded program, and then run it (Ramsbrock, D., Berthier, R., Cukier, M. 2007)

BACK ON TRACK

DRAWBACKS

- Data volume
- Limited field of view
- Unknown attacks not always handled
- May be compromised together with the gathered data (VM breakouts)
- May allow further compromise
- Needs to be well monitored and maintained
- Needs to have limitations, which may alert or inhibit the attackers
- Legal responsibility

DESIGN
EXAMPLES

DESIGN CONSIDERATIONS – MUST HAVE

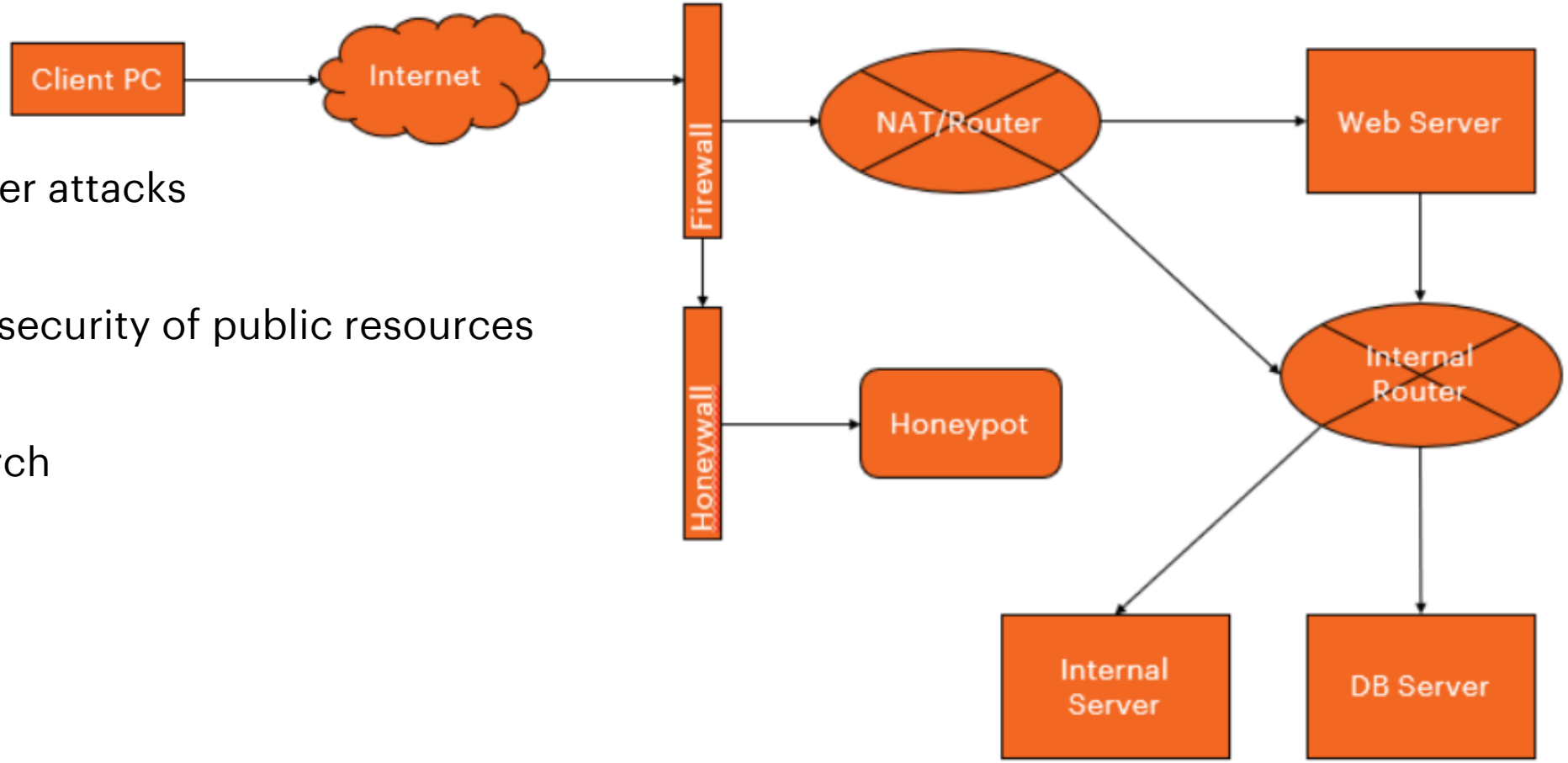
- Basic hardening
- Monitoring
  - Interaction with services
  - Per-session traffic => IPS/Firewall for filtering
- Intrusion Detection/Prevention System
- Firewall
- Remote logging
- Redundant logging
- Credibility/Detection Evasion

DESIGN CONSIDERATIONS – TIP-OFFS

- Too many open ports => Strive to reveal a realistic set of services
- Default service deployments => Remove default files and configs
- Few users and little content => Generate some fake data, based on real assets of value
- Too easy to compromise => Not as secure as production, but not completely open
- Faulty service emulation => Test your deployment and use compatible versions and environments
- Firewall too restrictive => Limit malicious traffic, while allowing ‘some’ outbound traffic (ICMP, DNS, FTP), or limit the number of new connections
- Heavy egress traffic due to remote logging => Use an alternative protocol, such as IPX to export logs

DESIGN CONSIDERATIONS – LOCATION

- Intranet or VPC
  - for catching insider attacks
- DMZ
  - for assessing the security of public resources
- Public Internet
  - for general research



PRACTICAL IMPLEMENTATION

- Services
  - Kippo SSH Honeypot
  - Web application deployed on Apache
  - Honeyd
- Hardening
  - UFW Firewall
  - Snort Intrusion Detection System
- Monitoring
  - Rsyslog
  - Sebek
  - Shadowd
  - Snort
- Detection Evasion
  - MacChanger

KIPPO

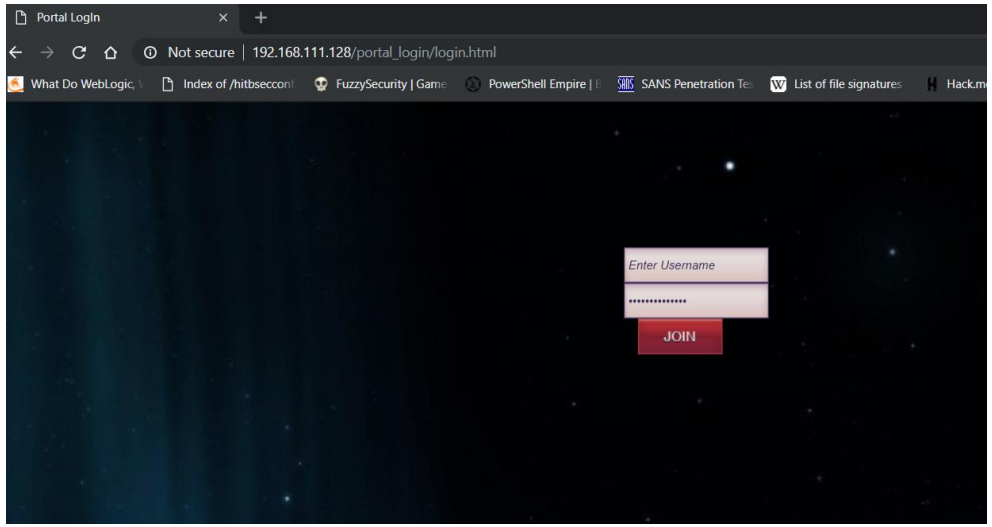
- Install dependencies
- Change SSH port to 2222
  - /etc/ssh/sshd_config
- Create ‘kippo’ user
- Use authbind to allow kippo to listen on 22
- Configure kippo
  - Change kippo.cfg
  - Change start.sh

```
kippo@ubuntu:~/kippo$ ./start.sh
twistd (the Twisted daemon) 15.1.0
Copyright (c) 2001-2015 Twisted Matrix Laboratories.
See LICENSE for details.
Starting kippo in the background...
Removing stale pidfile /home/kippo/kippo/kippo.pid
kippo@ubuntu:~/kippo$ ssh localhost
Password:
Password:
Password:
kippo@localhost's password:
Permission denied, please try again.
kippo@localhost's password:
Permission denied, please try again.
kippo@localhost's password:
Permission denied (password,keyboard-interactive).
```

```
root@ubuntu:/home/kippo/kippo/log# netstat -antp
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN      124316/mysqld
tcp        0      0 127.0.0.1:587           0.0.0.0:*               LISTEN      1872/sendmail: MTA:
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      124109/python
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN      1872/sendmail: MTA:
tcp        0      0 127.0.0.1:42706         127.0.0.1:22            TIME_WAIT   -
```

CUSTOM WEBAPP

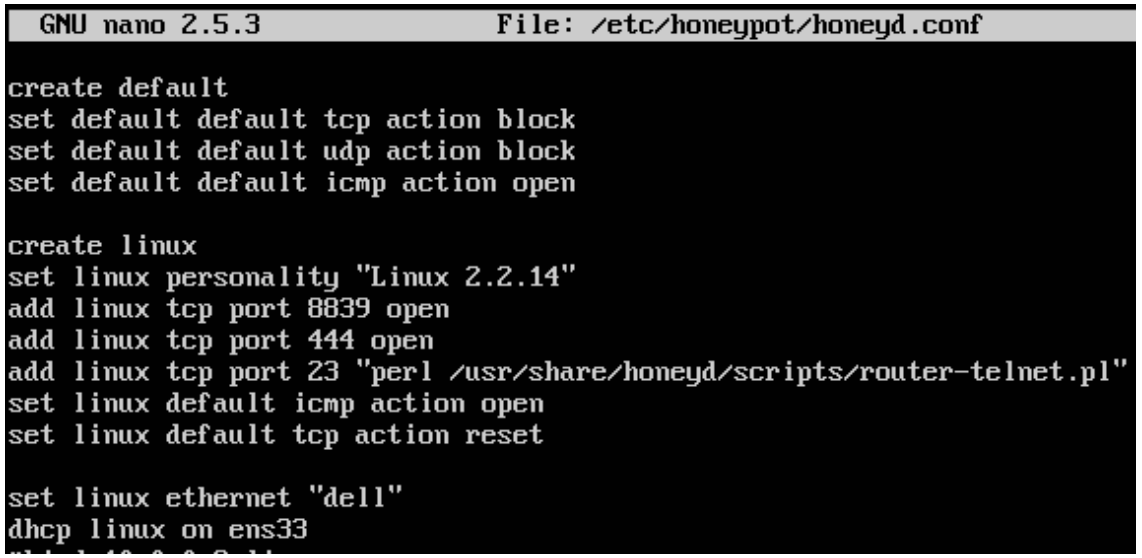
- Use login to gather credentials (and other inputs)
- No company logos
- Enticing name like ‘Portal Login’
- Use .htaccess to prevent access to other web interfaces, such as shadowd console or kippo graph

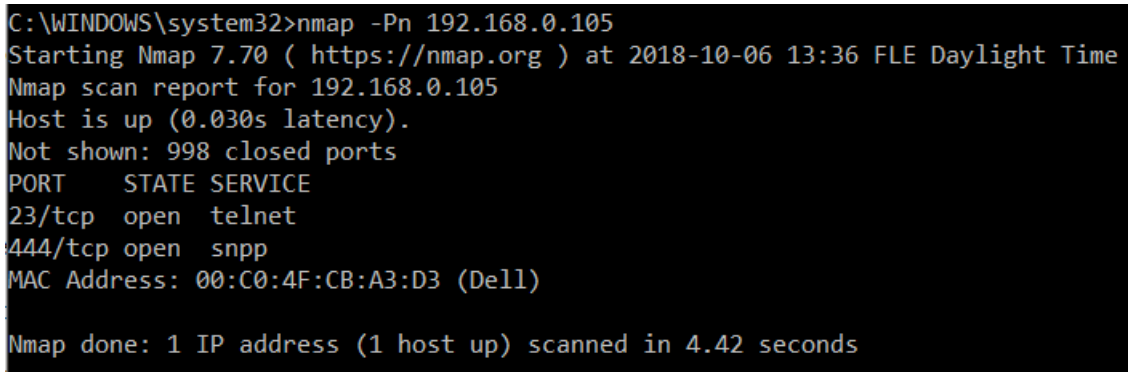



```
 GNU nano 2.5.3                          File: .htaccess

<Files ~ "\.php$">
  #Order allow,deny
  #Allow from 144.36.114.94
  #Deny from all
</Files>
```

HONEYD

- Start VM in bridged mode
- Set up to emulate, for example, telnet and open a couple of ports
  - /etc/honeypot/honeyd.conf
- Enable interface routing on the local subnet:
  - route -n add -net 192.168.111.0/24 ens33
- Start honeyd as a background process:
  - /usr/bin/honeyd -d -f /etc/honeypot/honeyd.conf –l /var/log/honeypot/honeyd.log -i ens33 192.168.111.0/24 &
- Scan the DHCP-assigned address from the host OS on the same network, ensuring it is allowed through the firewall





HONEYD (2)

```
root@ubuntu:~# honeyd[4393]: arp reply 192.168.0.103 is-at 00:c0:4f:8e:cf:08
```

```
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:8099)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:90)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:1175)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:667)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:54328)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:12000)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:2006)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:301)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:2604)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:1688)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:50000)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:2100)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:2800)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:11110)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:1434)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:20031)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:9102)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:5825)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:7070)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:51493)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:6002)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:2602)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:6059)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:8800)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:25735)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:4004)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:7103)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:8222)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:10000)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:5907)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:9415)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:5050)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:41511)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:4125)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:2035)
honeyd[4499]: Killing attempted connection: tcp (192.168.0.104:44014 - 192.168.0.105:3001)
```

UNCOMPLICATED FIREWALL

- Enable the ubuntu firewall
  - sudo ufw enable
- Allow or deny specific ports
  - sudo ufw allow 80/tcp
  - sudo ufw allow 22/tcp
  - sudo ufw allow 23/tcp
  - sudo ufw allow 21/tcp
  - sudo ufw deny 53/tcp
  - sudo ufw deny 5900/tcp
  - sudo ufw deny 513/tcp
- ..or services
  - sudo ufw allow ssh
  - sudo ufw deny snmp

```
root@ubuntu:/usr/share/honeyd# ufw status
Status: active

To                         Action      From
--                         ------      ----
Apache                     ALLOW       Anywhere
22/tcp                     ALLOW       Anywhere
23/tcp                     ALLOW       Anywhere
Apache (v6)                ALLOW       Anywhere (v6)
22/tcp (v6)                ALLOW       Anywhere (v6)
23/tcp (v6)                ALLOW       Anywhere (v6)
```

SNORT

- Install Snort
- Download and install rules
- Configure Snort
- Validate configuration with:
  - snort -T -i eth0 -c /etc/snort/snort.conf
- Create custom Snort rules
  - https://blog.rapid7.com/2016/12/09/understanding-and-configuring-snort-rules/
- Test by running:
  - snort -A console -q -c /etc/snort/snort.conf -i ens33
- Configure auto-start by creating /lib/systemd/system/snort.service
- Use systemctl to start Snort and check its status
- Read Snort logs with:
  - snort –r /var/log/snort/<logname>

SNORT (2)

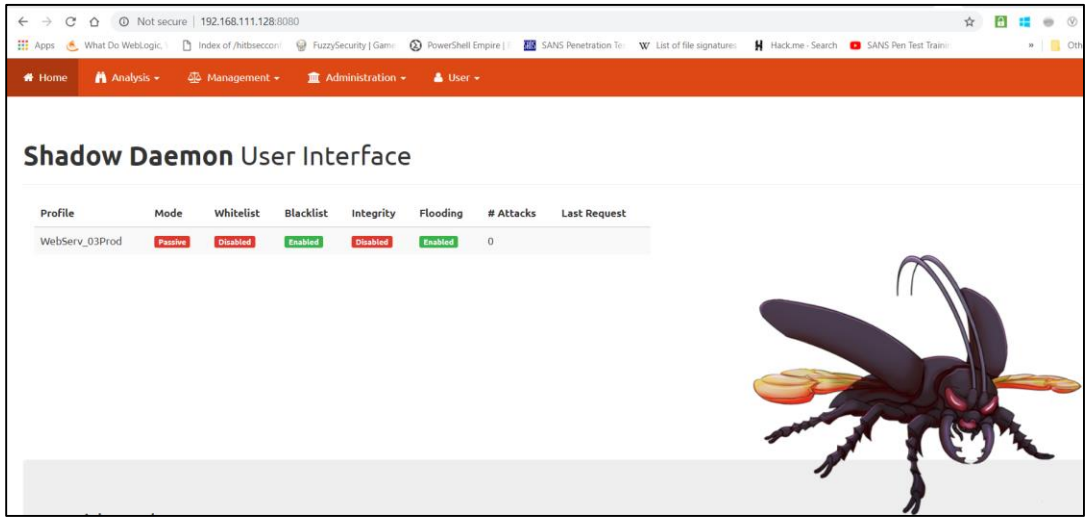
```
 GNU nano 2.5.3                File: /etc/snort/rules/local.rules

# $Id: local.rules,v 1.11 2004/07/23 20:15:44 bmc Exp $
# ----------------
# LOCAL RULES
# ----------------
# This file intentionally does not come with signatures.  Put your local
# additions here.
alert tcp any any -> $HOME_NET 21 (msg: "FTP connection attempt"; sid:1000001; rev:1;)
alert icmp any any -> $HOME_NET any (msg:"ICMP connection attempt"; sid:1000002; rev:1;)
alert tcp any any -> $HOME_NET 22 (msg:"SSH connection attempt"; sid:1000003; rev:1;)
```

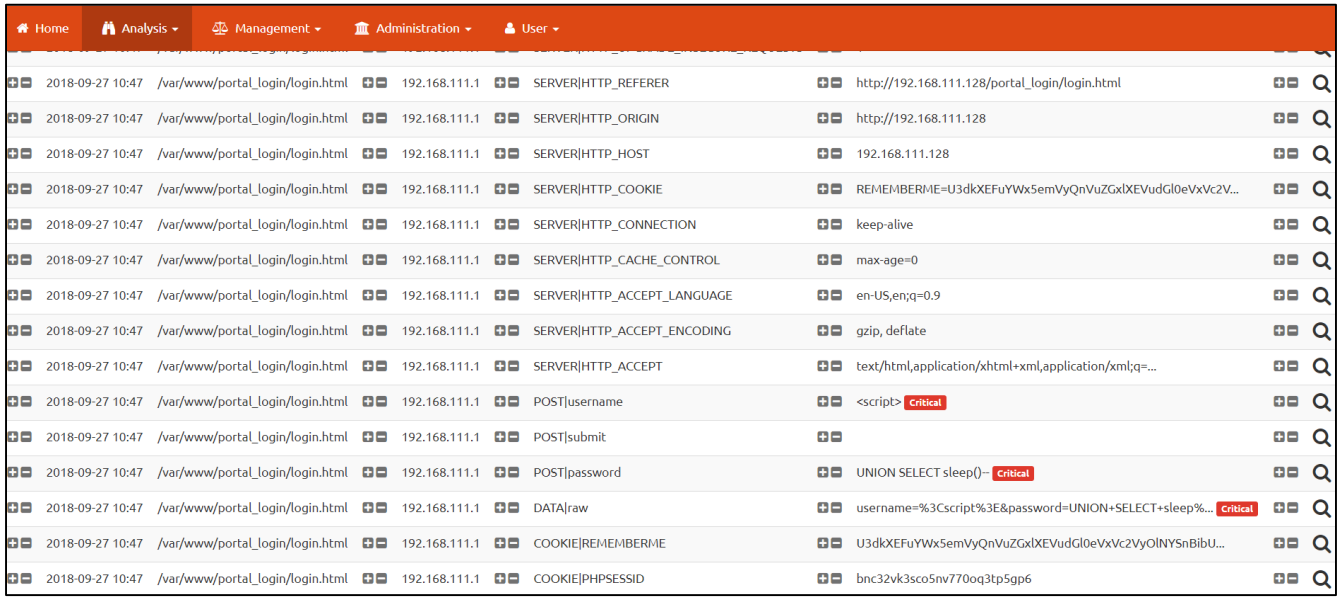
```
C:\Users\alise.silde>ping 192.168.111.128

Pinging 192.168.111.128 with 32 bytes of data:
Reply from 192.168.111.128: bytes=32 time<1ms TTL=64
Reply from 192.168.111.128: bytes=32 time=1ms TTL=64
Reply from 192.168.111.128: bytes=32 time<1ms TTL=64
Reply from 192.168.111.128: bytes=32 time=15ms TTL=64

Ping statistics for 192.168.111.128:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 15ms, Average = 4ms
```

```
root@ubuntu:~/software/snort/snort-2.9.11.1# snort -A console -q -c /etc/snort/snort.conf -i ens33
10/04-09:13:49.403542  [**] [1:1000002:1] ICMP connection attempt [**] [Priority: 0] {ICMP} 192.168.
111.1 -> 192.168.111.128
10/04-09:13:49.403580  [**] [1:1000002:1] ICMP connection attempt [**] [Priority: 0] {ICMP} 192.168.
111.128 -> 192.168.111.1
10/04-09:13:50.406433  [**] [1:1000002:1] ICMP connection attempt [**] [Priority: 0] {ICMP} 192.168.
111.1 -> 192.168.111.128
10/04-09:13:50.406480  [**] [1:1000002:1] ICMP connection attempt [**] [Priority: 0] {ICMP} 192.168.
111.128 -> 192.168.111.1
10/04-09:13:51.411102  [**] [1:1000002:1] ICMP connection attempt [**] [Priority: 0] {ICMP} 192.168.
111.1 -> 192.168.111.128
10/04-09:13:51.411131  [**] [1:1000002:1] ICMP connection attempt [**] [Priority: 0] {ICMP} 192.168.
111.128 -> 192.168.111.1
10/04-09:13:52.430293  [**] [1:1000002:1] ICMP connection attempt [**] [Priority: 0] {ICMP} 192.168.
111.1 -> 192.168.111.128
10/04-09:13:52.430331  [**] [1:1000002:1] ICMP connection attempt [**] [Priority: 0] {ICMP} 192.168.
111.128 -> 192.168.111.1
```

SHADOWD

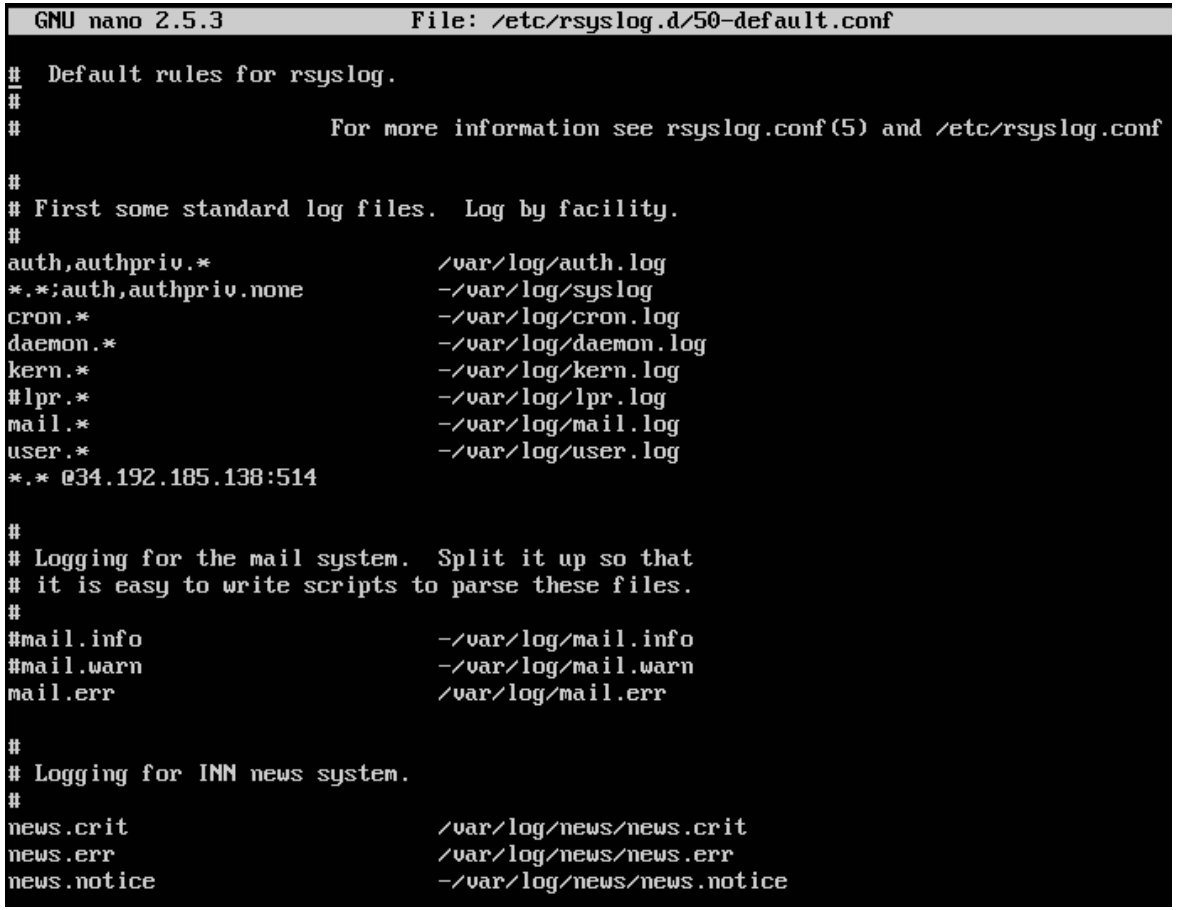
- Install Shadowd and its PHP connector
- Configure Shadowd as a honeypot (follow the guidelines)
- Change php.ini to include the connector and restart apache
- Set up console user:
  - ./shadowdctl exec web ./app/console swd:register --admin --name=arg (--email=arg)
- Access console on port 8080
- Create application profile in ‘learning’ mode to log all requests
- Use rules from Git or define your own





RSYSLOG

- Configure the receiving server in /etc/rsyslog.conf
  - $ModLoad imtcp
  - $InputTCPServerRun 514
- Configure the honeypot server in /etc/rsyslog.d/50-default.conf
  - *.* @<server_IP>:514
- Restart rsyslog on both



MACCHANGER

- Install
- List vendors to find appropriate first 3 bytes for the MAC
- Set MAC to a spoofed value
- Use a permanent MAC address for DHCP reliability

```
0012 - 00:05:5d - D-Link DWL-650, DWL-650H
0013 - 00:06:25 - Linksys WPC11 v2.5, D-Link DCF-650W, Linksys WPC11 v3
root@ubuntu:~# macchanger --help
GNU MAC Changer
Usage: macchanger [options] device

  -h,  --help                  Print this help
  -V,  --version               Print version and exit
  -s,  --show                  Print the MAC address and exit
  -e,  --ending                Don't change the vendor bytes
  -a,  --another               Set random vendor MAC of the same kind
  -A                           Set random vendor MAC of any kind
  -p,  --permanent             Reset to original, permanent hardware MAC
  -r,  --random                Set fully random MAC
  -l,  --list[=keyword]        Print known vendors
  -b,  --bia                   Pretend to be a burned-in-address
  -m,  --mac=XX:XX:XX:XX:XX:XX
       --mac XX:XX:XX:XX:XX:XX  Set the MAC XX:XX:XX:XX:XX:XX

Report bugs to https://github.com/alobbs/macchanger/issues
root@ubuntu:~# macchanger --mac=00:06:25:0C:33:54
GNU MAC Changer
Usage: macchanger [options] device

Try `macchanger --help' for more options.
root@ubuntu:~# macchanger --mac=00:06:25:0C:33:54 ens33
Current MAC:   00:0c:29:61:48:08 (VMware, Inc.)
Permanent MAC: 00:0c:29:61:48:08 (VMware, Inc.)
New MAC:       00:06:25:0c:33:54 [wireless] (Linksys WPC11 v2.5, D-Link DCF-650W, Linksys WPC11 v3)
```

```
ens33     Link encap:Ethernet  HWaddr 00:06:25:0c:33:54
          inet addr:192.168.111.128  Bcast:192.168.111.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe61:4808/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:373 errors:0 dropped:0 overruns:0 frame:0
          TX packets:428 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:233917 (233.9 KB)  TX bytes:53466 (53.4 KB)
```

PRACTICAL IMPLEMENTATION

- Services
  - Kippo SSH Honeypot
  - Web application deployed on Apache
  - Honeyd
- Hardening
  - UFW Firewall
  - Snort Intrusion Detection System
- Monitoring
  - Rsyslog
  - ~~Sebek~~
  - Shadowd
  - Snort
- Detection Evasion
  - MacChanger

T-POT

FUTURE IMPROVEMENTS

- Realistic file structure
- HTTPS for Web applications
- Separate kippo and shadowd server and console
- Separate syslog server
- Vulnerable web application frameworks
- Additional services, like FTP, SMTP, DNS
- Better IDS and WAF configuration
- Covert exfiltration of log data

EPI-
LOGUE

SELF-ADAPTIVE HONEYPOTS

- Game theory + Reinforcement learning
- A system that defends itself => Challenge
- May attract highly skilled attackers
- Provides better post-compromise information
- Attacker: leave the honeypot, retry the executed command, select an alternative command or insult the honeypot
- Honeypot: allow advances of an attacker, it can block the advance, substitute the command or insult the attacker
- Insults => Reverse Turing Test

SELF-ADAPTIVE HONEYPOTS (2)

- Logs network traffic, executed processes, system calls, system calls related to the VM; Transmitted via a separate network interface
- Queued traffic to avoid exceeding normal bandwidth
- Aguri for traffic capture and aggregation, allowing to create traffic profiles
- Directly patched Linux kernel for interfering with attacker commands and minimizing detectability
- Fast concurrent reinforcement learning – to account for the fact that attackers can learn

Q&A THANK YOU!

ACCENTURE LATVIA SECURITY PRACTICE

CYBER SECURITY TESTING · VULNERABILITY MANAGEMENT

SECURITY ADVISORY SERVICES · DATA PRIVACY & GDPR · SECURITY RISK ASSESSMENTS

ISO 27001 · SPLUNK

CREST Registered Penetration Tester

AppScan IBM Security

ISO-27001 Lead Auditor

CISA Certified Information Systems Auditor

splunk> Splunk Certified Power User

splunk> Splunk Certified Admin

C|EH Certified Ethical Hacker

C)PTE Certified Penetration Testing Engineer

C)DFE Certified Digital Forensics Examiner

Contact Latvia.SecurityST@accenture.com

RESOURCES

How To Guides and Tools

- SANS How to Build A Honeypot https://www.giac.org/paper/gsec/2986/building-simple-honeypot-windows/104999
- How to Build and Use a Honeypot https://www.infosecwriters.com/text_resources/pdf/build_and_use_honeypot.pdf
- List of honeypot tools: http://securitytools.wikidot.com/honeypot-utilities, https://github.com/paralax/awesome-honeypots
- HoneyD Configuration http://www.honeyd.org/configuration.php, https://linuxsecurityblog.com/2018/06/25/honeyd-tutorial-part-1-getting-started/, https://bruteforcelab.com/getting-started-honeyd.html,
- Deception ToolKit http://all.net/dtk/download.html
- Shadowd https://shadowd.zecure.org/tutorials/honeypots/
- Kippo https://bruteforcelab.com/installing-kippo-ssh-honeypot-on-ubuntu.html, https://www.digitalocean.com/community/tutorials/how-to-install-kippo-an-ssh-honeypot-on-an-ubuntu-cloud-server
- Kippo-Graph https://github.com/ikoniaris/kippo-graph
- Glastopf https://media.readthedocs.org/pdf/glastopf/latest/glastopf.pdf
- Dinoaea http://www.edgis-security.org/honeypot/dionaea/, https://github.com/andrewmichaelsmith/honeypot-setup-script/
- UFW http://manpages.ubuntu.com/manpages/cosmic/en/man8/ufw.8.html
- Rsyslog https://vexxhost.com/resources/tutorials/how-to-setup-remote-system-logging-with-rsyslog-on-ubuntu-14-04-lts/, https://www.digitalocean.com/community/tutorials/how-to-centralize-logs-with-rsyslog-logstash-and-elasticsearch-on-ubuntu-14-04
- Snort https://www.snort.org/#documents, https://www.upcloud.com/support/installing-snort-on-ubuntu/, https://blog.rapid7.com/2017/01/11/how-to-install-snort-nids-on-ubuntu-linux/, https://blog.rapid7.com/2016/12/09/understanding-and-configuring-snort-rules/
- Sebek https://projects.honeynet.org/sebek/wiki/Building%20and%20Installing%20Sebek%20client%20in%20Ubuntu%20Server%207.10

Projects

- HoneyDrive https://bruteforcelab.com/honeydrive
- The HoneyNet Project https://www.honeynet.org/project
- TPlot Honeypot Project http://dtag-dev-sec.github.io/
- The «Google Hack» Honeypot http://ghh.sourceforge.net/
- Hacker Profiling Project http://www.isecom.org/research/hpp.html
- Project HoneyPot https://www.projecthoneypot.org/
- Log Sharing Project http://log-sharing.dreamhosters.com/

REFERENCES

[1] OccupyTheWeb. 2014. How to Set Up a Honeypot & How to Avoid Them. https://null-byte.wonderhowto.com/how-to/hack-like-pro-set-up-honeypot-avoid-them-0153391/

[2] TheInnocent. 2017. How to Set Up Your Own Honeypot? https://www.deepdotweb.com/2017/08/24/how-to-setup-your-own-honeypot/

[3] Rossi, B. 2014. How to Set up a Cybersecurity Honeypot for Your Business. http://www.information-age.com/how-set-cybersecurity-honeypot-your-business-123458520/

[4] Stevenson, J. 2016. What's a Honeypot, and How to Set One up in Under an Hour. http://www.hackinginsider.com/2016/03/whats-a-honeypot-and-how-to-set-one-up-in-under-an-hour/

[5] Dargin, M. 2017. Increase Your Network Security: Deploy a Honeypot. https://www.networkworld.com/article/3234692/lan-wan/increase-your-network-security-deploy-a-honeypot.html

[6] Spitzner, L. 2003. Honeypot Farms. https://www.symantec.com/connect/articles/honeypot-farms

[7] Pyorre, J. 2016. Building a Better Honeypot Network. https://www.youtube.com/watch?v=g1WXml_6NOE

[8] 1aN0rmus. 2012. HoneyDrive at TekTip. https://www.youtube.com/watch?v=Ibe_uDojUnc

[9] Carrasco, I. 2017. Monitoring Honeypots. https://blog.pandorafms.org/honeypots/

[10] Xie, R. 2015. Hunting For Honeypot Attackers: A Data Scientist's Adventure. https://www.endgame.com/blog/technical-blog/hunting-honeypot-attackers-data-scientist-s-adventure

[11] Nunes, S., Correia, M. 2010. From Risk Awareness to Security Controls: Benefits of Honeypots to Companies. https://pdfs.semanticscholar.org/presentation/2e9a/54a53db2b076a2384b53c927edaea63934a2.pdf

[12] Smith, S. D. 2016. Catching Flies: A Guide to the Various Flavors of Honeypots. https://www.sans.org/reading-room/whitepapers/attacking/catching-flies-guide-flavors-honeypots-36897

[13] Diebold, P., Hess, A., Schaefer, G. 2005. A Honeypot Architecture for Detecting and Analyzing Unknown Network Attacks. In Proc. Of 14th Kommunikation in Verteilten Systemen 2005 (KiVS05), Kaiserslautern, Germany, February 2005.

[14] McGrew, R., Vaughn, R. B. 2006. Experiences With Honeypot Systems: Development, Deployment, and Analysis. Proceedings of the 39th Hawaii International Conference on System Sciences.

[15] Wagener, G. 2011. Self-Adaptive Honeypots Coercing and Assessing Attacker Behaviour. Institut National Polytechnique de Lorraine.

[16] Lenin, A., Willemson, J., Sari, D. P. 2014. Attacker Profiling in Quantitative Security Assessment Based on Attack Trees. Tallinn University of Technology.

[17] Ramsbrock, D., Berthier, R., Cukier, M. 2007. Profiling Attacker Behaviour Following SSH Compromises. 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'07).

[18] Long, L. A. 2012. Profiling Hackers. https://www.sans.org/reading-room/whitepapers/hackers/profiling-hackers-33864

[19] Rossi, B. 2014. Know Your Cyber-Attacker: Profiling a Hacker. https://www.information-age.com/know-your-cyber-attacker-profiling-hacker-123457840/

[20] AdminUser. 2015. The Hacker – Profiling Cyber Criminals. https://www.securityri.com/the-hacker-profile/

[21] Biancuzzi, F. 2005. Inside the Hacker's Profiling Project. https://www.linux.com/news/inside-hackers-profiling-project

[22] Rogers, M. 2008. Psychological Crime Scene Analysis: Cyber Crime Scenes. Dept. of Computer Technology at Purdue University. https://issuu.com/steve_austin/docs/hacker_profiling

[23] Bakos, G. Honeypots and the Enterprise: Intelligence-Based Risk Management. Institute for Security Technology Studies of Dartmouth College. http://www.ists.dartmouth.edu/library/97.pdf

[24] Provos, N. 2004. Honeyd: A Virtual Honeypot Daemon. http://metro.citi.umich.edu/u/provos/papers/honeyd-eabstract.pdf

[25] Bar, A., Shapira, B., Rokach, L., Unger, M. 2016. Identifying Attack Propagation Patterns in Honeypots using Markov Chains Modeling and Complex Network Analysis. IEEE International Conference on Software Science, Technology and Engineering.

[26] Kwan, L., Ray, P., Stephens, G. 2008. Towards a Methodology for Profiling Cyber Criminals. Proceedings of the 41st Hawaii International Conference on System Sciences.

[27] Nunes, S., Correia, M. 2010. Web Application Risk Awareness with High Interaction Honeypots.