A Whirlwind Introduction to Honeypots

Marcus J. Ranum

<mjr@ranum.com>

What is a honeypot?

- A security resource thats value lies in being attacked, probed, or compromised
  - A honeypot is more a *state of mind* than a specific implementation
  - You can set up a production system to be a honeypot or you can laboriously construct your own dedicated honeypot system

Cool Record:

- Shortest time in which a honeypot was attacked and compromised:
  - 17 seconds from connection to the Internet (by a worm)

Two Kinds of Honeypots

- Research
- Production

Research Honeypots

- Research honeypots are for the hardcore hacker hunters
  - Learn what the bad guys are doing
  - Study their methods
  - Capture their keystrokes
  - Capture their tools
  - Monitor their conversations
- Takes a ***lot*** of work!!

Production Honeypots

- Production honeypots are more in line with conventional intrusion detection
  - Identify hostile activity
  - Generate an alert
  - Capture a minimum of data
- Takes a ***minimum*** amount of work!

Interaction

- Low Interaction
- Middle Interaction (chroot/jail)
- High Interaction

Low Interaction Honeypots

- Present the Bad Guy with emulators of vulnerable programs
  - Summarize or capture limited interactions
  - Simpler to deploy (no system administration)
  - Less likely to be penetrated
  - More likely to be detected by the Bad Guy
- Tend to be production honeypots

Middle Interaction Honeypots

- Tend to be application-centric chroot/jail systems
  - It is hard to set up a reliable chroot/jail
  - Limited usefulness: no chroot for Windows
  - Requires exhaustive specialized knowledge (chroot semantics vary between versions of UNIX)
  - Really more of an operational / application security process than a honeypot
- Too much work for most people

High Interaction Honeypots

- Present the Bad Guy with a complete operational environment that you assume will be penetrated completely
  - Monitor everything they do
  - Install data control to reduce likelihood of outgoing attacks
  - Collect their tools and keystrokes
- Tend to be research honeypots

Risk

- What are we afraid of?
  - Primary fear - someone uses our honeypot as a jumping-off point for an attack against someone and does them harm
    - E.g.: a distributed denial of service attack against cnn.com that came from us!
  - Secondary fear - someone uses our honeypot to attack our own systems

Mitigating Risk

- Risk is greater with high interaction honeypots
  - Must use traffic control (e.g.: firewall filtering etc - see the honeynet rules on their site) to prevent jump-off attacks
- Risk still somewhat present but largely eliminated in low interaction honeypots

Fingerprinting

- Fingerprinting occurs when a Bad Guy realizes that he’s on a honeypot
  - Very rare in research honeypots
  - More common with low interaction honeypots
- May trigger destructive attacks
- May trigger Bad Guy simply vanishing
- There will almost *always* be a *potential* fingerprint on any honeypot!

Minimizing Fingerprinting

- First: ***decide if you care!***
  - Production honeypots have succeeded in their mission by the time they are fingerprinted!
  - Research honeypots care more
- Provide the best possible emulation of a real vulnerable system
  - Usually, that's simply a matter of providing a real vulnerable system!

Legal Issues

- Entrapment
- Privacy
- Attacks against 3rd parties

Entrapment

- Entrapment is a *defense* **not** an *offense*!
  - Nobody can prosecute you for "entrapment"
  - Defendant's lawyer might try to get their client off because the plaintiff "entrapped" the defendant
  - Plaintiff must be law enforcement / Gov't
  - Plaintiff must have modified the defendant's behavior
    - So don't promote your honeypot; they will come

Privacy

- This is actually the one you have to worry about most!
  - Lots of legal issues regarding privacy
    - Somewhat contradictory
    - Common carrier versus network administrator
    - What about conversations with multiple parties?
    - ECPA (electronic communications privacy act)
  - No case law regarding honeypots

Attacks on 3rd Parties

- Potential for liability if a 3rd party is injured by an attacker that used your honeypot as a jump-off point
  - Attempt to show diligence in protecting 3rd parties using data control
  - Nobody (that we've heard of yet) has gone after someone for being a jump-off point, whether their system was a honeypot or just a normal, insecure machine

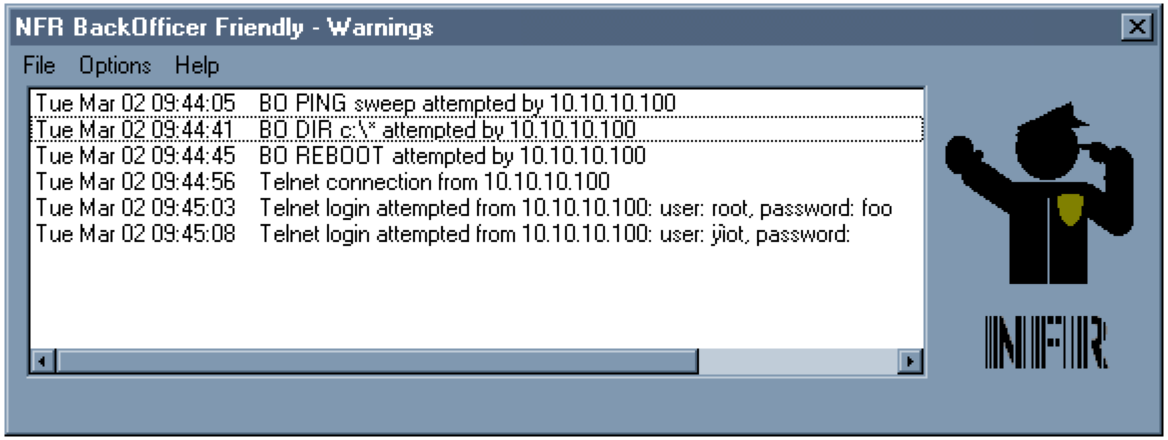
Techniques and Tools

- These are just a few examples
- There are loads of tools and techniques for building honeypots
  - Remember - they should all be different enough that they are harder to fingerprint
  - It is the unknown defense that may block the unknown attack (no, Sun Tzu didn't say that…)

NFR BOF (backofficer friendly)

- Low-interaction ***free*** honeypot for Windows (9*,2*) that installs in seconds
  - Emulates a machine infected with BackOrifice 1.0
  - Emulates other services with fake replies:
    - http
    - telnet
    - ftp
    - pop / imap

BOF Alerts



```
Tue Mar 02 09:44:05   BO PING sweep attempted by 10.10.10.100
Tue Mar 02 09:44:41   BO DIR c:\* attempted by 10.10.10.100
Tue Mar 02 09:44:45   BO REBOOT attempted by 10.10.10.100
Tue Mar 02 09:44:56   Telnet connection from 10.10.10.100
Tue Mar 02 09:45:03   Telnet login attempted from 10.10.10.100: user: root, password: foo
Tue Mar 02 09:45:08   Telnet login attempted from 10.10.10.100: user: ÿiot, password:
```

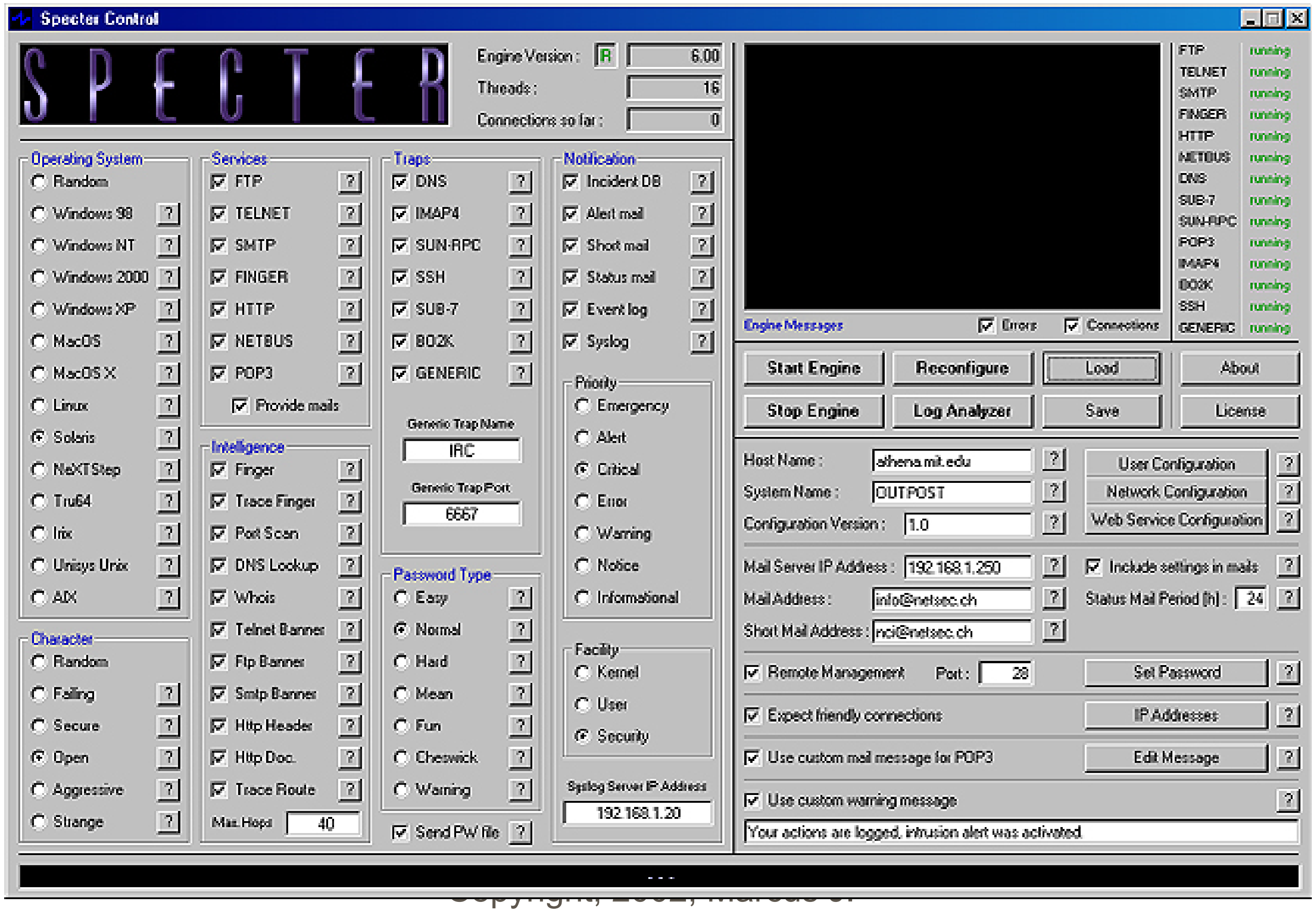
BOF From the Other Side

```
BO>host 10.10.10.1
New host: 10.10.10.1:31337
BO:10.10.10.1>dir
------- Packet received from 10.10.10.1 port 31337 -------
Error 53:The network path was not found opening file c:\*
------- End of data -------
BO:10.10.10.1>reboot
------- Packet received from 10.10.10.1 port 31337 -------
Naughty, naughty. Bad hacker! No donut!
------- End of data -------
BO:10.10.10.1>quit
```

Spectre

- Low-interaction commercial honeypot for windows NT systems
  - Emulates a variety of operating systems at the application level
    - Does not perform IP-level emulation
  - Includes centralized management and reporting
  - Includes session capture and logging

Spectre U/I


Specter Control

SPECTER

Engine Version: R 6.00
Threads: 16
Connections so far: 0

Operating System
- Random
- Windows 98
- Windows NT
- Windows 2000
- Windows XP
- MacOS
- MacOS X
- Linux
- [✓ Solaris]
- NeXTStep
- Tru64
- Irix
- Unisys Unix
- AIX

Character
- Random
- Failing
- Secure
- [✓ Open]
- Aggressive
- Strange

Services
- ✓ FTP
- ✓ TELNET
- ✓ SMTP
- ✓ FINGER
- ✓ HTTP
- ✓ NETBUS
- ✓ POP3
- ✓ Provide mails

Intelligence
- ✓ Finger
- ✓ Trace Finger
- ✓ Port Scan
- ✓ DNS Lookup
- ✓ Whois
- ✓ Telnet Banner
- ✓ Ftp Banner
- ✓ Smtp Banner
- ✓ Http Header
- ✓ Http Doc.
- ✓ Trace Route

Max. Hops: 40

Traps
- ✓ DNS
- ✓ IMAP4
- ✓ SUN-RPC
- ✓ SSH
- ✓ SUB-7
- ✓ BO2K
- ✓ GENERIC

Generic Trap Name: IRC
Generic Trap Port: 6667

Password Type
- Easy
- [✓ Normal]
- Hard
- Mean
- Fun
- Cheswick
- Warning

✓ Send PW file

Notification
- ✓ Incident DB
- ✓ Alert mail
- ✓ Short mail
- ✓ Status mail
- ✓ Event log
- ✓ Syslog

Priority
- Emergency
- Alert
- [✓ Critical]
- Error
- Warning
- Notice
- Informational

Facility
- Kernel
- User
- [✓ Security]

Syslog Server IP Address: 192.168.1.20

Engine Messages ✓ Errors ✓ Connections

FTP	running
TELNET	running
SMTP	running
FINGER	running
HTTP	running
NETBUS	running
DNS	running
SUB-7	running
SUN-RPC	running
POP3	running
IMAP4	running
BO2K	running
SSH	running
GENERIC	running

Host Name: athena.mit.edu
System Name: OUTPOST
Configuration Version: 1.0
Mail Server IP Address: 192.168.1.250
Mail Address: info@netsec.ch
Short Mail Address: nci@netsec.ch
✓ Include settings in mails
Status Mail Period [h]: 24
✓ Remote Management Port: 28
✓ Expect friendly connections
✓ Use custom mail message for POP3
✓ Use custom warning message
Your actions are logged, intrusion alert was activated.

Mantrap

- High-interaction commercial honeypot
- Builds multiple virtual installations of Solaris
  - Each looks like a complete system
  - All interactions with the "jail" systems are recorded
  - Remotely manageable (via ssh into the host system)

Port Suckers

- Netcat is a great tool for collecting data on a port
  - `nc -l -p 80 > capture.txt`
  - This will capture all traffic coming in on port 80 to the output file
  - Can easily be harnessed into a .BAT file

Netcat Port Sucker

```
@echo off
echo port sucker off and running
echo > capture.txt
:top
  nc -l -p 80 >> capture.txt
  netstat -a | find "_wait" >> capture.txt
goto top
```

Honeyd

Overview of Honeyd

- Developed by Niels Provos
- Ver 0.1 released April, 2002
- Low interaction honeypot
- OpenSource for Unix systems
- Monitor entire networks of millions of systems
- Emulation at both application and IP stack level, 473 different systems

Honeyd’s Value

- Detection
- Research (blackholing)

Advantages

- Free
- Access to source code
- Share emulated services
- Monitor millions of IP addresses, exponentially increasing value

Disadvantages

- No-support
- No nice, easy to use GUI

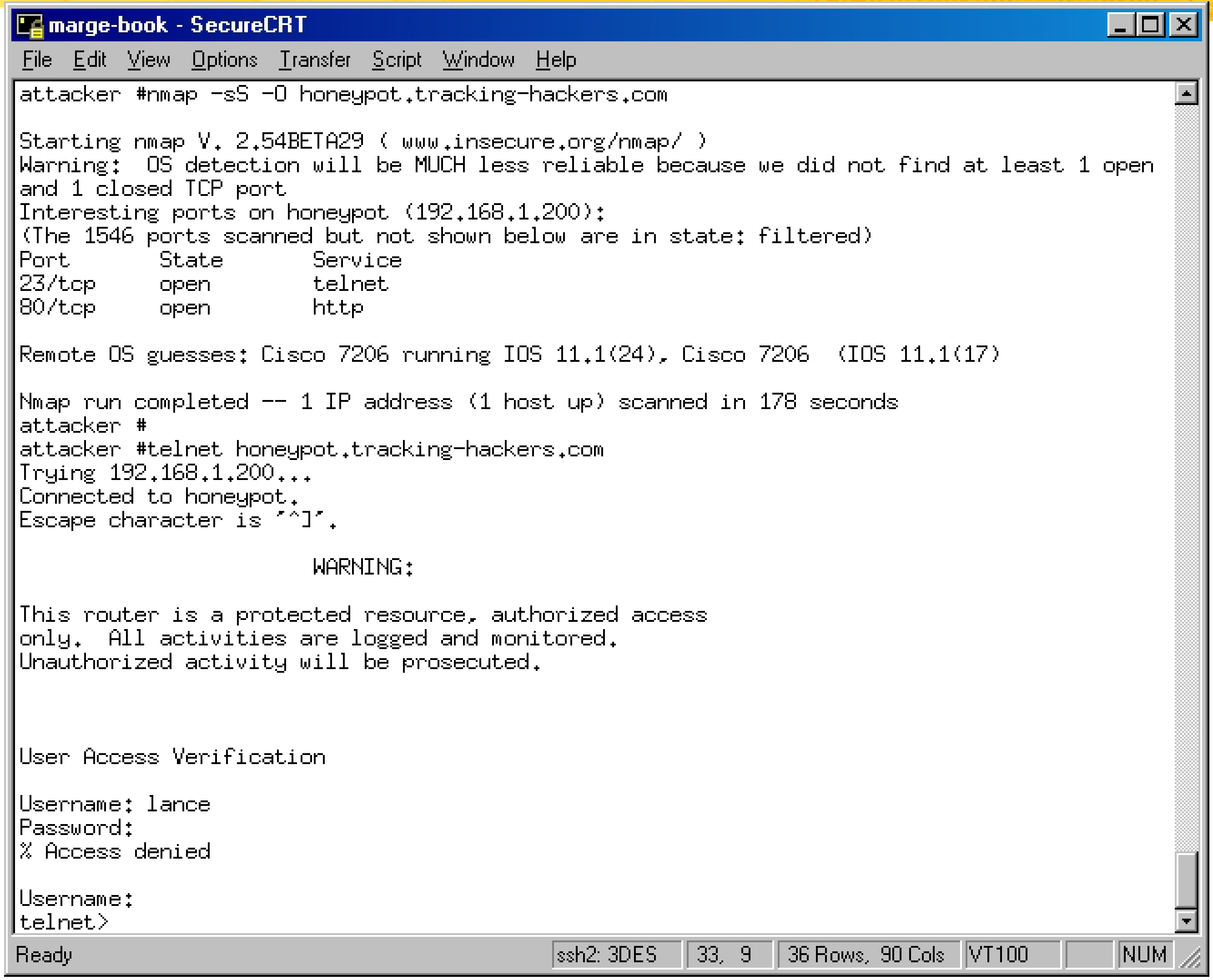
How Honeypot Works

- Monitors network for connections to non-existent IP addresses.
- When sees connection assumes attack.
- Assumes IP address of victim, interacts with attacker.

Honeyd Process

- Honeyd process
  - Receives attack
  - Assumes IP address of victim
  - Starts up emulated service
  - Interacts with attacker
  - Exists emulated service after attack is finished.

Cisco Router



```
attacker #nmap -sS -O honeypot.tracking-hackers.com

Starting nmap V. 2.54BETA29 ( www.insecure.org/nmap/ )
Warning:  OS detection will be MUCH less reliable because we did not find at least 1 open
and 1 closed TCP port
Interesting ports on honeypot (192.168.1.200):
(The 1546 ports scanned but not shown below are in state: filtered)
Port       State       Service
23/tcp     open        telnet
80/tcp     open        http

Remote OS guesses: Cisco 7206 running IOS 11.1(24), Cisco 7206  (IOS 11.1(17)

Nmap run completed -- 1 IP address (1 host up) scanned in 178 seconds
attacker #
attacker #telnet honeypot.tracking-hackers.com
Trying 192.168.1.200...
Connected to honeypot.
Escape character is '^]'.

                       WARNING:

This router is a protected resource, authorized access
only.  All activities are logged and monitored.
Unauthorized activity will be prosecuted.



User Access Verification

Username: lance
Password:
% Access denied

Username:
telnet>
```

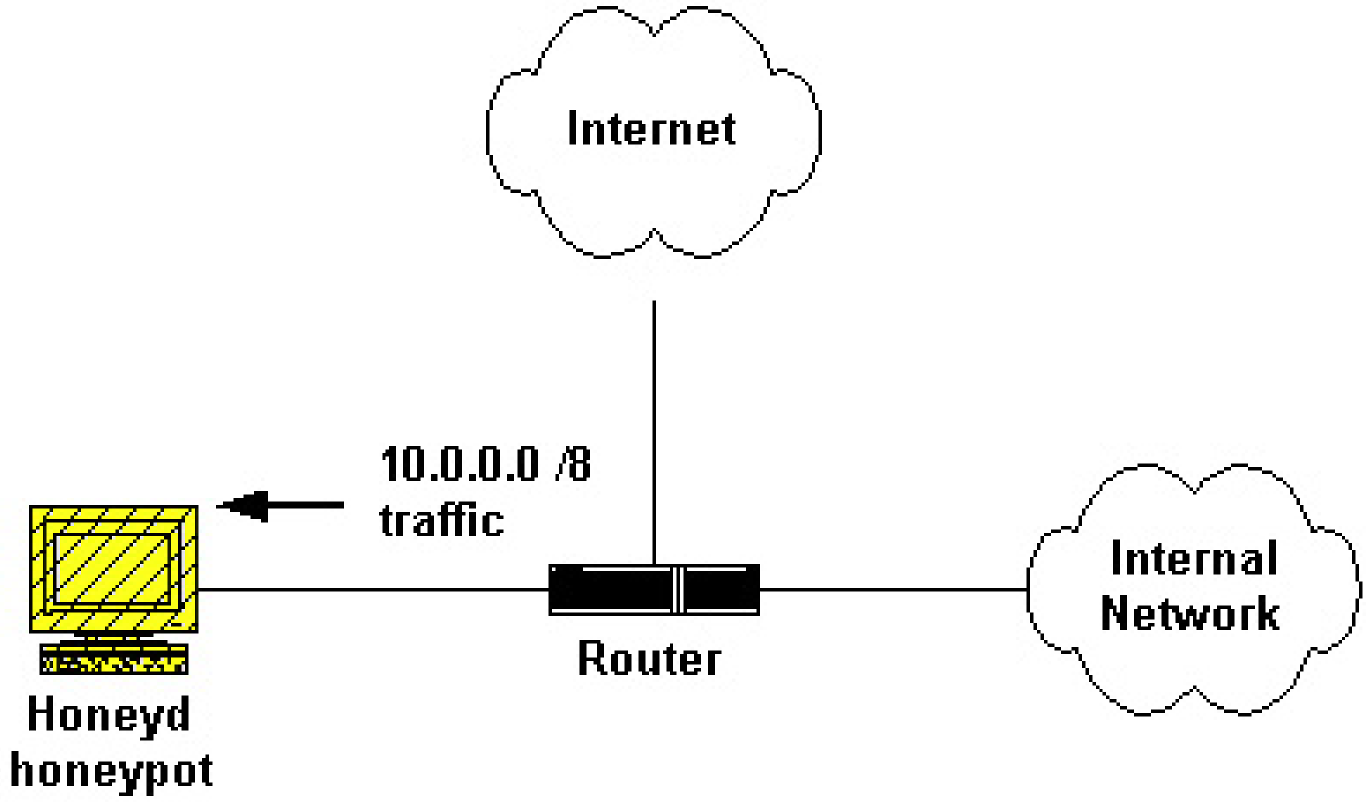
Receiving Attacks

- Does not have multiple IP's at the same time. Instead, two ways Honeyd can receive attacks.
  - Blackholing
  - ARP Spoofing

Blackholing

- Identify entire networks with non-existent systems.
- Route all traffic from that network to Honeyd honeypot.

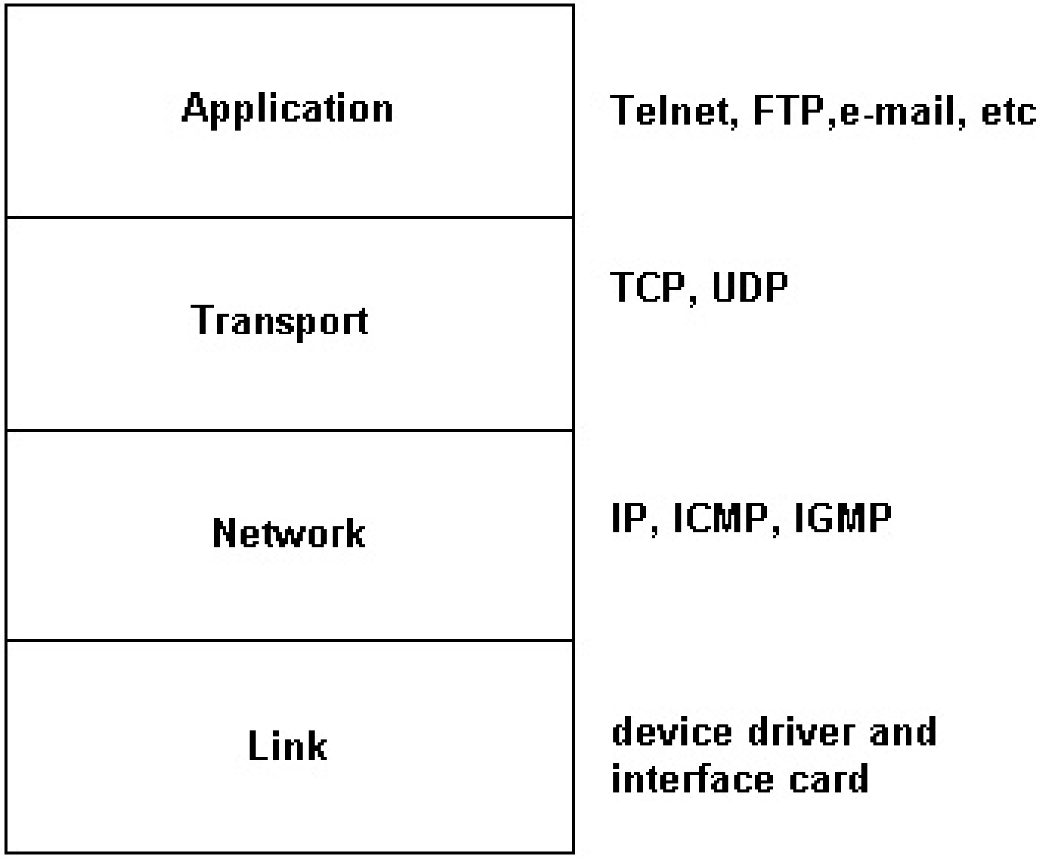
Blackhole Diagram



ARP Spoofing

- Layer two attack
- Bind IP address of intended victim to MAC address of honeypot.
- All systems on network (including router) send IP packets of non-existent system to honeypot.
  - Arpd process
  - ARP proxy

Four Layer Model



ARP Table

```
router (192.168.1.254) at 08:00:20:F8:2A:D2 [ether] on eth0
otto (192.168.1.9) at 00:50:04:67:6A:6C [ether] on eth0
itchy (192.168.1.100) at 00:50:DA:D8:7B:1C [ether] on eth0
scratchy (192.168.1.8) at 00:10:4B:70:14:E7 [ether] on eth0
apu (192.168.1.20) at 00:40:96:48:A9:54 [ether] on eth0
```

ARP Request

```
04/18-14:26:23.193451 ARP who-has 192.168.1.200 tell
192.168.1.10

04/18-14:26:23.193633 ARP reply 192.168.1.200 is-at
0:10:4B:70:14:E7
```

Arpd Process

- Written by Dug Song
- Monitors connections to non-existent systems
- Confirms system does not exist by sending ARP request
- Once confirmed, ARP spoofing.

ARP Proxy

```
arp -s 192.168.1.201 00:10:4B:70:14:E7 permanent pub
arp -s 192.168.1.202 00:10:4B:70:14:E7 permanent pub
arp -s 192.168.1.203 00:10:4B:70:14:E7 permanent pub
arp -s 192.168.1.204 00:10:4B:70:14:E7 permanent pub
arp -s 192.168.1.205 00:10:4B:70:14:E7 permanent pub
```

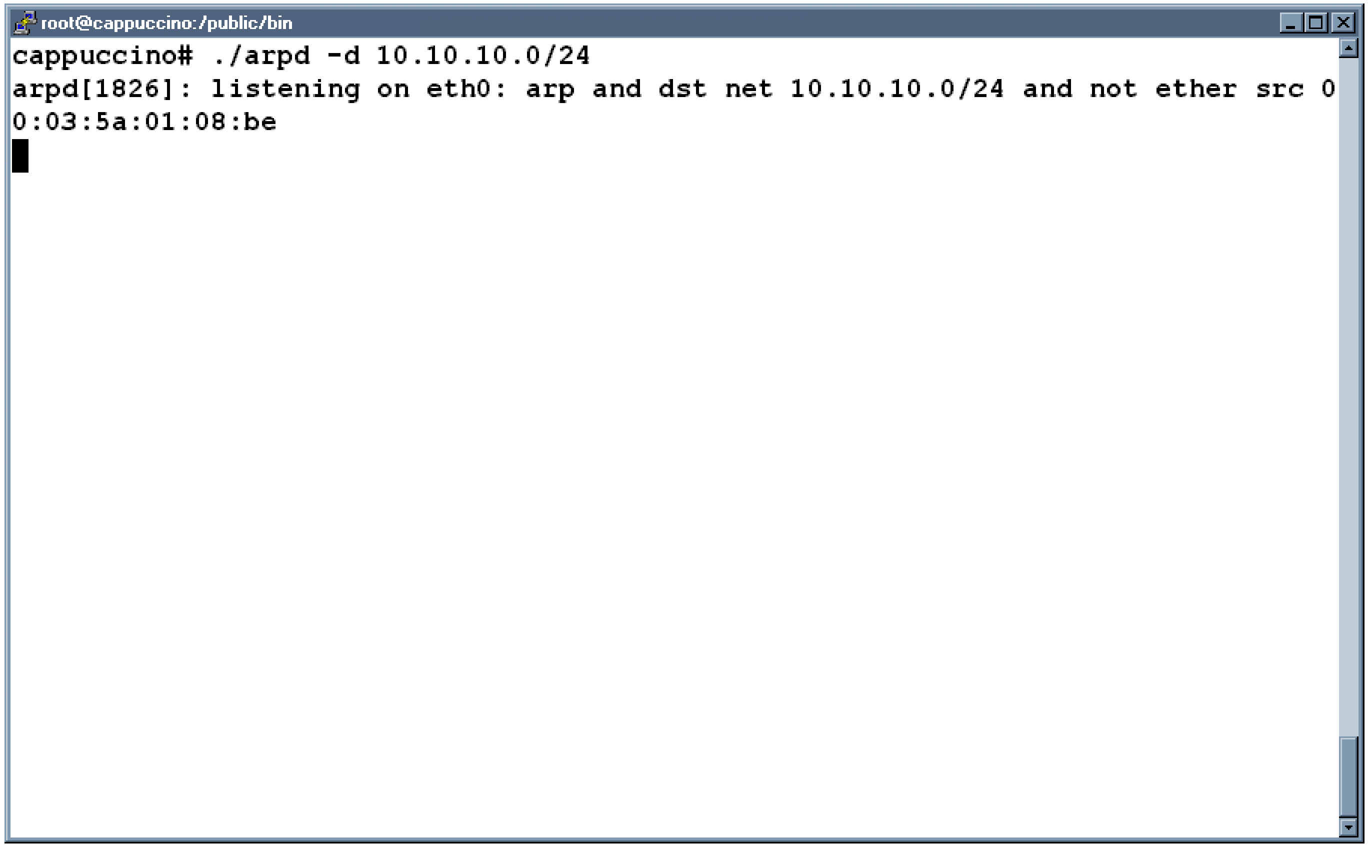
Installing and Configuring

- Compile and install Honeyd on system.
- Compile and install Arpd on system.

Config Files

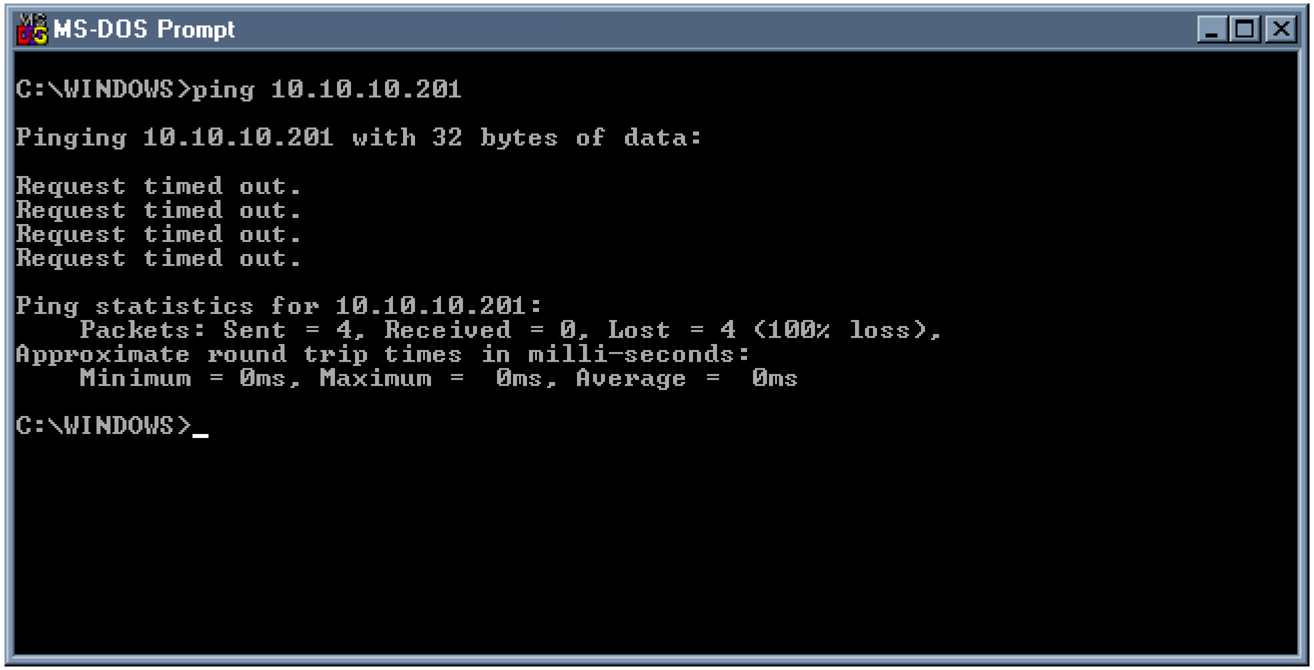
- nmap.prints
- honeyd.conf

Starting arpd



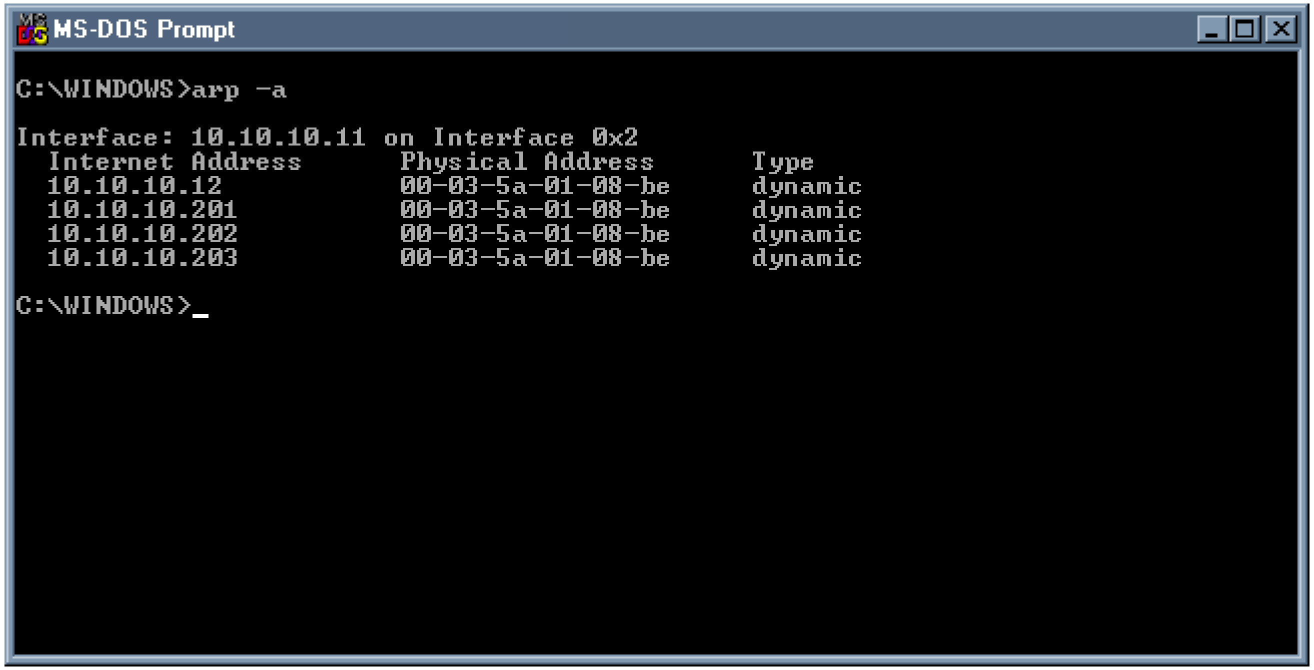
```
cappuccino# ./arpd -d 10.10.10.0/24
arpd[1826]: listening on eth0: arp and dst net 10.10.10.0/24 and not ether src 0
0:03:5a:01:08:be
```

Pinging an unused IP



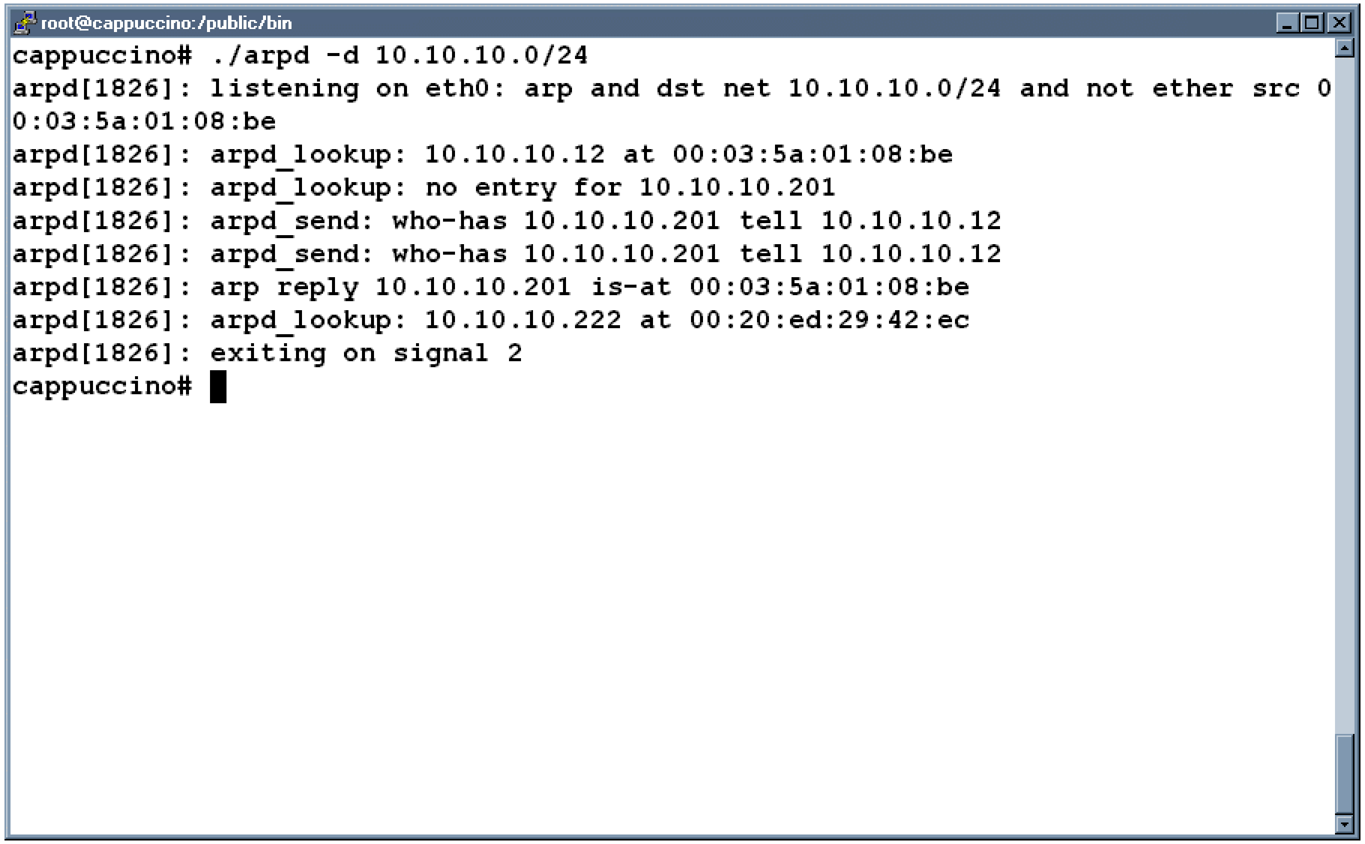
```
C:\WINDOWS>ping 10.10.10.201

Pinging 10.10.10.201 with 32 bytes of data:

Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 10.10.10.201:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum =  0ms, Average =  0ms

C:\WINDOWS>_
```

Arp table entries


MS-DOS Prompt

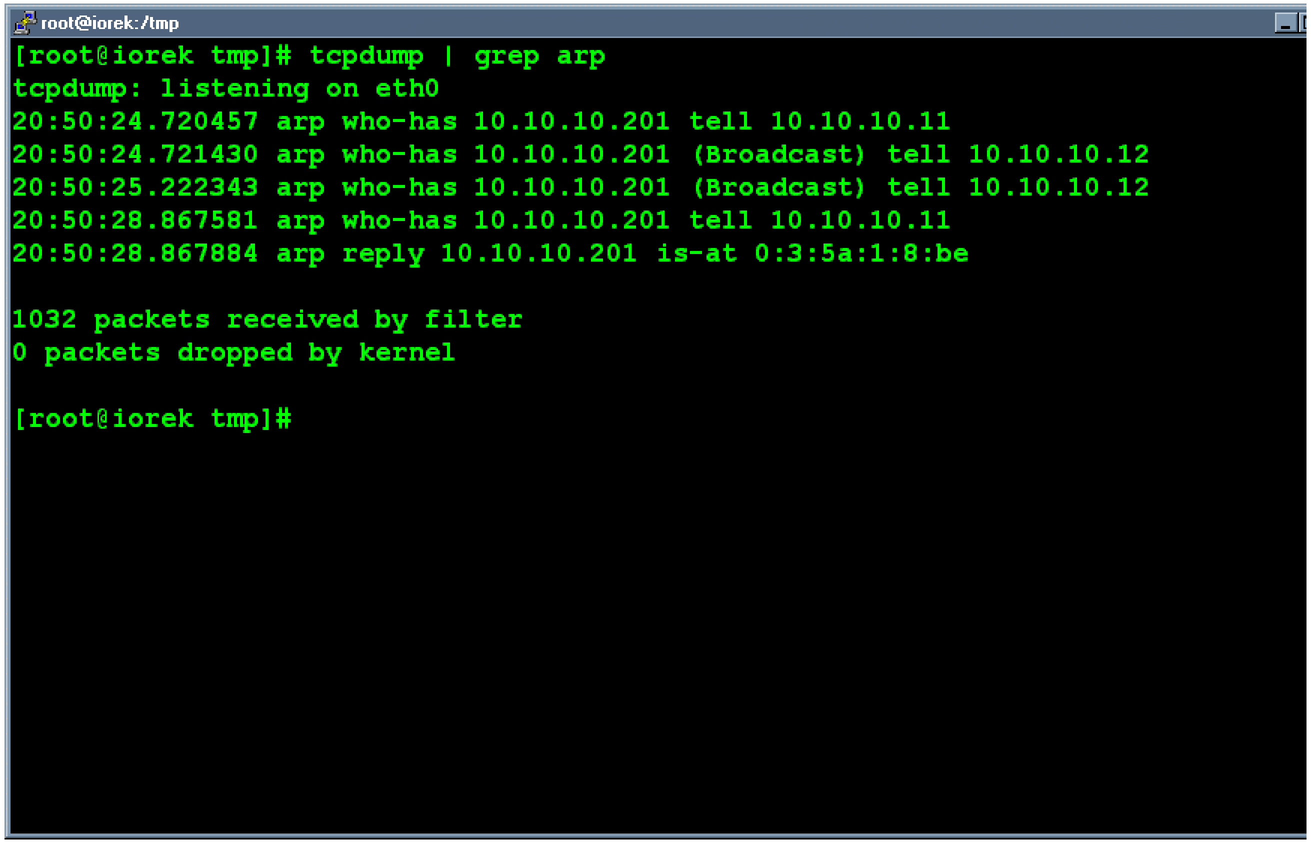
```
C:\WINDOWS>arp -a

Interface: 10.10.10.11 on Interface 0x2
  Internet Address      Physical Address      Type
  10.10.10.12           00-03-5a-01-08-be     dynamic
  10.10.10.201          00-03-5a-01-08-be     dynamic
  10.10.10.202          00-03-5a-01-08-be     dynamic
  10.10.10.203          00-03-5a-01-08-be     dynamic

C:\WINDOWS>_
```

Arpd grabs unused IP



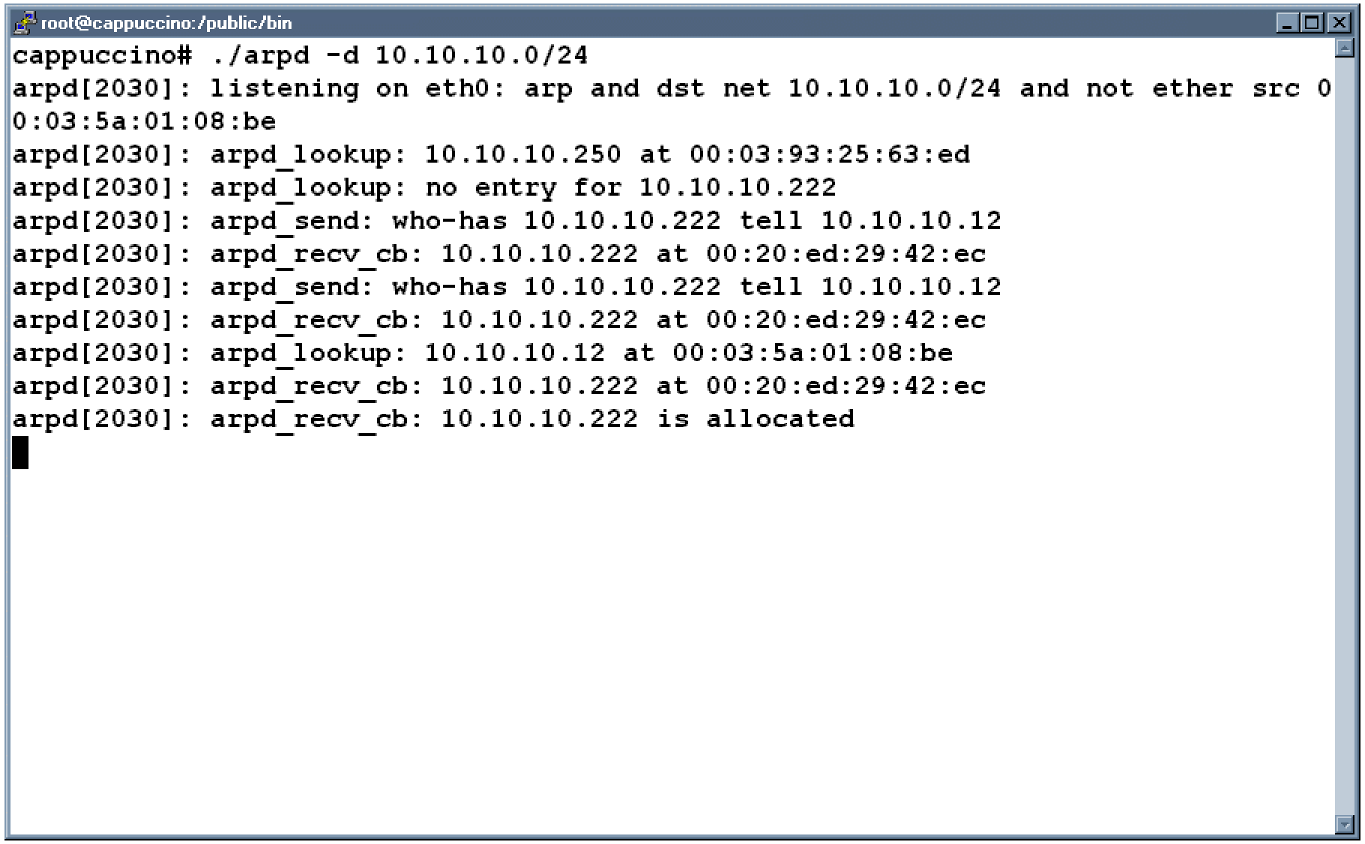
```
cappuccino# ./arpd -d 10.10.10.0/24
arpd[1826]: listening on eth0: arp and dst net 10.10.10.0/24 and not ether src 0
0:03:5a:01:08:be
arpd[1826]: arpd_lookup: 10.10.10.12 at 00:03:5a:01:08:be
arpd[1826]: arpd_lookup: no entry for 10.10.10.201
arpd[1826]: arpd_send: who-has 10.10.10.201 tell 10.10.10.12
arpd[1826]: arpd_send: who-has 10.10.10.201 tell 10.10.10.12
arpd[1826]: arp reply 10.10.10.201 is-at 00:03:5a:01:08:be
arpd[1826]: arpd_lookup: 10.10.10.222 at 00:20:ed:29:42:ec
arpd[1826]: exiting on signal 2
cappuccino# 
```

Arpd as seen by tcpdump



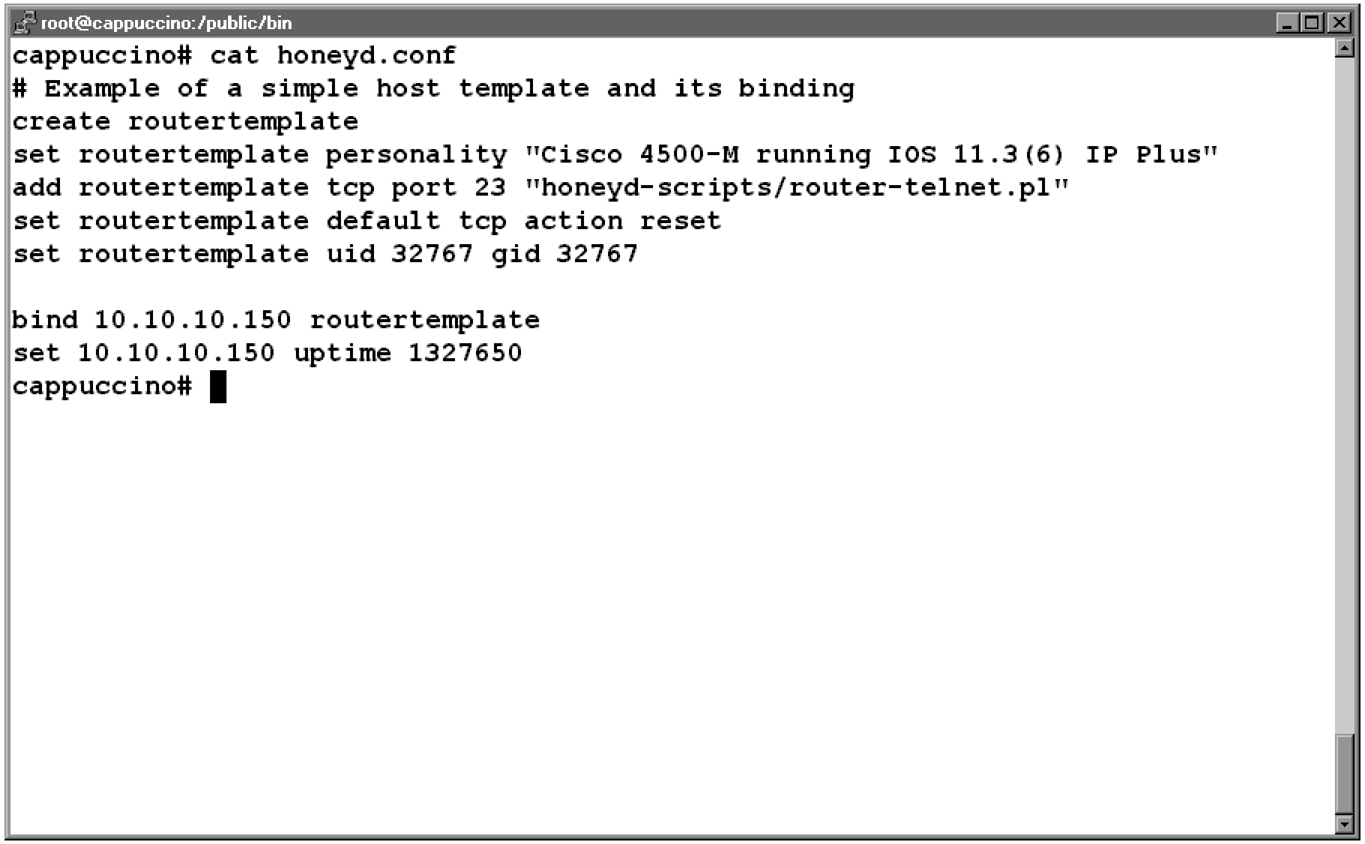
```
[root@iorek tmp]# tcpdump | grep arp
tcpdump: listening on eth0
20:50:24.720457 arp who-has 10.10.10.201 tell 10.10.10.11
20:50:24.721430 arp who-has 10.10.10.201 (Broadcast) tell 10.10.10.12
20:50:25.222343 arp who-has 10.10.10.201 (Broadcast) tell 10.10.10.12
20:50:28.867581 arp who-has 10.10.10.201 tell 10.10.10.11
20:50:28.867884 arp reply 10.10.10.201 is-at 0:3:5a:1:8:be

1032 packets received by filter
0 packets dropped by kernel

[root@iorek tmp]#
```

Arpd learning inuse IPs


root@cappuccino:/public/bin

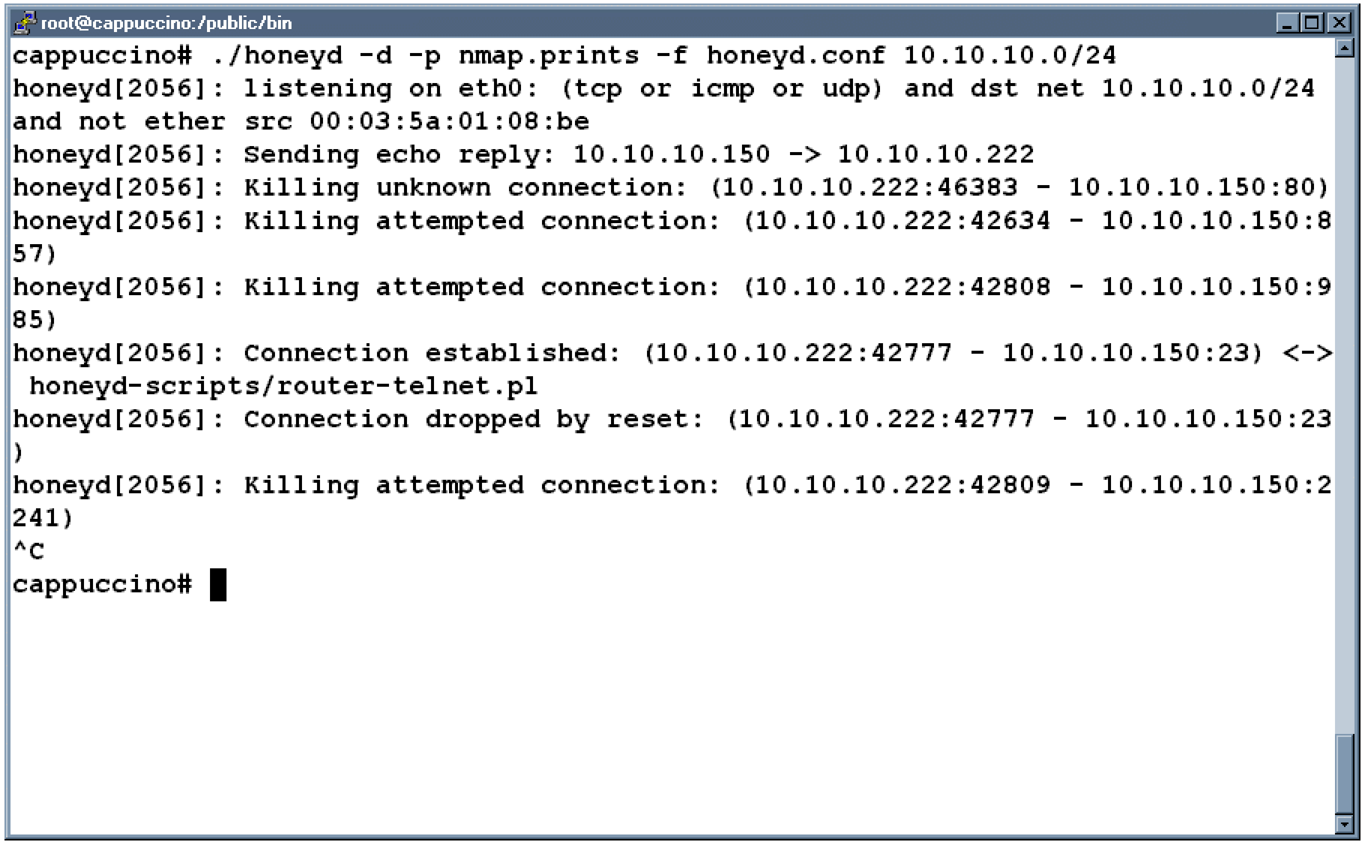
```
cappuccino# ./arpd -d 10.10.10.0/24
arpd[2030]: listening on eth0: arp and dst net 10.10.10.0/24 and not ether src 0
0:03:5a:01:08:be
arpd[2030]: arpd_lookup: 10.10.10.250 at 00:03:93:25:63:ed
arpd[2030]: arpd_lookup: no entry for 10.10.10.222
arpd[2030]: arpd_send: who-has 10.10.10.222 tell 10.10.10.12
arpd[2030]: arpd_recv_cb: 10.10.10.222 at 00:20:ed:29:42:ec
arpd[2030]: arpd_send: who-has 10.10.10.222 tell 10.10.10.12
arpd[2030]: arpd_recv_cb: 10.10.10.222 at 00:20:ed:29:42:ec
arpd[2030]: arpd_lookup: 10.10.10.12 at 00:03:5a:01:08:be
arpd[2030]: arpd_recv_cb: 10.10.10.222 at 00:20:ed:29:42:ec
arpd[2030]: arpd_recv_cb: 10.10.10.222 is allocated
```

A simple honeyd setup



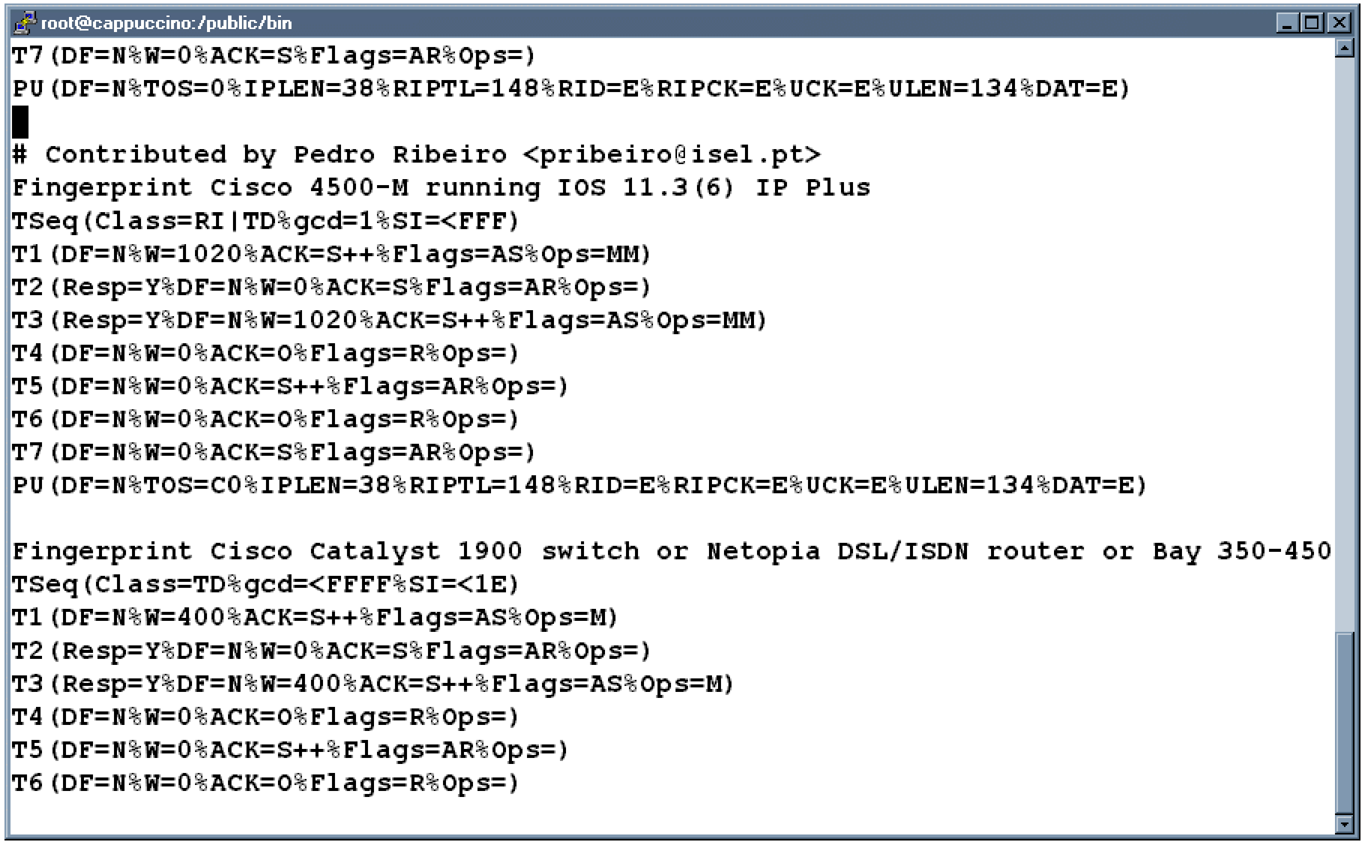
```
cappuccino# cat honeyd.conf
# Example of a simple host template and its binding
create routertemplate
set routertemplate personality "Cisco 4500-M running IOS 11.3(6) IP Plus"
add routertemplate tcp port 23 "honeyd-scripts/router-telnet.pl"
set routertemplate default tcp action reset
set routertemplate uid 32767 gid 32767

bind 10.10.10.150 routertemplate
set 10.10.10.150 uptime 1327650
cappuccino# 
```

honeyd lies!



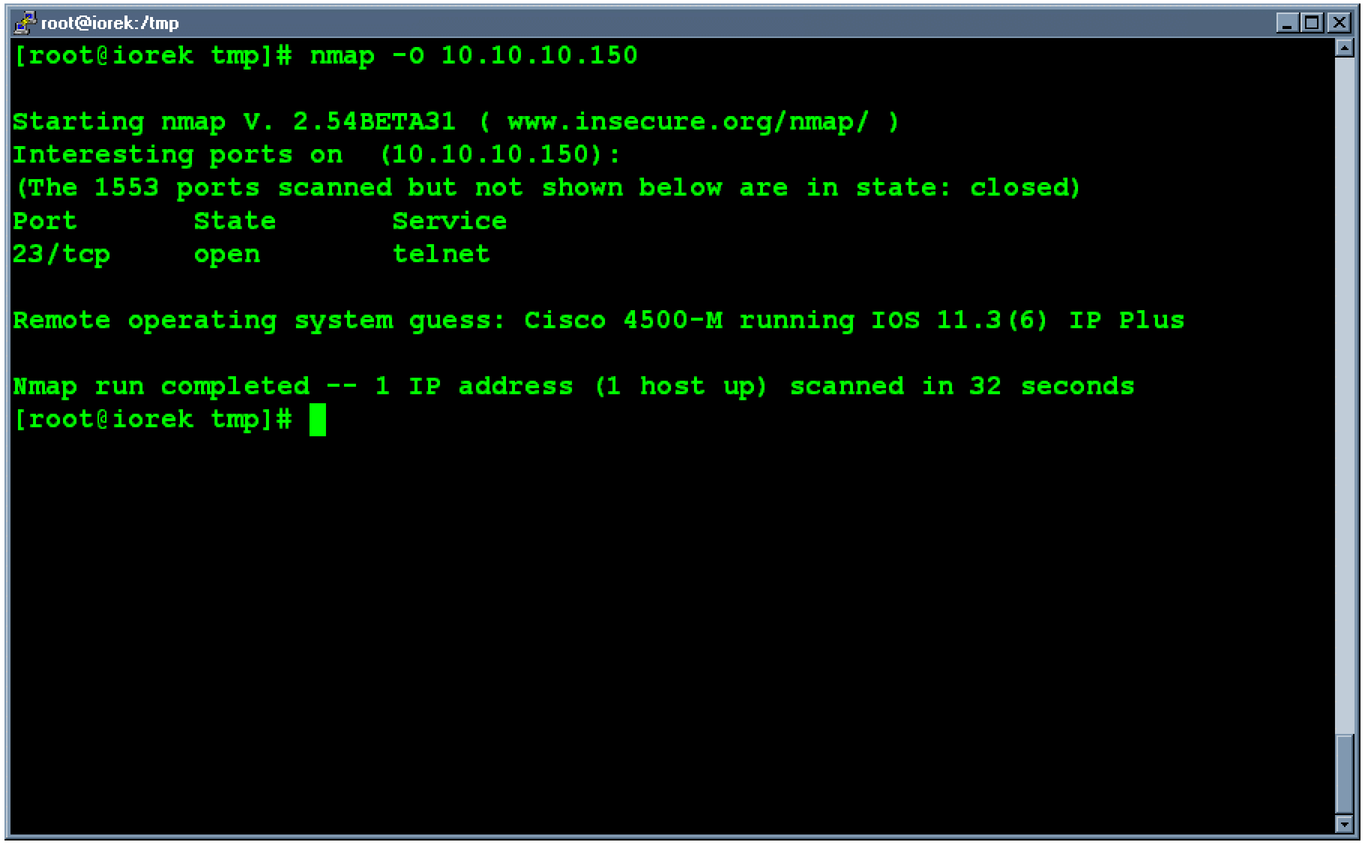
```
cappuccino# ./honeyd -d -p nmap.prints -f honeyd.conf 10.10.10.0/24
honeyd[2056]: listening on eth0: (tcp or icmp or udp) and dst net 10.10.10.0/24
and not ether src 00:03:5a:01:08:be
honeyd[2056]: Sending echo reply: 10.10.10.150 -> 10.10.10.222
honeyd[2056]: Killing unknown connection: (10.10.10.222:46383 - 10.10.10.150:80)
honeyd[2056]: Killing attempted connection: (10.10.10.222:42634 - 10.10.10.150:8
57)
honeyd[2056]: Killing attempted connection: (10.10.10.222:42808 - 10.10.10.150:9
85)
honeyd[2056]: Connection established: (10.10.10.222:42777 - 10.10.10.150:23) <->
 honeyd-scripts/router-telnet.pl
honeyd[2056]: Connection dropped by reset: (10.10.10.222:42777 - 10.10.10.150:23
)
honeyd[2056]: Killing attempted connection: (10.10.10.222:42809 - 10.10.10.150:2
241)
^C
cappuccino# █
```

Honeyd uses nmap sig. file


root@cappuccino:/public/bin

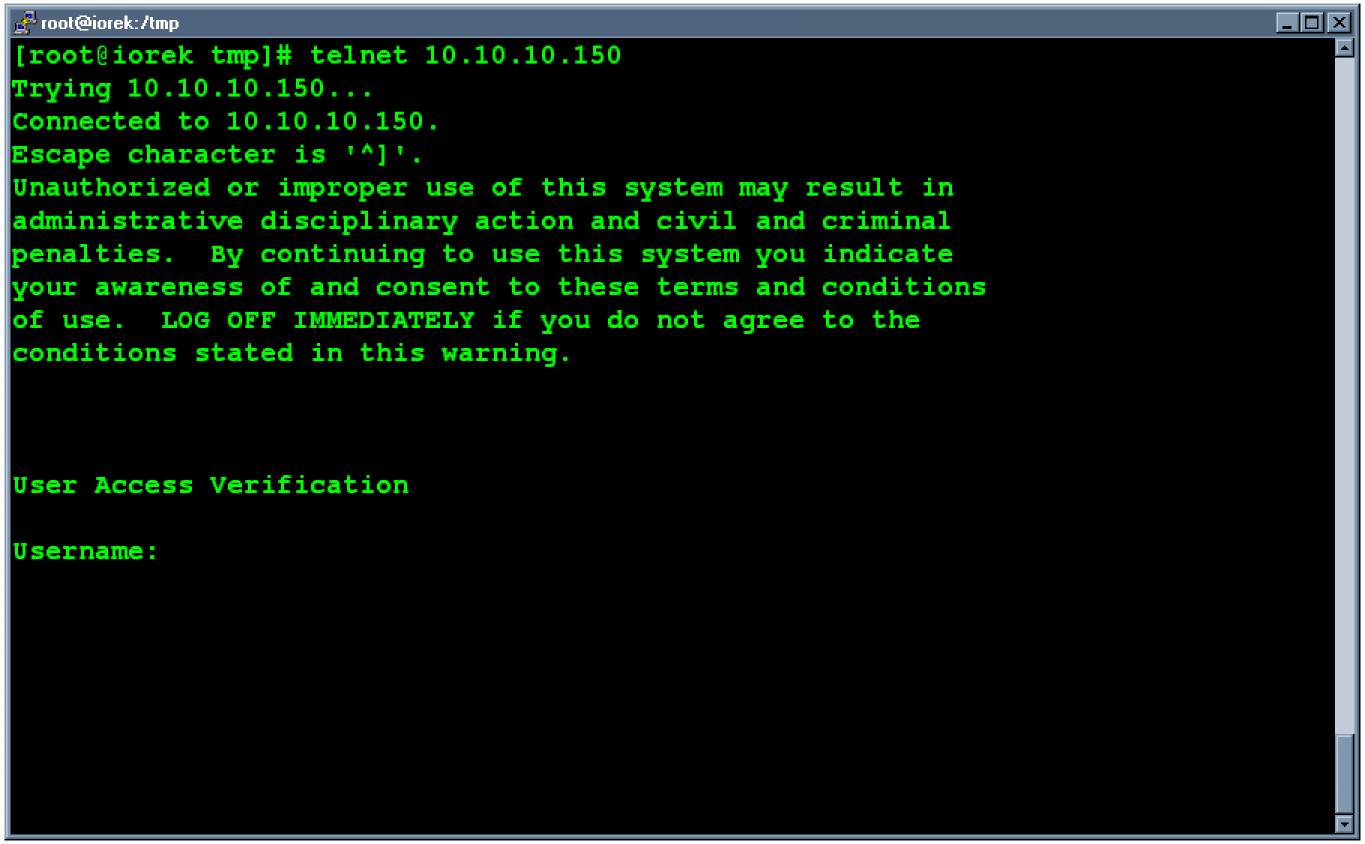
```
T7(DF=N%W=0%ACK=S%Flags=AR%Ops=)
PU(DF=N%TOS=0%IPLEN=38%RIPTL=148%RID=E%RIPCK=E%UCK=E%ULEN=134%DAT=E)

# Contributed by Pedro Ribeiro <pribeiro@isel.pt>
Fingerprint Cisco 4500-M running IOS 11.3(6) IP Plus
TSeq(Class=RI|TD%gcd=1%SI=<FFF)
T1(DF=N%W=1020%ACK=S++%Flags=AS%Ops=MM)
T2(Resp=Y%DF=N%W=0%ACK=S%Flags=AR%Ops=)
T3(Resp=Y%DF=N%W=1020%ACK=S++%Flags=AS%Ops=MM)
T4(DF=N%W=0%ACK=O%Flags=R%Ops=)
T5(DF=N%W=0%ACK=S++%Flags=AR%Ops=)
T6(DF=N%W=0%ACK=O%Flags=R%Ops=)
T7(DF=N%W=0%ACK=S%Flags=AR%Ops=)
PU(DF=N%TOS=C0%IPLEN=38%RIPTL=148%RID=E%RIPCK=E%UCK=E%ULEN=134%DAT=E)

Fingerprint Cisco Catalyst 1900 switch or Netopia DSL/ISDN router or Bay 350-450
TSeq(Class=TD%gcd=<FFFF%SI=<1E)
T1(DF=N%W=400%ACK=S++%Flags=AS%Ops=M)
T2(Resp=Y%DF=N%W=0%ACK=S%Flags=AR%Ops=)
T3(Resp=Y%DF=N%W=400%ACK=S++%Flags=AS%Ops=M)
T4(DF=N%W=0%ACK=O%Flags=R%Ops=)
T5(DF=N%W=0%ACK=S++%Flags=AR%Ops=)
T6(DF=N%W=0%ACK=O%Flags=R%Ops=)
```

What nmap sees



```
[root@iorek tmp]# nmap -O 10.10.10.150

Starting nmap V. 2.54BETA31 ( www.insecure.org/nmap/ )
Interesting ports on  (10.10.10.150):
(The 1553 ports scanned but not shown below are in state: closed)
Port       State       Service
23/tcp     open        telnet

Remote operating system guess: Cisco 4500-M running IOS 11.3(6) IP Plus

Nmap run completed -- 1 IP address (1 host up) scanned in 32 seconds
[root@iorek tmp]#
```

Telnet results



```
[root@iorek tmp]# telnet 10.10.10.150
Trying 10.10.10.150...
Connected to 10.10.10.150.
Escape character is '^]'.
Unauthorized or improper use of this system may result in
administrative disciplinary action and civil and criminal
penalties.  By continuing to use this system you indicate
your awareness of and consent to these terms and conditions
of use.  LOG OFF IMMEDIATELY if you do not agree to the
conditions stated in this warning.



User Access Verification

Username:
```

honeyd.conf

```
create default
set default personality "FreeBSD 2.2.1-STABLE"
set default default tcp action reset
add default tcp port 80 "sh /etc/honeyd/scripts/web.sh"
add default tcp port 22 "sh /etc/honeyd/scripts/test.sh"
add default tcp port 113 open
add default tcp port 1 reset

create windows
set windows personality "Windows NT 4.0 Server SP5-SP6"
set windows default tcp action reset
add windows tcp port 80 ”perl /etc/honeyd/scripts/iis/main.pl"
add windows tcp port 25 block
add windows tcp port 23 proxy real-server.tracking-hackers.com:23
add windows tcp port 22 proxy $ipsrc:22
set windows uptime 3284460

bind 192.168.1.200 windows
```

Templates

- Used to describe the behaviour of a honeypot.
- Bind templates to specific destination IP addresses.
- Bind general templates that default to everyone else.

```
create windows
```

Personality

- Determines the IP stack behavior of the honeypot.

```
set windows personality "Windows NT 4.0 Server SP5-SP6"
```

Service Behavior

- Reset - Send RST or ICMP unreachable
- Open - Acknowledges Connection
- Block - Does nothing, drops connection
- Script - Executes predefined script.

Default Action

- What the emulated port does when there is no specific script defined.

```
set windows default tcp action reset
set windows default udp action reset
```

Add TCP Port

- Define specific behavior of a port.

```
add windows tcp port 80 ”perl /etc/honeyd/scripts/iis/main.pl"
add windows tcp port 25 block
```

Proxy

- To another system
- To attacker

```
add windows tcp port 23 proxy real-server.tracking-hackers.com:23
add windows tcp port 22 proxy $ipsrc:22
```

Additional Options

- uptime

```
set template uptime 3284460
```

Deploying and Maintaining

- Networks with no live systems
  - Blackholing
- Networks with live systems

Information Gathering

- Snort to capture all content activity (such as keystrokes)
- Syslogd logs connections
- Use service scripts to define more logging

syslogd

```
Apr 22 12:42:04 honeypot honeyd[27614]: Connection
request: (192.168.1.10:1768 - 192.168.1.200:23)

Apr 22 12:42:04 honeypot honeyd[27614]: Connection
established: (192.168.1.10:1768 - 192.168.1.200:23) <->
perl /etc/honeyd/scripts/router-telnet.pl

Apr 22 12:42:09 honeypot honeyd[27614]:
E(192.168.1.10:1768 - 192.168.1.200:23): Attempted
login: lance/cisco

Apr 22 12:42:13 honeypot honeyd[27614]: Connection
dropped with reset:(192.168.1.10:1768 - 92.168.1.200:23)
```

Risk

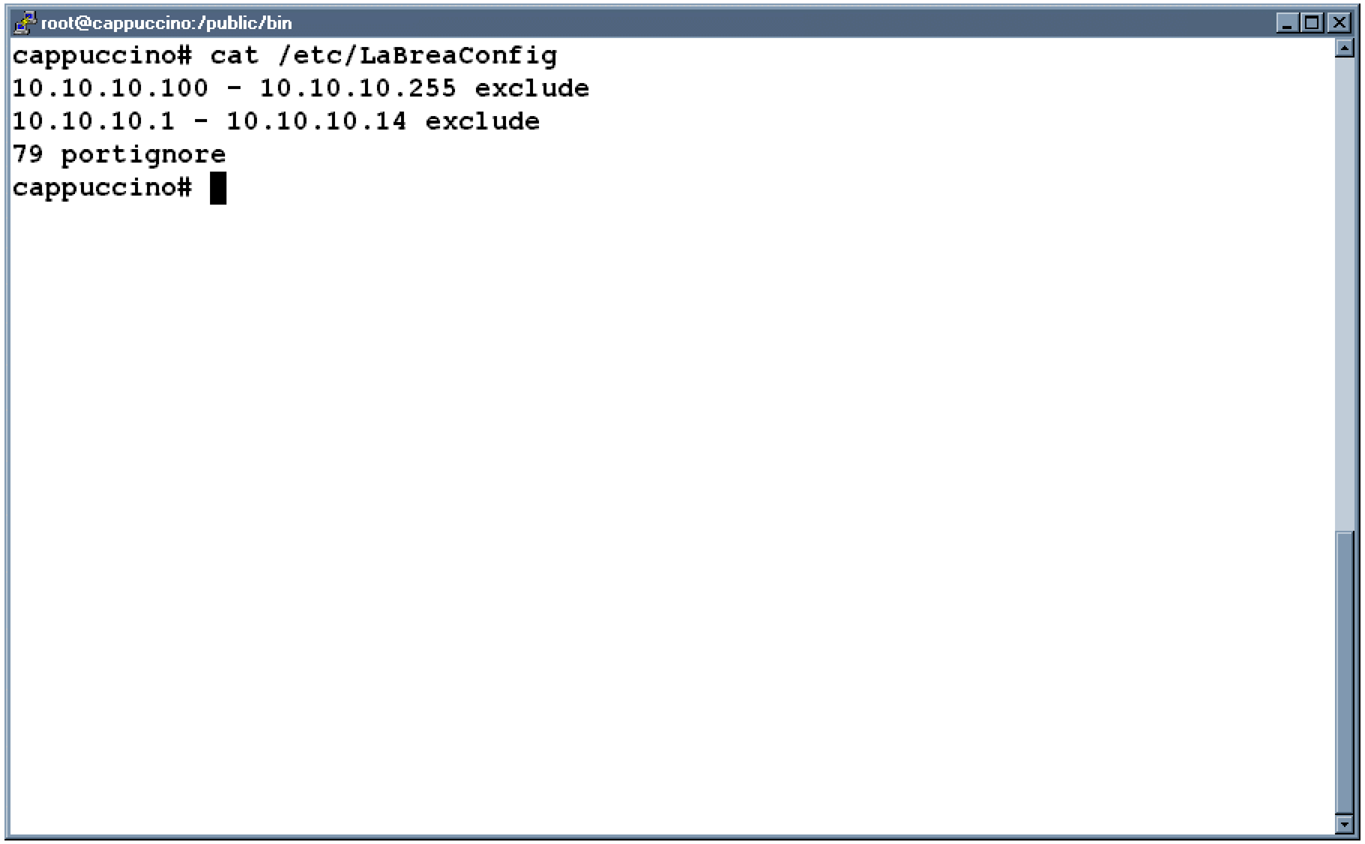
- Emulated services have limited interaction.
- Risk mainly with operating system.
- Arpd may cause problems in some networks.

LaBrea

LaBrea

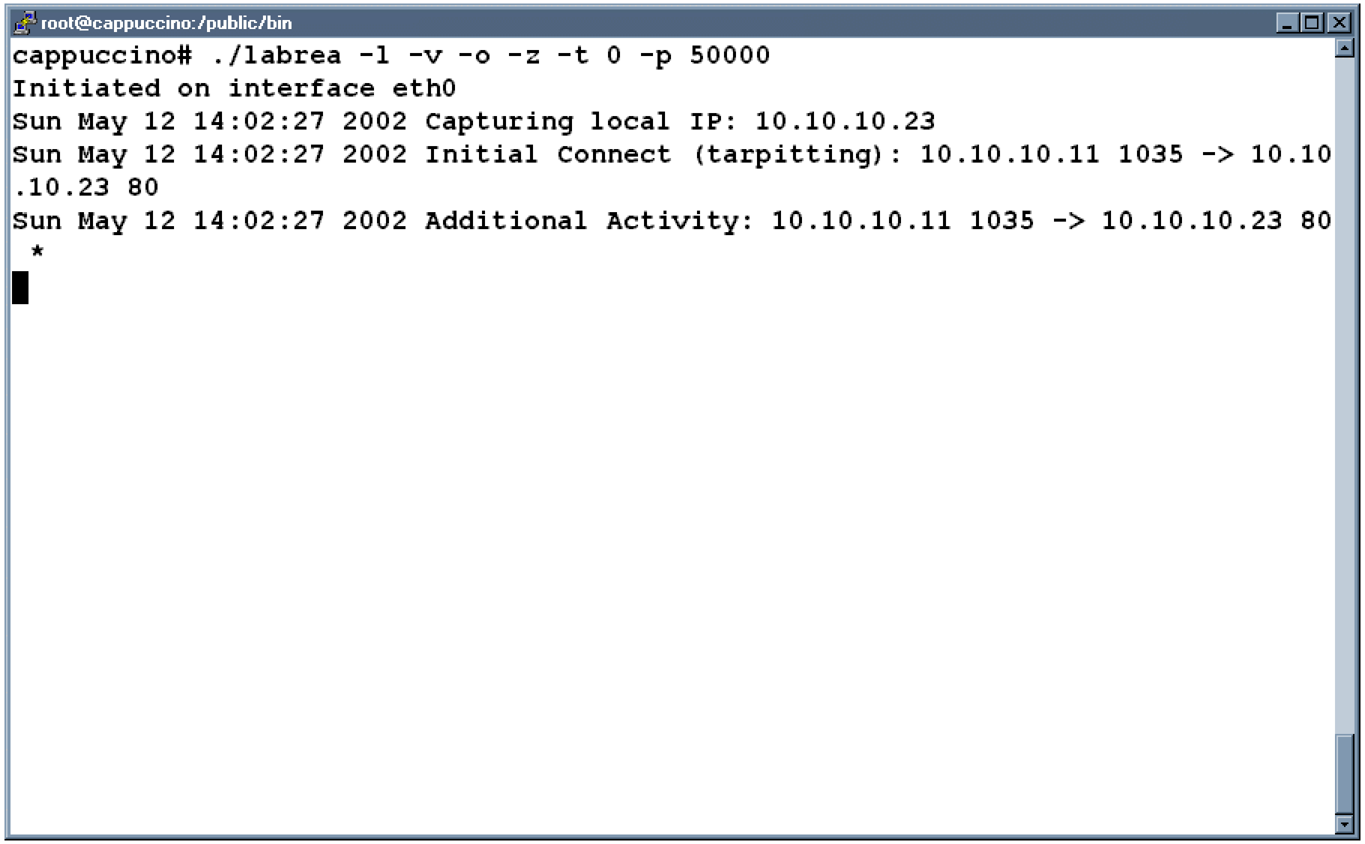
- What is it?
  - A no-interaction honeypot designed to interfere with scanning programs and worms
  - Detects and delays scanners and worms by blackholing their traffic and playing TCP-level games designed to “jam” connections
  - Available from http://www.hackbusters.net
    - UNIX and Windows versions are supported

LaBrea Configuration



```
cappuccino# cat /etc/LaBreaConfig
10.10.10.100 - 10.10.10.255 exclude
10.10.10.1 - 10.10.10.14 exclude
79 portignore
cappuccino#
```

LaBrea Invocation



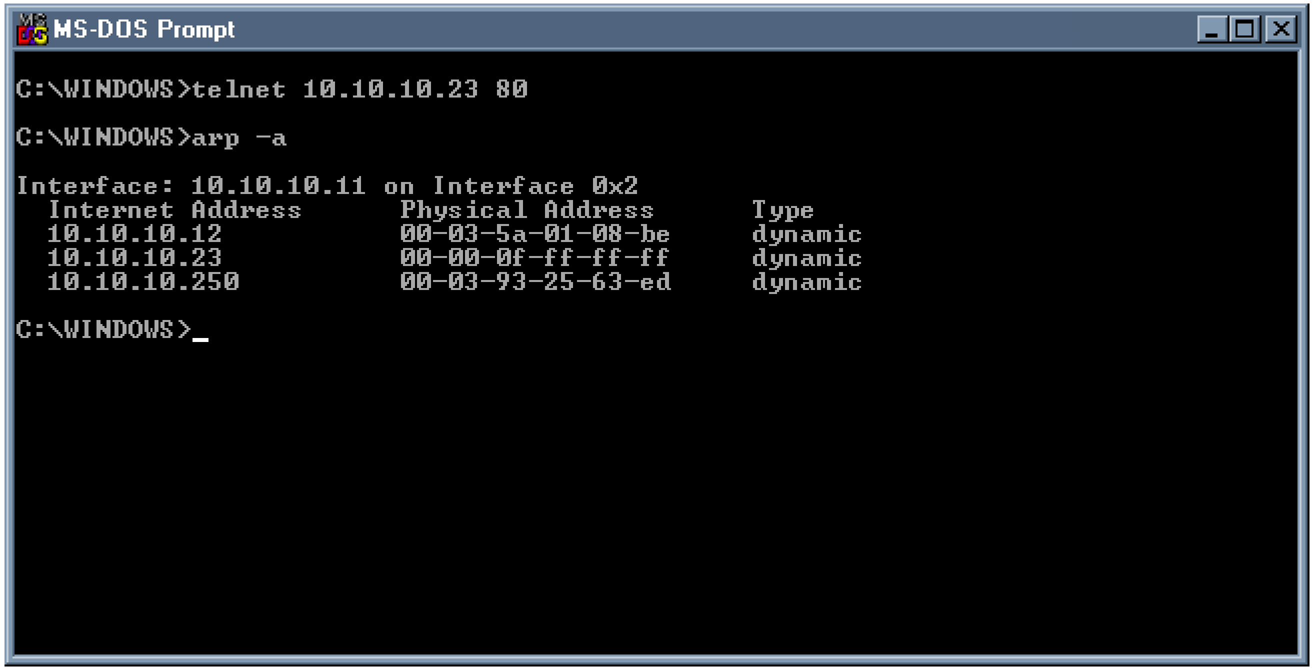
root@cappuccino:/public/bin

```
cappuccino# ./labrea -l -v -o -z -t 0 -p 50000
Initiated on interface eth0
Sun May 12 14:02:27 2002 Capturing local IP: 10.10.10.23
Sun May 12 14:02:27 2002 Initial Connect (tarpitting): 10.10.10.11 1035 -> 10.10
.10.23 80
Sun May 12 14:02:27 2002 Additional Activity: 10.10.10.11 1035 -> 10.10.10.23 80
 *
```

From the Other Side

- From the calling side you see - nothing
  - The connection just hangs
  - A browser or telnet session simply sits there and does nothing
  - In order to prevent this effect worms will have to become interrupt-driven (harder to code) above the TCP level

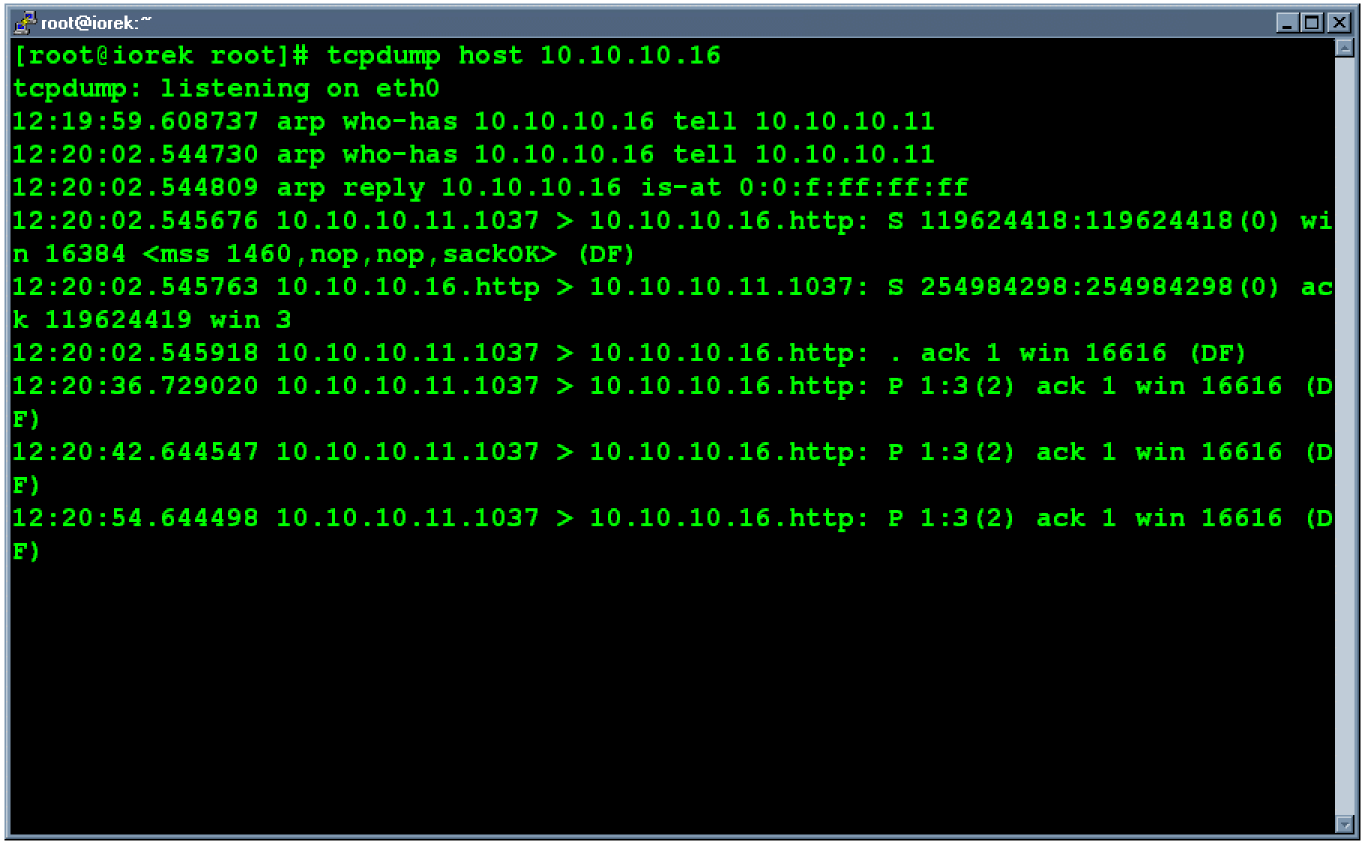
From the Other Side



MS-DOS Prompt

```
C:\WINDOWS>telnet 10.10.10.23 80

C:\WINDOWS>arp -a

Interface: 10.10.10.11 on Interface 0x2
  Internet Address      Physical Address      Type
  10.10.10.12           00-03-5a-01-08-be     dynamic
  10.10.10.23           00-00-0f-ff-ff-ff     dynamic
  10.10.10.250          00-03-93-25-63-ed     dynamic

C:\WINDOWS>_
```

Tcpdump View


root@iorek:~

```
[root@iorek root]# tcpdump host 10.10.10.16
tcpdump: listening on eth0
12:19:59.608737 arp who-has 10.10.10.16 tell 10.10.10.11
12:20:02.544730 arp who-has 10.10.10.16 tell 10.10.10.11
12:20:02.544809 arp reply 10.10.10.16 is-at 0:0:f:ff:ff:ff
12:20:02.545676 10.10.10.11.1037 > 10.10.10.16.http: S 119624418:119624418(0) wi
n 16384 <mss 1460,nop,nop,sackOK> (DF)
12:20:02.545763 10.10.10.16.http > 10.10.10.11.1037: S 254984298:254984298(0) ac
k 119624419 win 3
12:20:02.545918 10.10.10.11.1037 > 10.10.10.16.http: . ack 1 win 16616 (DF)
12:20:36.729020 10.10.10.11.1037 > 10.10.10.16.http: P 1:3(2) ack 1 win 16616 (D
F)
12:20:42.644547 10.10.10.11.1037 > 10.10.10.16.http: P 1:3(2) ack 1 win 16616 (D
F)
12:20:54.644498 10.10.10.11.1037 > 10.10.10.16.http: P 1:3(2) ack 1 win 16616 (D
F)
```

User Mode Linux

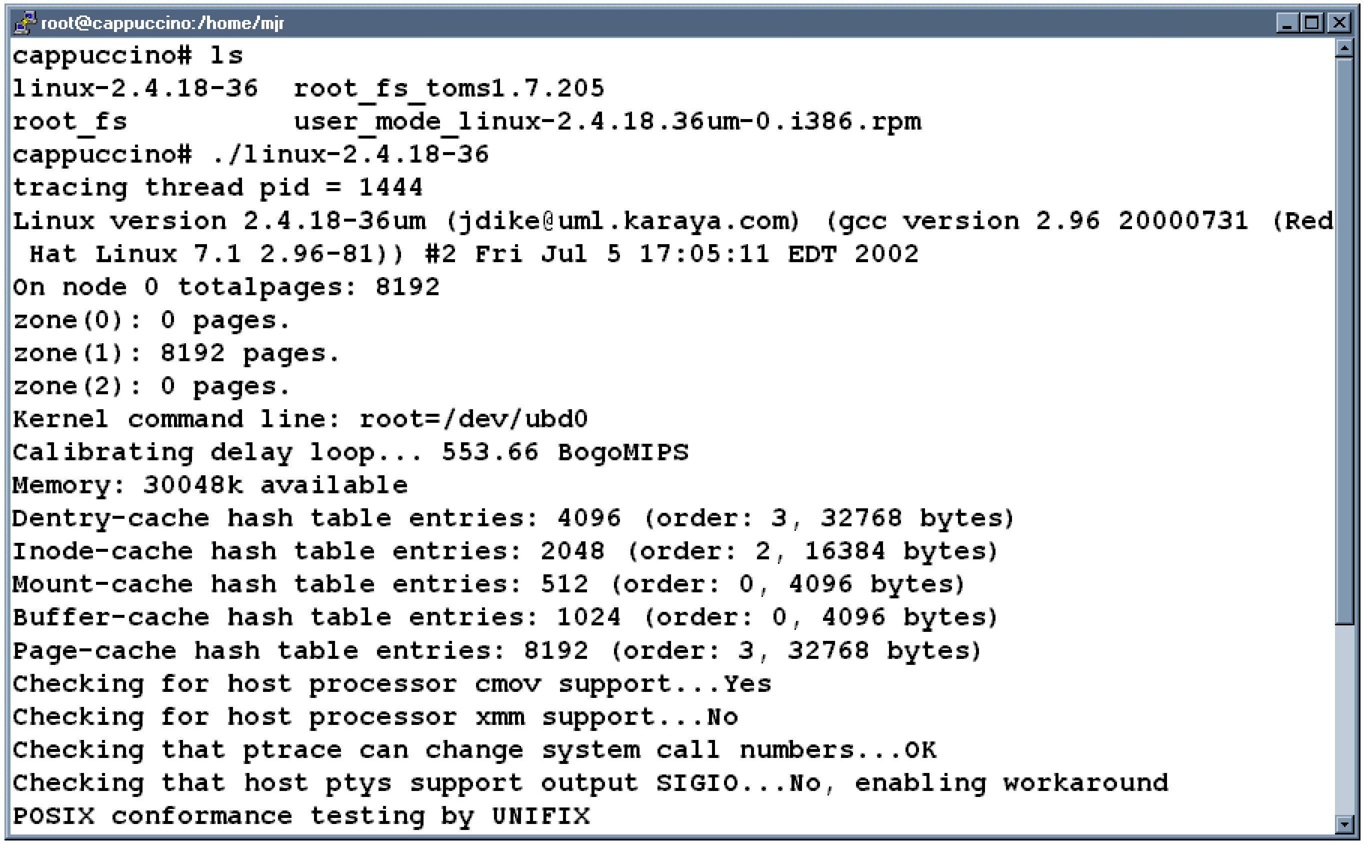
User-Mode Linux

- What is it?
  - A copy of the Linux kernel ported to use the Linux system call interface
    - Disk devices are emulated as files on the host
  - In other words, a kernel running on top of another kernel!
  - You can run multiple kernels on a single machine, each of which has its own unique virtual world-view

User-Mode Linux

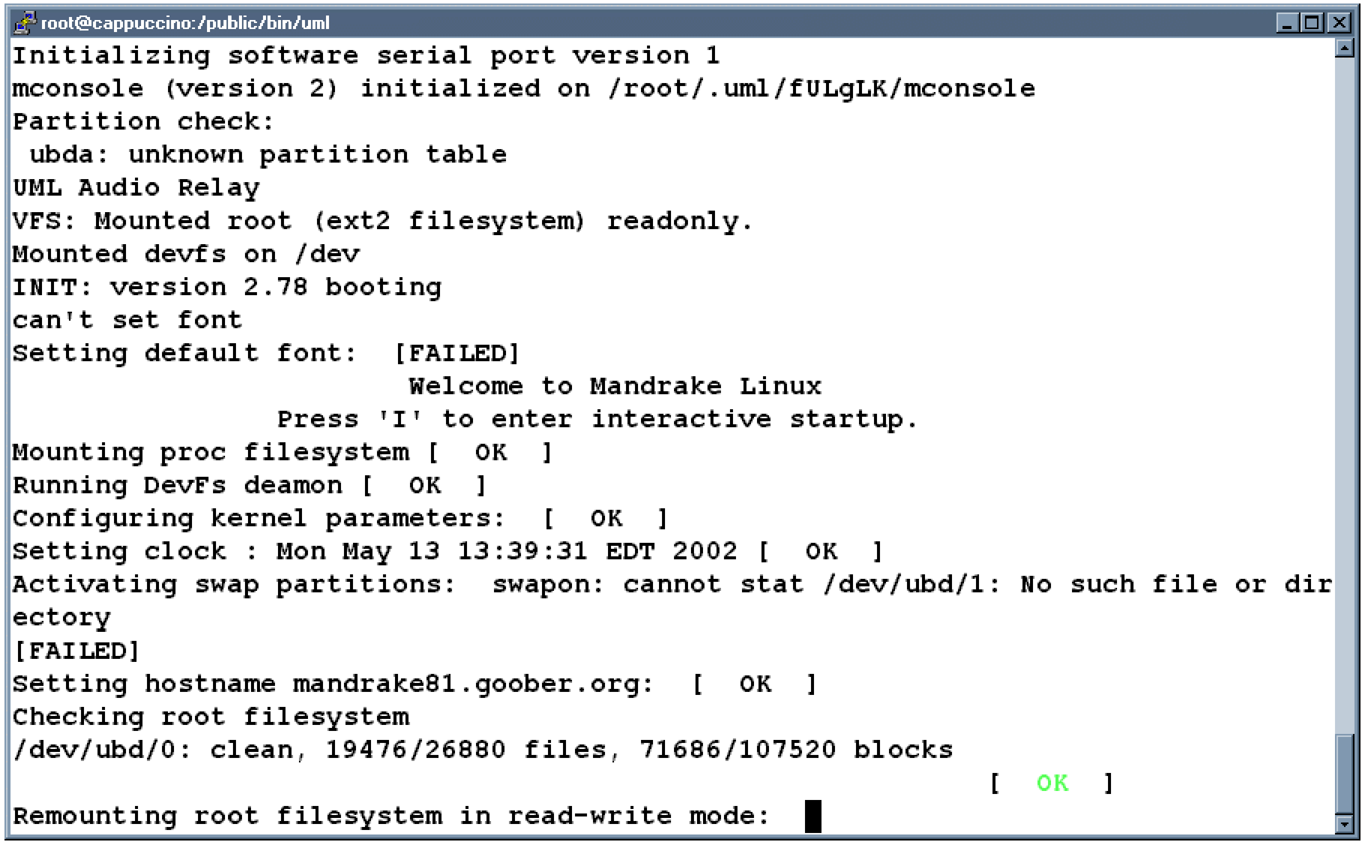
- Where does it come from?
  - http://user-mode-linux.sourceforge.net/
- To install it you need to:
  - Build your own kernel or download an existing kernel
  - Build your own filesystem or download a pre-packaged filesystem image (10 - 100Mb)
    - Filesystems for various configurations available

Booting a UML



```
cappuccino# ls
linux-2.4.18-36   root_fs_toms1.7.205
root_fs           user_mode_linux-2.4.18.36um-0.i386.rpm
cappuccino# ./linux-2.4.18-36
tracing thread pid = 1444
Linux version 2.4.18-36um (jdike@uml.karaya.com) (gcc version 2.96 20000731 (Red
 Hat Linux 7.1 2.96-81)) #2 Fri Jul 5 17:05:11 EDT 2002
On node 0 totalpages: 8192
zone(0): 0 pages.
zone(1): 8192 pages.
zone(2): 0 pages.
Kernel command line: root=/dev/ubd0
Calibrating delay loop... 553.66 BogoMIPS
Memory: 30048k available
Dentry-cache hash table entries: 4096 (order: 3, 32768 bytes)
Inode-cache hash table entries: 2048 (order: 2, 16384 bytes)
Mount-cache hash table entries: 512 (order: 0, 4096 bytes)
Buffer-cache hash table entries: 1024 (order: 0, 4096 bytes)
Page-cache hash table entries: 8192 (order: 3, 32768 bytes)
Checking for host processor cmov support...Yes
Checking for host processor xmm support...No
Checking that ptrace can change system call numbers...OK
Checking that host ptys support output SIGIO...No, enabling workaround
POSIX conformance testing by UNIFIX
```

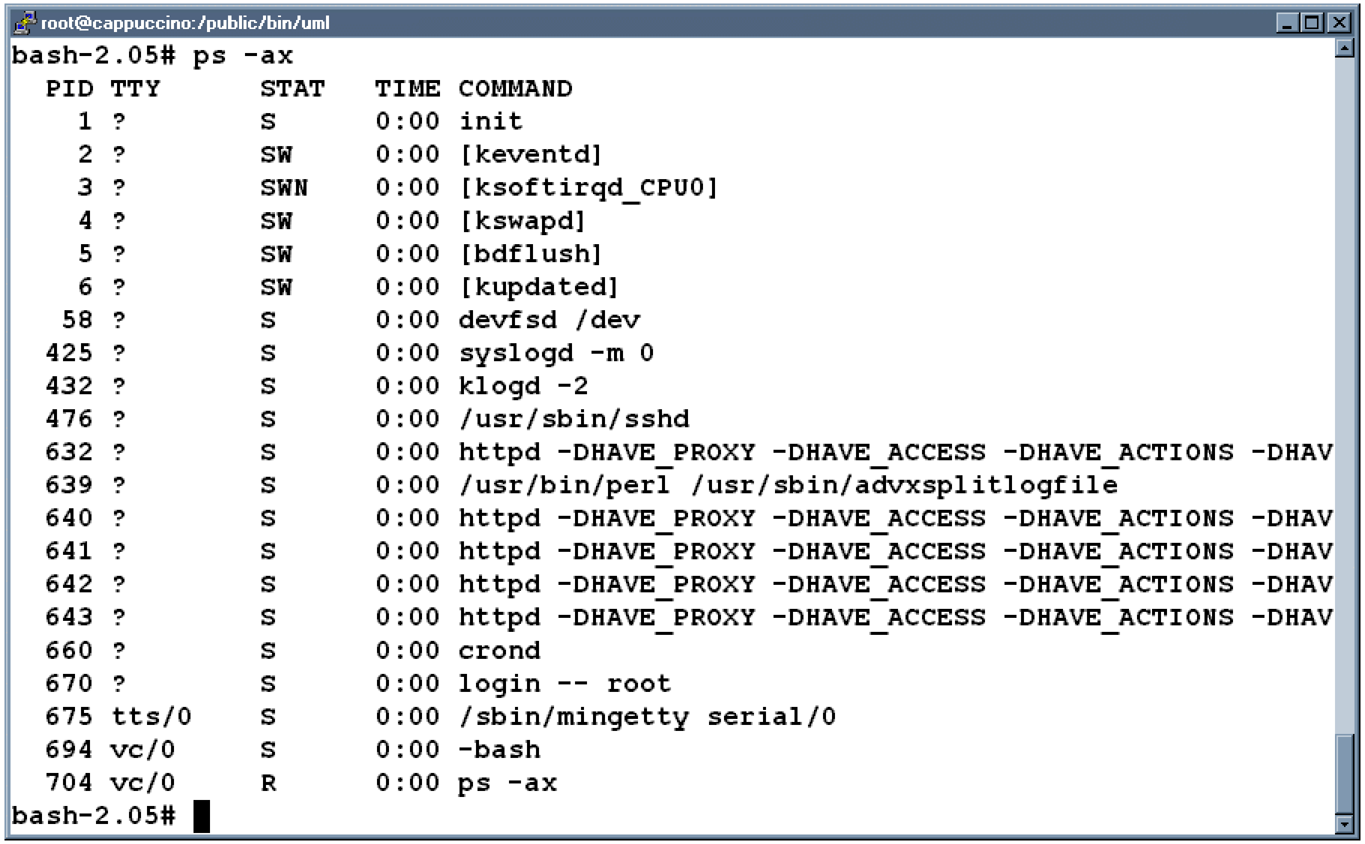
Booting a UML: 2



root@cappuccino:/public/bin/uml

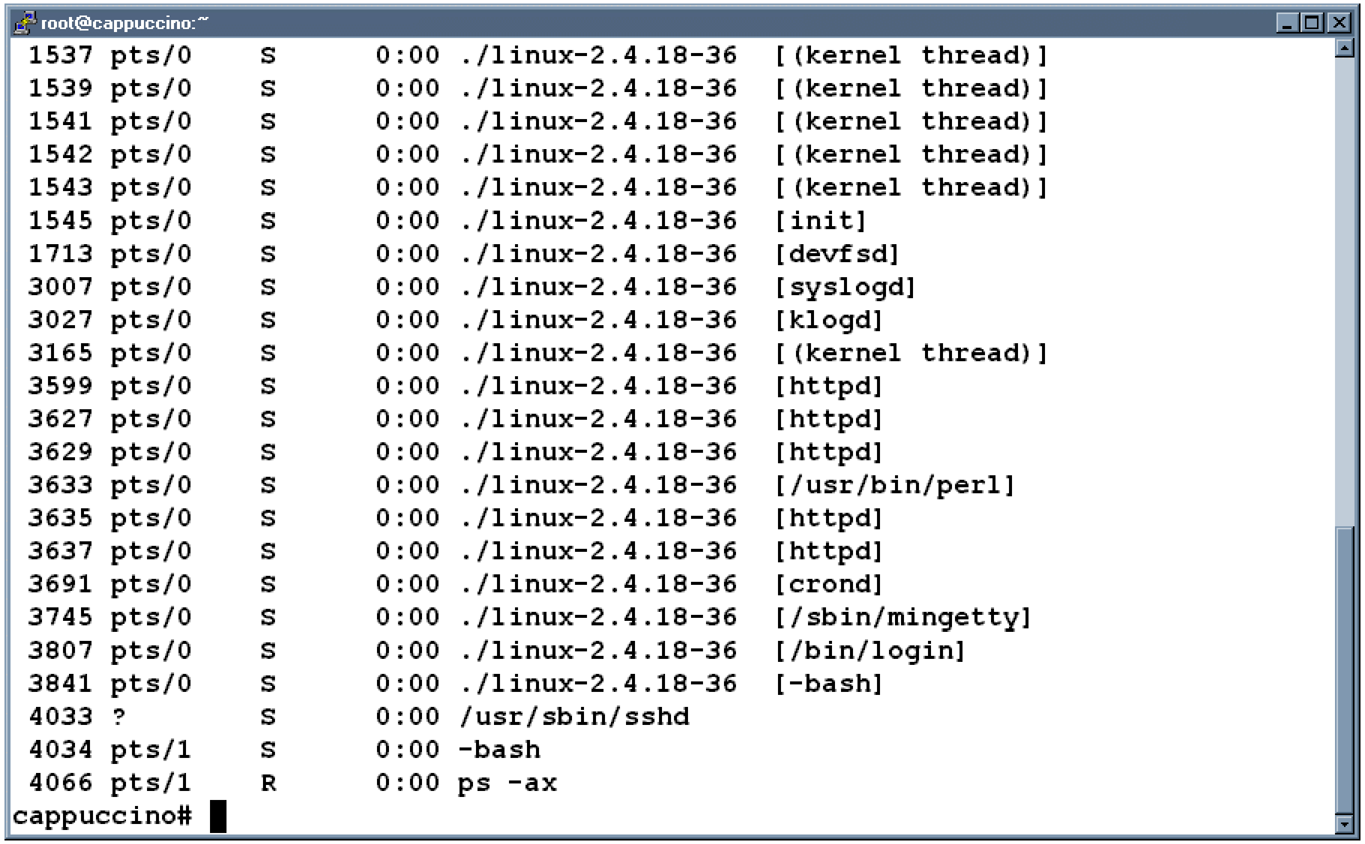
```
Initializing software serial port version 1
mconsole (version 2) initialized on /root/.uml/fULgLK/mconsole
Partition check:
 ubda: unknown partition table
UML Audio Relay
VFS: Mounted root (ext2 filesystem) readonly.
Mounted devfs on /dev
INIT: version 2.78 booting
can't set font
Setting default font:  [FAILED]
                        Welcome to Mandrake Linux
                Press 'I' to enter interactive startup.
Mounting proc filesystem [  OK  ]
Running DevFs deamon [  OK  ]
Configuring kernel parameters:  [  OK  ]
Setting clock : Mon May 13 13:39:31 EDT 2002 [  OK  ]
Activating swap partitions:  swapon: cannot stat /dev/ubd/1: No such file or dir
ectory
[FAILED]
Setting hostname mandrake81.goober.org:  [  OK  ]
Checking root filesystem
/dev/ubd/0: clean, 19476/26880 files, 71686/107520 blocks
                                                           [  OK  ]
Remounting root filesystem in read-write mode: █
```

Within the UML


root@cappuccino:/public/bin/uml

```
bash-2.05# ps -ax
  PID TTY      STAT   TIME COMMAND
    1 ?        S      0:00 init
    2 ?        SW     0:00 [keventd]
    3 ?        SWN    0:00 [ksoftirqd_CPU0]
    4 ?        SW     0:00 [kswapd]
    5 ?        SW     0:00 [bdflush]
    6 ?        SW     0:00 [kupdated]
   58 ?        S      0:00 devfsd /dev
  425 ?        S      0:00 syslogd -m 0
  432 ?        S      0:00 klogd -2
  476 ?        S      0:00 /usr/sbin/sshd
  632 ?        S      0:00 httpd -DHAVE_PROXY -DHAVE_ACCESS -DHAVE_ACTIONS -DHAV
  639 ?        S      0:00 /usr/bin/perl /usr/sbin/advxsplitlogfile
  640 ?        S      0:00 httpd -DHAVE_PROXY -DHAVE_ACCESS -DHAVE_ACTIONS -DHAV
  641 ?        S      0:00 httpd -DHAVE_PROXY -DHAVE_ACCESS -DHAVE_ACTIONS -DHAV
  642 ?        S      0:00 httpd -DHAVE_PROXY -DHAVE_ACCESS -DHAVE_ACTIONS -DHAV
  643 ?        S      0:00 httpd -DHAVE_PROXY -DHAVE_ACCESS -DHAVE_ACTIONS -DHAV
  660 ?        S      0:00 crond
  670 ?        S      0:00 login -- root
  675 tts/0    S      0:00 /sbin/mingetty serial/0
  694 vc/0     S      0:00 -bash
  704 vc/0     R      0:00 ps -ax
bash-2.05#
```

On the UML's host



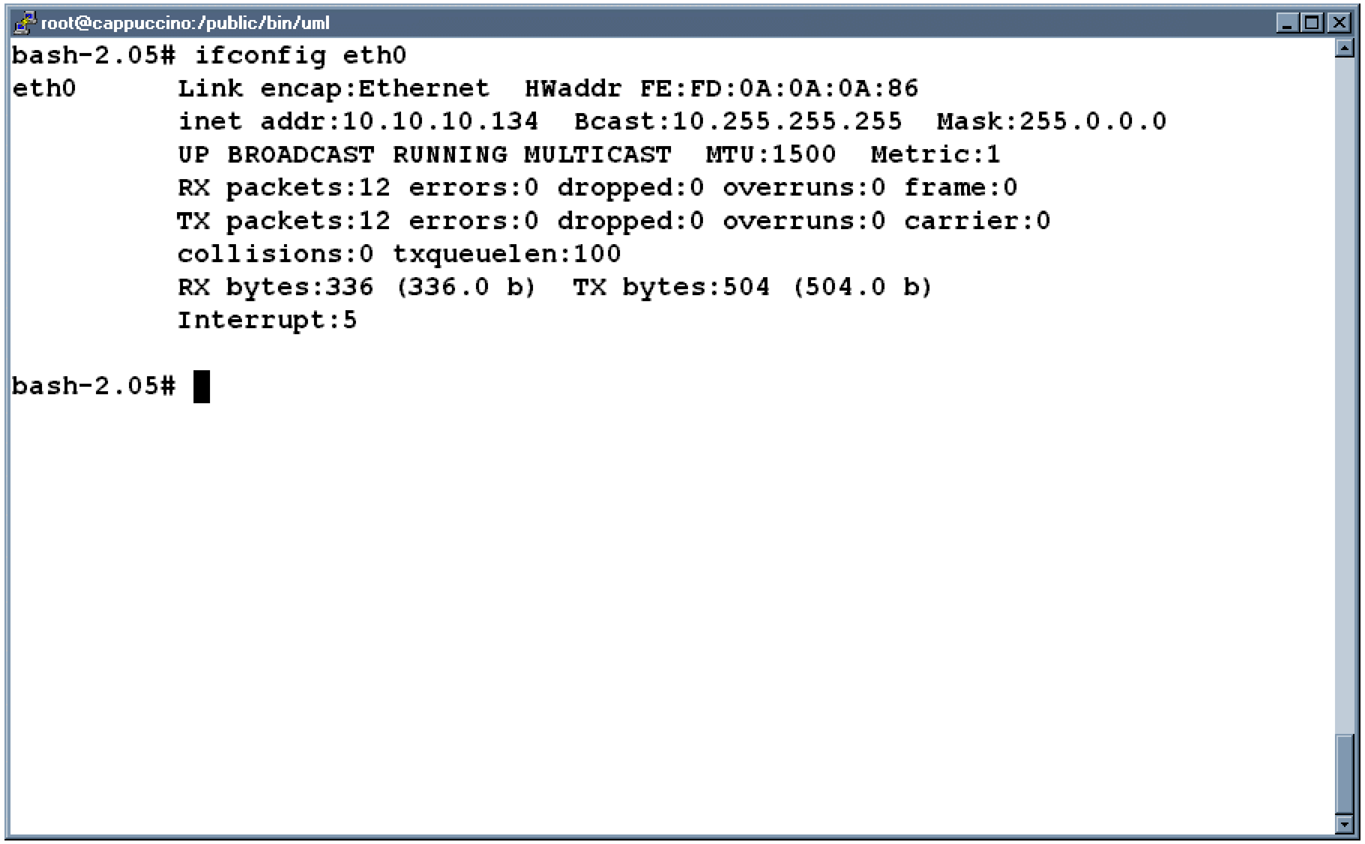
root@cappuccino:~

```
 1537 pts/0    S      0:00 ./linux-2.4.18-36  [(kernel thread)]
 1539 pts/0    S      0:00 ./linux-2.4.18-36  [(kernel thread)]
 1541 pts/0    S      0:00 ./linux-2.4.18-36  [(kernel thread)]
 1542 pts/0    S      0:00 ./linux-2.4.18-36  [(kernel thread)]
 1543 pts/0    S      0:00 ./linux-2.4.18-36  [(kernel thread)]
 1545 pts/0    S      0:00 ./linux-2.4.18-36  [init]
 1713 pts/0    S      0:00 ./linux-2.4.18-36  [devfsd]
 3007 pts/0    S      0:00 ./linux-2.4.18-36  [syslogd]
 3027 pts/0    S      0:00 ./linux-2.4.18-36  [klogd]
 3165 pts/0    S      0:00 ./linux-2.4.18-36  [(kernel thread)]
 3599 pts/0    S      0:00 ./linux-2.4.18-36  [httpd]
 3627 pts/0    S      0:00 ./linux-2.4.18-36  [httpd]
 3629 pts/0    S      0:00 ./linux-2.4.18-36  [httpd]
 3633 pts/0    S      0:00 ./linux-2.4.18-36  [/usr/bin/perl]
 3635 pts/0    S      0:00 ./linux-2.4.18-36  [httpd]
 3637 pts/0    S      0:00 ./linux-2.4.18-36  [httpd]
 3691 pts/0    S      0:00 ./linux-2.4.18-36  [crond]
 3745 pts/0    S      0:00 ./linux-2.4.18-36  [/sbin/mingetty]
 3807 pts/0    S      0:00 ./linux-2.4.18-36  [/bin/login]
 3841 pts/0    S      0:00 ./linux-2.4.18-36  [-bash]
 4033 ?        S      0:00 /usr/sbin/sshd
 4034 pts/1    S      0:00 -bash
 4066 pts/1    R      0:00 ps -ax
cappuccino#
```

Networking on a UML

- Networking is a bit tricky since the UML doesn’t really have an interface to access
  - Preferred method is by using a tunnel/tap, or SLIP driver to route packets down to the host
  - Can now use multicast as well
- You can set the host up as a router into a whole network of hosted UMLs if you want a Linux virtual honeynet

UML's virtual interface



```
bash-2.05# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr FE:FD:0A:0A:0A:86
          inet addr:10.10.10.134  Bcast:10.255.255.255  Mask:255.0.0.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:12 errors:0 dropped:0 overruns:0 frame:0
          TX packets:12 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:336 (336.0 b)  TX bytes:504 (504.0 b)
          Interrupt:5

bash-2.05# 
```

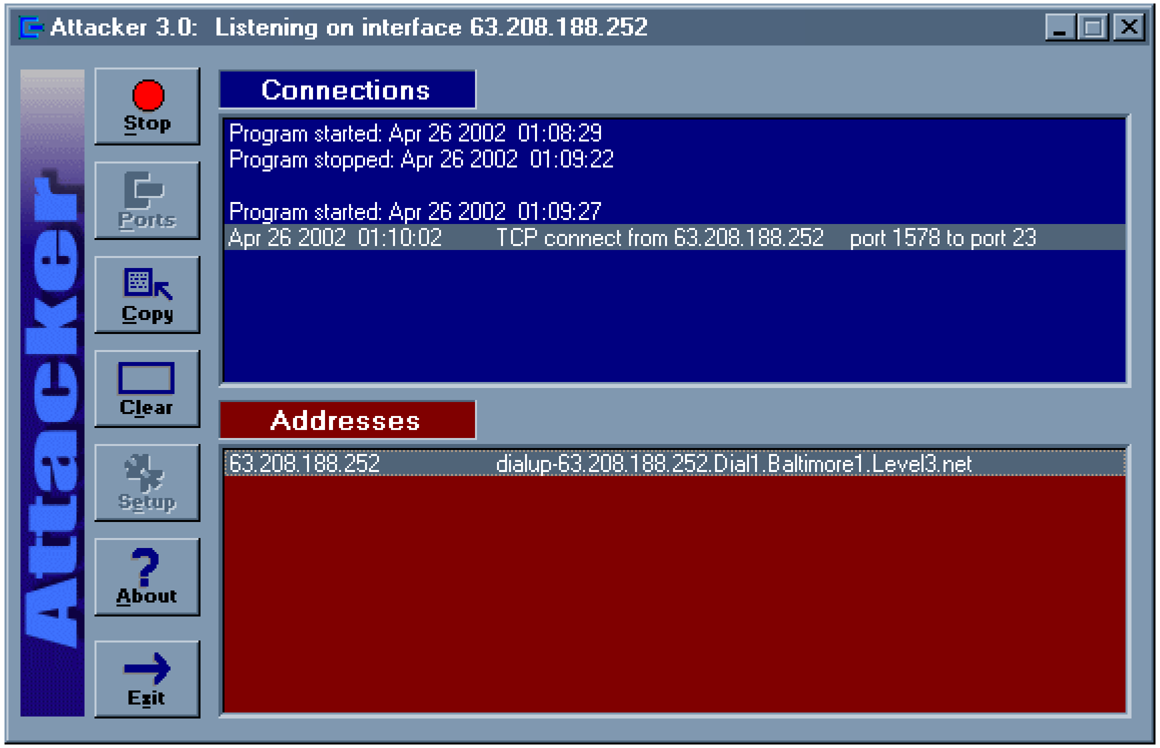
Summary: User Mode Linux

- User-mode Linux is a very powerful tool for experienced Linux system administrators
  - Run a whole cluster of computers within a single physical unit!
  - Expect it will only get better and easier to use over time

Attacker

- What is it?
  - Free port sucker from foundstone
  - Win-32 application, very simple to install
    - Can listen to a large number of UDP and TCP ports
    - Does not emulate services
    - Does not capture traffic - just beeps
  - Listens on external interface (not loopback) - very non-intrusive!

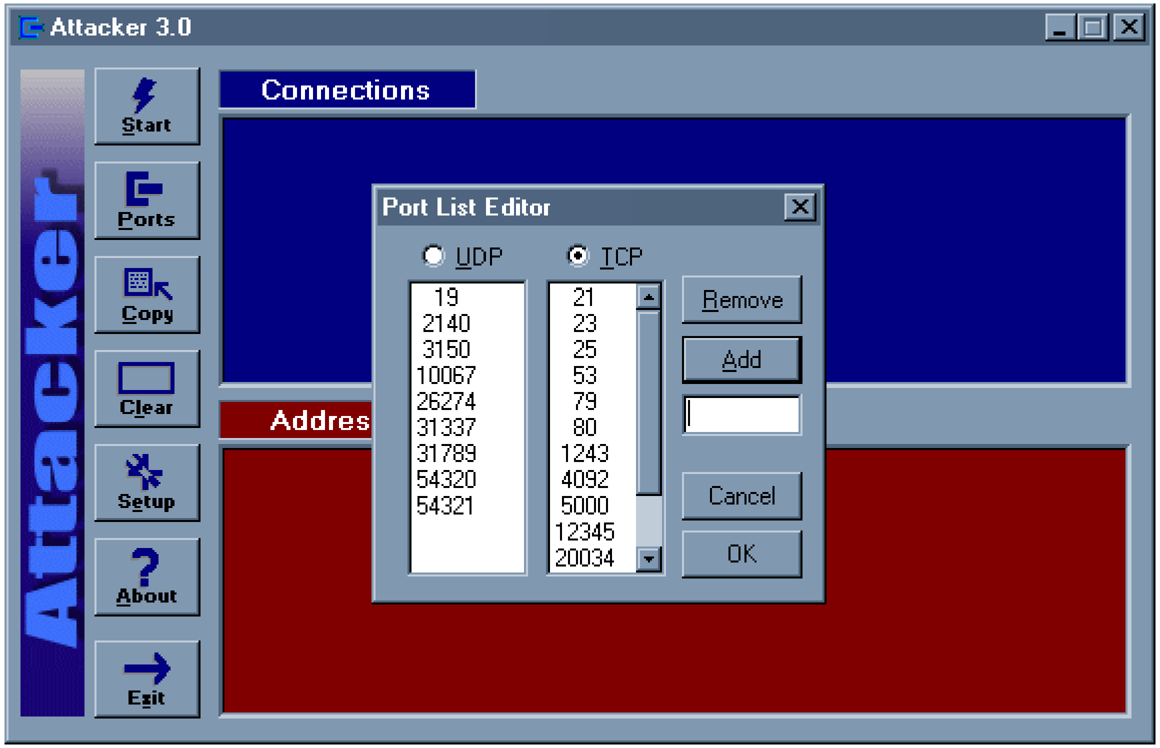
Attacker In Action



```
Program started: Apr 26 2002  01:08:29
Program stopped: Apr 26 2002  01:09:22

Program started: Apr 26 2002  01:09:27
Apr 26 2002  01:10:02        TCP connect from 63.208.188.252     port 1578 to port 23
```

```
63.208.188.252               dialup-63.208.188.252.Dial1.Baltimore1.Level3.net
```

Attacker Port Configuration



( ) UDP [✓ TCP]

UDP	TCP
19	21
2140	23
3150	25
10067	53
26274	79
31337	80
31789	1243
54320	4092
54321	5000
	12345
	20034

Jailing Applications

- Use `chroot` to “jail” applications that are frequently attacked
  - Look for unusual activity within the jail area - such as files being created, or executed
  - Make sure the jail has a writeable /tmp directory! Lots of hack-tools wind up in /tmp
    - Consider setting /tmp as no-exec

Trapping Jailbreaks

- Lots of hacker scripts rely not only on /tmp, they rely on the `rm` command - here are some fun ideas: (you can get fancier!)
  - Put a version of `/bin/false` in the jail area called `/bin/rm` - when the hackers' scripts delete the files it'll leave them there intact
  - Put a version of something called `/bin/sh` that simply records its parameters and inputs then sounds an alarm

SMTP Spam Honeypot

- Instead of running sendmail in queue-processing mode, use:

  sendmail -bd

- Turn on open relaying:

```
FEATURE(`promiscuous_relay')dnl
```

- Change delivery mode to queue instead of performing automatic delivery

Queue Options

- Change:

```
# default delivery mode
O DeliveryMode=background
```

- To:

```
# default delivery mode
# Mail never gets delivered if sendmail
is run without
# a "-q" option
O DeliveryMode=queue
```

Now Get Spammed

- You now have an open relay that accepts anything but doesn't re-send it
- Fun things to try:
  - Write scripts to watch the queue
  - Send spammers' test messages on their way but don't deliver the actual messages
  - Contact spammers' base of operations ISPs
- This idea brought to you by *Brad Spencer*

References

- Lots of stuff on:

http://www.tracking-hackers.com

http://www.honeynet.org

- Great free tools on:

http://www.foundstone.com/knowledge/forensics.html

- Mailing list:

honeypots-subscribe@securityfocus.com