

FACE: Automated Digital Evidence Discovery and Correlation

Andrew Case, Andrew Cristina, **Lodovico Marziale**,
Golden G. Richard III, Vassil Roussev



Me: PhD student and Research Assistant

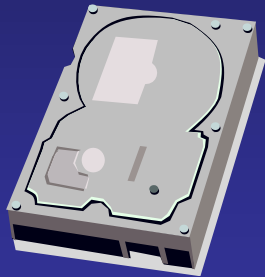
Department of Computer Science

vico@cs.uno.edu

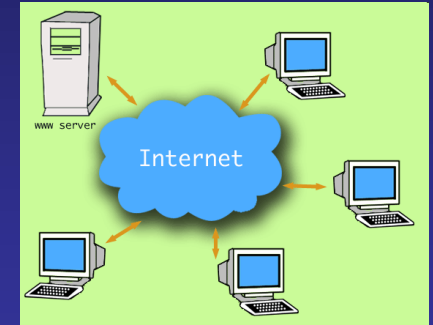
Forensics Challenge '08

- o 3 Pieces of evidence
 - pcap network capture
 - Linux RAM dump
 - Files from a user home directory
- o Object
 - Find user activity
 - Anything suspicious
 - Collaboration?
- o Live forensics

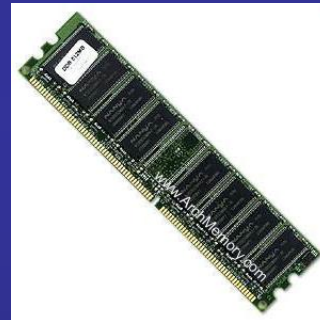
The Problem



RAM has info on running processes, loaded modules, and live network connections.



Disk has: filesystem, files, and MAC times



Network capture has data in packets; part of logical network connection.

Plethora of tool exist for each types of data from each source.

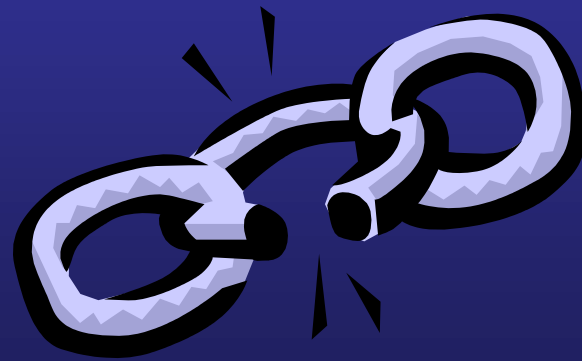
But, the objects of interest (users, processes, network connections) are described in part by each. Investigator must “connect the dots.”

I am far too **lazy** to do this manually!



Solution

- o FACE: Forensic Automated Correlation Engine
 - Correlate data for object across sources
 - Automate some of the “dot-connecting”
 - Penguin Power!
- o Good: Have disk and network stuff
 - Wireshark ...
 - Sleuthkit, Scalpel (shameless plug: new version soon!)
- o Bad: linux RAM tools?
 - Idetect for 2.4
 - Crash and gdb
 - ???



Introducing: *ramparser*

o Overview

- Linux 2.6, x86 (caveat later)
- C
- Processes
- Loaded modules
- Network connections
- More!
- Focus of this talk



Processes

o The Plan

- Phase 1: Find `task_struct` for “init”
- Phase 2: ?
- Phase 3: Profit!



o `task_struct`: the mother load

- `System.map (init_task)`
- Carving
- Caveat: `task_struct` representation
 - Kernel version
 - `.config`
 - **CRAZY** distro patches



Processes (more)

o Follow list or carve

o Basic: like **ps aux**

o Hardcore:

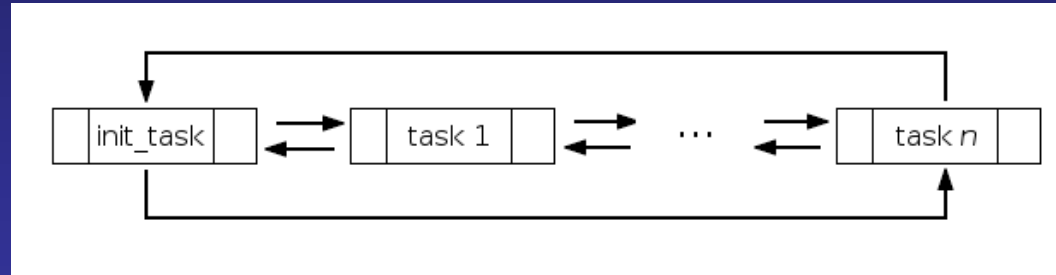
- Code and data segments
- Stack and heap

o Open files and network sockets: **/proc/<pid>/fd**

- Linkage to disk image and network capture

o Mappings: **/proc/<pid>/maps**

- Files
- libraries
- anonymous



Process Listing

```
#./ramparser challenge.mem -x
```

```
2152          501          501          gnome-session
```

```
/usr/bin/gnome-session
```

```
2262          501          501          bt-applet
```

```
bt-applet --sm-disable
```

```
2266          501          501          puplet
```

```
/usr/bin/python -tt /usr/bin/puplet
```

```
2269          501          501          vmware-user
```

```
/usr/lib/vmware-tools/bin32/vmware-user >/dev/null 2>&1 -blockFd 11
```

```
3048          501          501          firefox-bin
```

```
/usr/lib/firefox-1.5.0.12/firefox-bin
```

sock, socket and sk_buff, Oh my!

- o Struct sock, struct socket
- o Duplicate basic **netstat** functionality
- o Buffers: struct sk_buff
 - More complete network capture
 - Sockets have per-connection
 - Send queue (more useful)
 - Receive queue (less useful)
 - Still in kernel (so not in network capture)
 - Exfiltration detection anyone?

Netstat Output

```
#!/ramparser challenge.mem -N
```

Proto	Local Address	Foreign Address	State	PID	Program name
TCP	0.0.0.0:111	0.0.0.0:0	LISTEN	1959	portmap
TCP	0.0.0.0:22	0.0.0.0:0	LISTEN	2311	sshd
TCP	0.0.0.0:60126	0.0.0.0:0	LISTEN	2332	rpc.statd
TCP	192.168.20.128:45351	192.168.20.129:20	ESTABLISHED	2548	ftp
TCP	192.168.20.128:55071	192.168.20.129:80	ESTABLISHED	2521	firefox-bin
TCP	192.168.20.128:59447	192.168.20.129:21	ESTABLISHED	2548	ftp
UDP	0.0.0.0:111	0.0.0.0:0		1959	portmap
UDP	0.0.0.0:32768	0.0.0.0:0		2332	rpc.statd
UNIX				2195	klogd
UNIX				2301	dhclient3

Modules

- o struct module
 - Carved or from list
 - Better carved, rootkits!
- o Duplicates some **lsmod** functionality

Module Listing

```
#!/ramparser challenge.mem -m
```

0xd154f100	uhci_hcd	25421	MODULE_STATE_LIVE
0xd1561880	ohci_hcd	23261	MODULE_STATE_LIVE
0xd156ee00	ehci_hcd	32845	MODULE_STATE_LIVE
0xd157cc00	mptspi	20041	MODULE_STATE_LIVE
0xd157f780	sd_mod	22977	MODULE_STATE_LIVE
0xd16a5a80	jbd	56553	MODULE_STATE_LIVE
0xd16c1a80	mptbase	52833	MODULE_STATE_LIVE
0xd170ae80	ext3	123081	MODULE_STATE_LIVE
0xd1735a00	scsi_mod	130637	MODULE_STATE_LIVE
0xd1805a00	sg	35933	MODULE_STATE_LIVE

Now what?

- o Have overview of ram contents
- o Disk <-> RAM <-> network linkage
- o *ramparser* dump mode
- o Feeds FACE correlation engine
 - Also simple network capture parsing
 - And some key config files
 - */etc/passwd*
 - */etc/group*
 - */var/log/wtmp*
- o Lookee here! (just a taste)

Process: ftp

PID: 2548

UID: [root, 0](#)

GID: [root, 0](#)

- Code: [Hexdump Raw](#)
- Data: [Hexdump Raw](#)
- Stack: [Hexdump Raw](#)
- Heap: [Hexdump Raw](#)

• Files:

- FD: 0 [/pts/0](#)
- FD: 1 [/pts/0](#)
- FD: 2 [/pts/0](#)
- FD: 3 [socket:\[6513\]](#)
- FD: 4 [socket:\[6513\]](#)
- FD: 5 [socket:\[6513\]](#)
- FD: 6 [/root/file2](#)

• Sockets:

- Inode: [6513](#) 192.168.20.128:59447 to 192.168.20.129:21 Send Buffer: 0 entries, Recv Buffer: 0 entries.
- Inode: [6513](#) 192.168.20.128:59447 to 192.168.20.129:21 Send Buffer: 0 entries, Recv Buffer: 0 entries.
- Inode: [6513](#) 192.168.20.128:59447 to 192.168.20.129:21 Send Buffer: 0 entries, Recv Buffer: 0 entries.
- Inode: [6515](#) 192.168.20.128:45351 to 192.168.20.129:20 Send Buffer: **243 entries**, Recv Buffer: 0 entries.

• Mappings:

/usr/bin/netkit-ftp	Code: Hexdump Raw Data: Hexdump Raw
Anonymous	Data: Hexdump Raw
/usr/lib/gconv/gconv-modules.cache	Data: Hexdump Raw
/usr/lib/locale/locale-archive	Data: Hexdump Raw
/lib/tls/i686/cmov/libnss_nis-2.3.6.so	Code: Hexdump Raw Data: Hexdump Raw
/lib/tls/i686/cmov/libnsl-2.3.6.so	Code: Hexdump Raw Data: Hexdump Raw
Anonymous	Data: Hexdump Raw
/lib/tls/i686/cmov/libnss_compat-2.3.6.so	Code: Hexdump Raw Data: Hexdump Raw
/lib/tls/i686/cmov/libnss_files-2.3.6.so	Code: Hexdump Raw Data: Hexdump Raw
Anonymous	Data: Hexdump Raw
/lib/tls/i686/cmov/libdl-2.3.6.so	Code: Hexdump Raw Data: Hexdump Raw
/lib/tls/i686/cmov/libc-2.3.6.so	Code: Hexdump Raw Data: Hexdump Raw
Anonymous	Data: Hexdump Raw
/lib/libncurses.so.5.5	Code: Hexdump Raw Data: Hexdump Raw

[Home](#) [Users](#) [Groups](#) [Processes](#) [Packets](#)

User: root

UID: 0

GID: [0](#), [root](#)

Shell: [/bin/bash](#)

Home directory: [/root](#)

Last login: 2008-3-12T9:34:57

Processes belonging to user:

Name:	PID:	Owner:	Open Files:	TCP/IP Sockets:	Mappings:
init	1	root	1	0	10
migration/0	2	root	0	0	0
ksoftirqd/0	3	root	0	0	0
watchdog/0	4	root	0	0	0
migration/1	5	root	0	0	0
ksoftirqd/1	6	root	0	0	0

getty	2383	root	3	0	7
bash	2395	root	4	0	16
startx	2400	root	4	0	11
xinit	2416	root	4	0	11
Xorg	2417	root	13	0	57
fluxbox	2421	root	4	0	45
xterm	2425	root	5	0	39
bash	2426	root	4	0	16
pdflush	2515	root	0	0	0
firefox-bin	2521	root	58	1	118
ftp	2548	root	8	4	19

Files open:

- [/dev/initctl](#) opened by [init 1](#)
- [/dev/input/mice](#) opened by [Xorg 2417](#)
- [/dev/null](#) opened by [cron 2349](#)
- [/dev/null](#) opened by [cron 2349](#)
- [/dev/null](#) opened by [cron 2349](#)
- [/dev/null](#) opened by [sshd 2311](#)
- [/dev/null](#) opened by [sshd 2311](#)
- [/dev/null](#) opened by [sshd 2311](#)
- [/dev/null](#) opened by [dhclient3 2301](#)
- [/dev/null](#) opened by [dhclient3 2301](#)

Conclusions

- o DFRWS challenge
 - I'm too lazy to connect all the dots
- o *ramparser*:
 - processes including ...
 - netstat and socket buffers
 - Modules
- o FACE for automatic correlation

Future Work

- o Really “present” work (half done)
- o Generic 2.6 version
 - Dynamically build task_struct representation
 - Static symbols in System.map
 - From scratch
- o Timestamp madness
- o More RAM parsing fun
 - Block cache
 - Integrate swap

Questions / Comments?

This is the end
Beautiful friend
This is the end
My only friend, the end
- JM

vico@cs.uno.edu