Live Investigation

Daniel Kouril

Utility of memory for digital forensics

- Specifics of memory analysis
  - Hard to hide something
  - Processes are “unlocked”
  - Independent view on OS structures
- What can be found
  - Rootkits, hidden processes
  - Sensitive data not available elsewhere
  - Unprotected program codes
  - Crucial information about the system state
- Tend to be complex
  - Data is very fragmented
  - Data might be incomplete

Examination of memory internals

- Pattern matching
  - R.E./patterns for URL, password prompts, sensitive data (keys, logins, history), . . .
  - Human readable strings
  - Common tools strings, grep available
- File carving
  - Obtaining text of program – scripts
- Specific data structures
  - Encryption keys
  - OS structures
    - Additional knowledge is necessary to establish the picture
- Process vs. system memory

HANDS-ON EXERCISES

Goal of the workshop

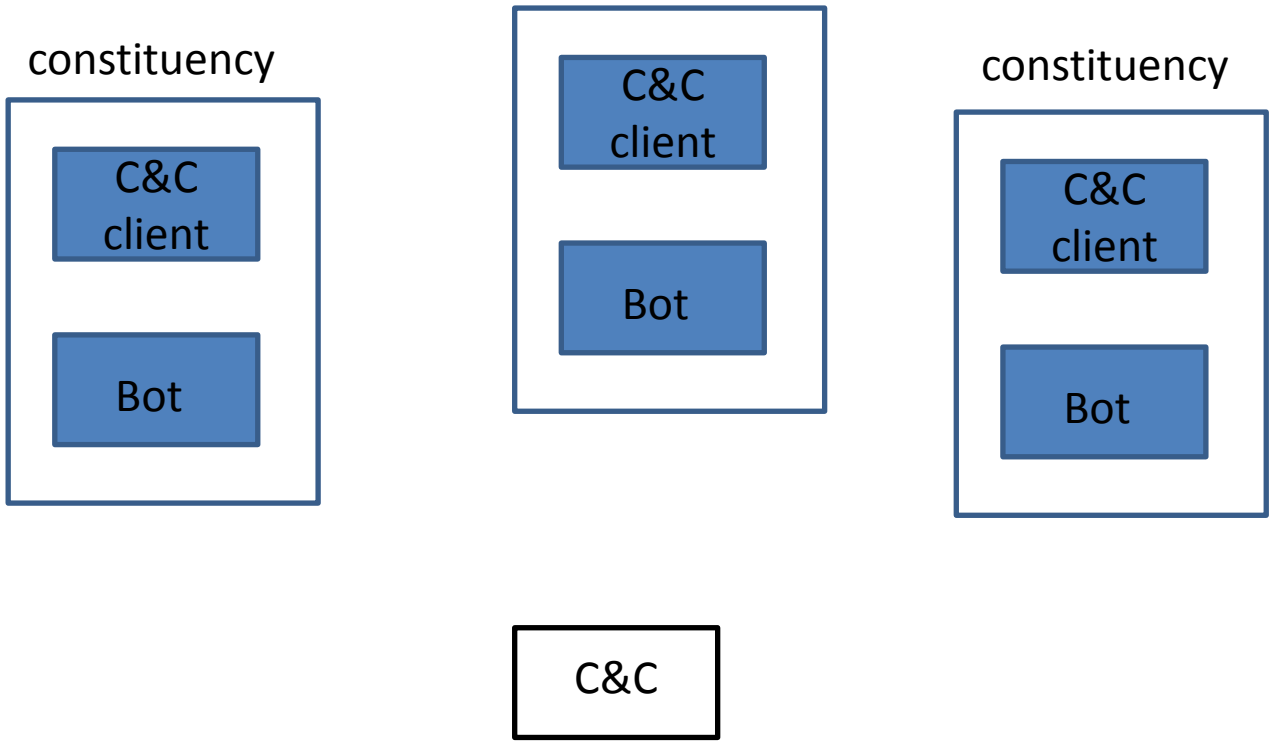
- Analysis of running process – open files, connections
- Analysis of running process memory

Your task

- You are an investigator of an incident in an organization
  - A suspicious traffic was detected from a machine
  - You need to find out more information about the situation (confirm or deny incident)
  - You are given access to the root account where needed
- Organization:
  - Limit your activities to your machines only
  - Several stages (not competition though)
    - Feel free to continue independently
- Stages:
  1. Find a suspicious process and reveal a suspicious connection to a C&C server (obtain access credentials)
  2. Examine the content of the server - identify another infected machine in your constituency and malware used
  3. Examine the other machine and determine what the attacker has recently done there
  4. Examine encrypted data stored in a process memory
  5. Extra: Identify how the attacker hides their processes

Access the machines

- Access the machine(s) (SSH), neede commands are available



QUICK PROCESS ANALYSIS

Processes

- Every running process has an entry in /proc filesystem containing process's metadata
- Processes can communicate
  - via network
  - via IPC, shared memory, ...
- Processes can manipulate with
  - files
  - devices (special type of file)
- Processes kernel structures are in memory

Process Analysis - procedure

- List running processes
  - Detect suspicious names of processes
  - Detect common processes running from non-standard location
- Analyze process metadata
  - Open file descriptors
  - Open network connections
- Dump and analyze process’s memory

Tools needed

- List of running processes
  - **ps**
- List of open file descriptors
  - **lsof, netstat**
- Dump process memory
  - **gcore**
- Common useful commands
  - **grep, strings, cat, w, lastlog**

Preparation

- Before you start process analysis, check if you are alone on the machine
- w

```
 14:30:17 up 11 days, 14:22,  2 users,  load average: 0.01, 0.02, 0.00
USER     TTY      FROM              LOGIN@   IDLE   JCPU   PCPU WHAT
root     ttyS0                     29Apr17 11days  0.02s  0.00s -bash
root     pts/0    poros.ics.muni.c 14:30    6.00s  0.00s  0.00s w
```

- la

```
Username         Port     From             Latest
root             pts/0    poros.ics.muni.c Wed May 10 14:30:11 +0200 2017
daemon                                     **Never logged in**
bin                                        **Never logged in**
```

Running process analysis

- List running processes

  ps auxf

Questions

- Any process consuming 100% CPU, strange names, owners, execution times, ….

Running process analysis

- List running processes

  ps auxf

Processes metadata

- Every process has metadata stored in /proc/{PID}/ filesystem

  ls /proc/1234/

attr
cpuset limits **net** projid_map statm **cwd** loginuid **ns** **root** status
autogroup
auxv
cgroup **exe** maps oom_adj sessionid **task** clear_refs **fd** mem oom_score setgroups timers
environ **map_files** numa_maps sched syscall
cmdline
comm
coredump_filter io mountstats personality stat

Open file descriptors

- Try to use commands
  - lsof –p 1234
  - netstat -tup | grep 1234

Open file descriptors

- List open file descriptors /proc/{PID}/fd/

ls -l /proc/1234/fd/

```
total 0
lrwx------ 1 root root 64 May 10 23:13 0 -> /dev/pts/3
lrwx------ 1 root root 64 May 10 23:13 1 -> /dev/pts/3
lrwx------ 1 root root 64 May 10 23:12 2 -> /dev/pts/3
lrwx------ 1 root root 64 May 10 23:13 3 -> socket:[180410]
```

Information might be hidden

- Does the output of lsof correspond to /proc/{PID}/fd?
- Does the output of the netstat correspond to /proc/{PID}/fd?

- Correlate output of ‘ps’ and content of /proc

Output of lsof

```
...
ftp       13290  root   0u   CHR  136,3      0t0      6 /dev/pts/3
ftp       13290  root   1u   CHR  136,3      0t0      6 /dev/pts/3
ftp       13290  root   2u   CHR  136,3      0t0      6 /dev/pts/3
ftp       13290  root   3u  IPv4 180410       0t0    TCP 10.4.0.59:36973->10.10.10.7:ftp (ESTABLISHED)
```

Questions

- How we can find e.g. passwords in memory?

Lets find where the password is

- Run own ftp client and provide username and password using fancy words
- e.g. ftp ftp.fi.muni.cz (user anonymous and some fancy password)
- Dump process memory
- Find fancy words
- Look around and remember the patterns

Process memory analysis

- Check availability of gcore utility

  apt-get install gdb

- Dump process memory

  gcore 1234

```
-r-- 1 root root 2336672 May 12 00:25 core.1234
```

Process memory analysis

- Analyse memory

  strings core.1234 | less

/home/sun/.terminfo /home/sun
``qS#
FqS#
AqS#
mickey
VqS#
0.0.0.0/96 14
ould hardly ever be necessary.

Questions

- What is the username used in ftp client?
- What is the password used in ftp client?
- Can we connect to the remote ftp server?

Connecting to FTP server

- ftp 202.169.169.xyz

```
ftp> passive
Passive mode on.
ftp> ls
 229 Entering Extended Passive Mode (|||47740|)
150 Opening ASCII mode data connection for file list
-rwxr-xr-x   1 0        0           10128 May 10 16:58 systemd-policy
-rw-r--r--   1 0        0             170 May 19  2015 welcome.msg
226 Transfer complete
```

Check the other machine

- Does the file occur somewhere on the machine?
- Is there a relevant process running?

Suspicious process

- Firstly search among running processes

  ps auxf

- Compare output with

  ps aux

root 11795 0.0 0.0 25816 2828 ? Ss 00:53 0:00 /lib/systemd/systemd-policy

Analyze suspicious process

- Look at open file descriptors

  ls –l /proc/1234/fd

```
total 0
lr-x------ 1 root root 64 May 10 23:41 0 -> pipe:[181524]
l-wx------ 1 root root 64 May 10 23:41 1 -> pipe:[181525]
l-wx------ 1 root root 64 May 10 23:41 2 -> pipe:[181526]
lrwx------ 1 root root 64 May 10 23:41 3 -> socket:[181535]
```

Dump memory

```
cd /tmp
 gcore 1234
```

Analyze process memory

- Look at strings output

  strings /tmp/core.1234 | less

- What do you see there?

Assignment

- The first machine was found to be used by the attacker to store encrypted messages. You are tasked to perform live analysis of a malicious process and find out what it hides.
- Task 1: There is a suspicious process running on the machine, which is listening on non-standard port (higher then 50000). Discover the port number that the process is bound to.
- Task 2: Extract the AES key that the process is using to decrypt received data.
- Task 3: Find the name of the file that the process is using to store encrypted data it received.
- Task 4: Decrypt the message stored in the file using the revealed encryption key

Hints

- The process in question does not do any attempt to hide the key in its memory.
- For the key extraction it is advised to use the aes-finder tool (available on the machine).
- The process to be examined keeps the file to store messages persistently open.
- You can use openssl tool to test the decryption (note that you need to specify proper parameters - salt, padding, type)