Understanding Linux Malware

Emanuele Cozzi[1], Mariano Graziano[2], Yanick Fratantonio[1], Davide Balzarotti[1]

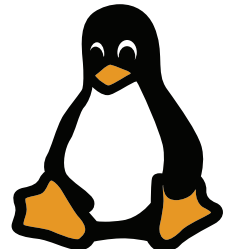
[1]EURECOM

[2]Cisco Systems, Inc.

IEEE Symposium on Security & Privacy, May 2018

Malware and operating systems

Malware and operating systems



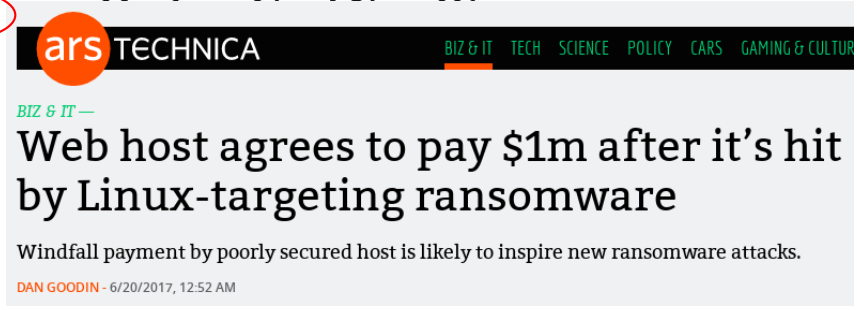
Linux malware on the rise

Linux malware on the rise

The New York Times

Mirai

Erebus



Linux malware on the rise

OutlawCountry

rai

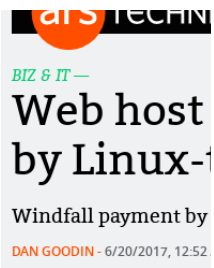
Linux malware on the rise

OutlawCountry

Linux malware: Leak exposes CIA's OutlawCoun

OutlawCountry malware sends

By Liam Tung | July 4, 2017 -- 11:50 GMT (



LILY HAY NEWMAN SECURITY 04.09.18 01:56 PM

A LONG-AWAITED IOT CRISIS IS HERE, AND MANY DEVICES AREN'T READY

Objectives

- Develop a dynamic analysis sandbox for Linux binaries (and IoT devices)

Objectives

- Develop a dynamic analysis sandbox for Linux binaries (and IoT devices)
  - Previous studies only looked at the network behavior [1] [2]

[1] Antonakakis et al. "Understanding the mirai botnet," USENIX Security Symposium 2017.

[2] Yin Minn Pa et al. "IoTPOT: analysing the rise of IoT compromises," USENIX Workshop on Offensive Technologies 2015.

Objectives

- Develop a dynamic analysis sandbox for Linux binaries (and IoT devices)
  - Previous studies only looked at the network behavior [1] [2]
- Identify challenges and limitations of porting traditional techniques to the new environment

[1] Antonakakis et al. "Understanding the mirai botnet," USENIX Security Symposium 2017.

[2] Yin Minn Pa et al. "IoTPOT: analysing the rise of IoT compromises," USENIX Workshop on Offensive Technologies 2015.

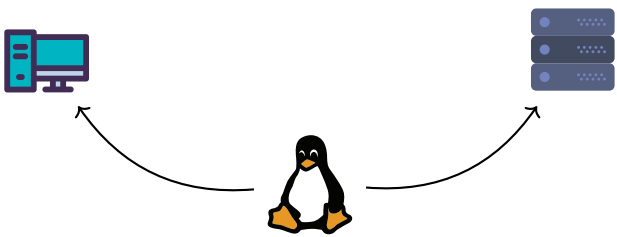
Objectives

- Develop a dynamic analysis sandbox for Linux binaries (and IoT devices)
  - Previous studies only looked at the network behavior [1] [2]
- Identify challenges and limitations of porting traditional techniques to the new environment
- Understand differences in the malware characteristics (packing, obfuscantion, VM detection, privilege excalation, persistence...) wrt Windows malware

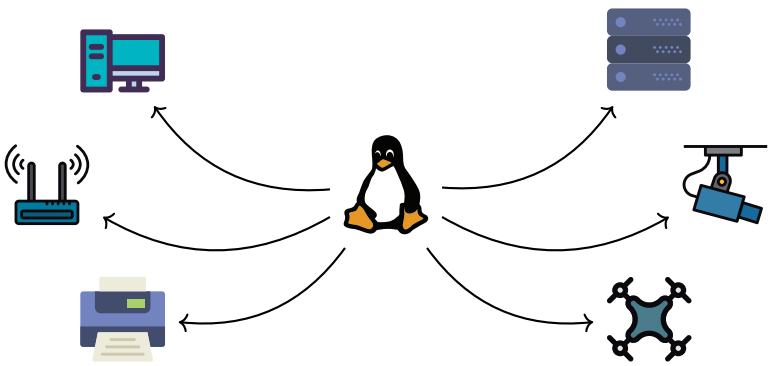
[1] Antonakakis et al. "Understanding the mirai botnet," USENIX Security Symposium 2017.

[2] Yin Minn Pa et al. "IoTPOT: analysing the rise of IoT compromises," USENIX Workshop on Offensive Technologies 2015.

Target devices

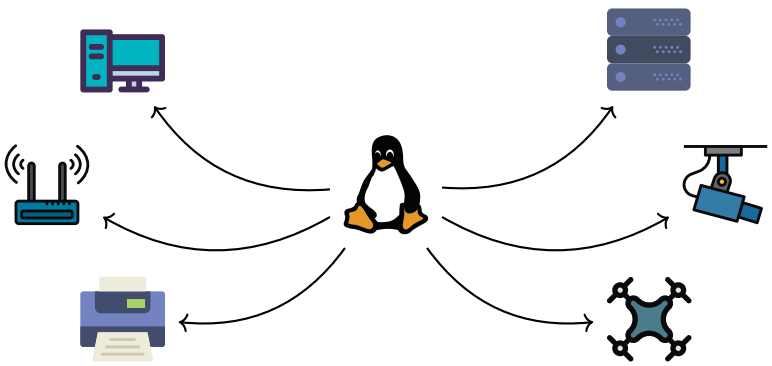


Target devices



Target devices

Diversity



Diversity

CPU: Intel

Diversity

CPU: Intel, ARM, MIPS, Motorola, Sparc

Diversity

CPU: Intel, ARM, MIPS, Motorola, Sparc

OS: Linux

Diversity

CPU: Intel, ARM, MIPS, Motorola, Sparc

OS: Linux, BSD, Android

Diversity

CPU: Intel, ARM, MIPS, Motorola, Sparc

OS: Linux, BSD, Android

Libraries: glibc

Diversity

CPU: Intel, ARM, MIPS, Motorola, Sparc

OS: Linux, BSD, Android

Libraries: glibc, uclibc, libpcap, libopencl

Diversity

```
invano at debian370-5 in ~:
$ file /tmp/malware
/tmp/malware: ELF 32-bit LSB executable, MIPS, MIPS-I version 1 (SYSV), statically linked, not stripped
```

Statically-linked ELF unportable

OS: Linux, BSD, Android

Libraries: glibc, uclibc, libpcap, libopencl

Diversity

```
invano at debian370-5 in ~:
$ file /tmp/malware
/tmp/malware: ELF 32-bit LSB executable, MIPS, MIPS-I version 1 (SYSV), statically linked, not stripped
```

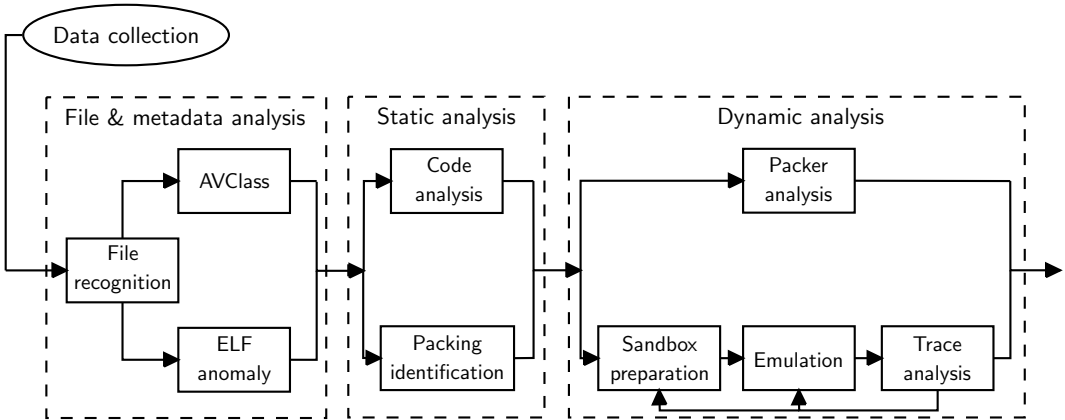
Statically-linked ELF unportable

```
Program Headers:
  Type           Offset   VirtAddr   PhysAddr   FileSiz MemSiz  Flg Align
  LOAD           0x000000 0x00008000 0x00008000 0x11404 0x11404 R E 0x8000
  LOAD           0x011408 0x00021408 0x00021404 0x001d0 0x0a7e4 RW  0x8000
  GNU_STACK      0x000000 0x00000000 0x00000000 0x00000 0x00000 RWE 0x4
```

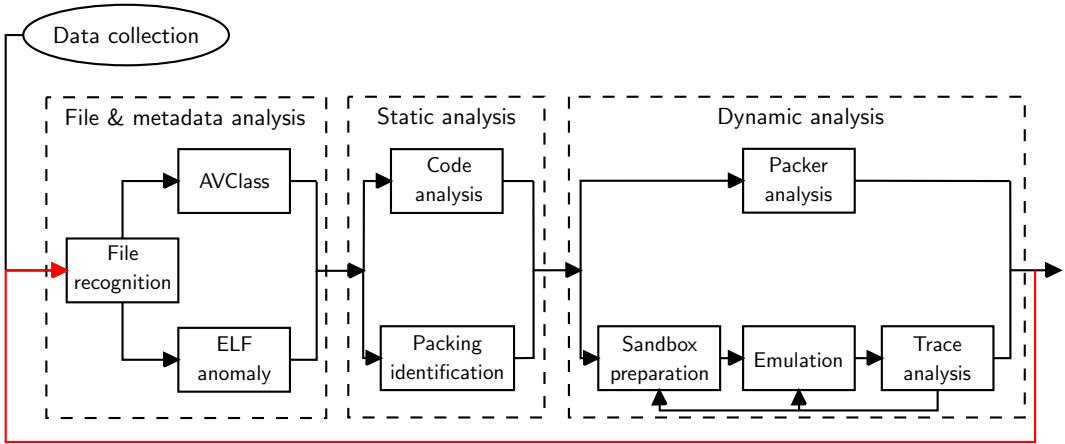
Libraries: glibc, uclibc, libpcap, libopenl

Unknown device

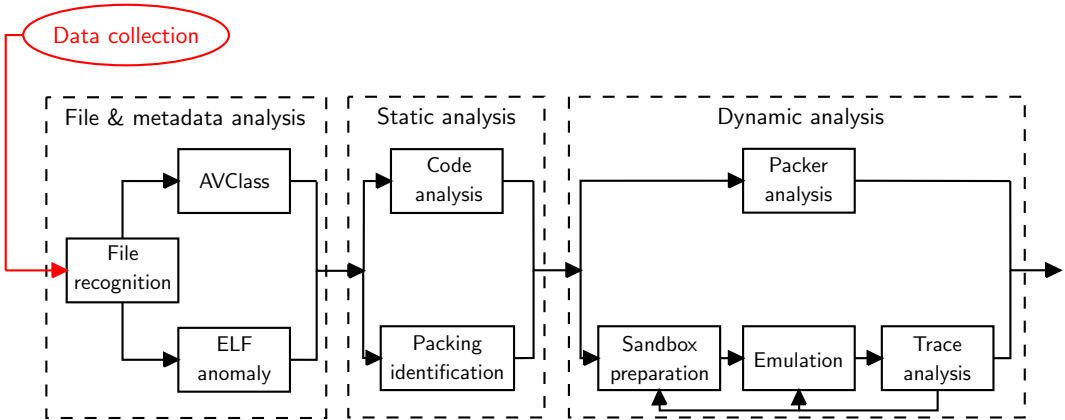
Analysis infrastructure



Analysis infrastructure



Analysis infrastructure



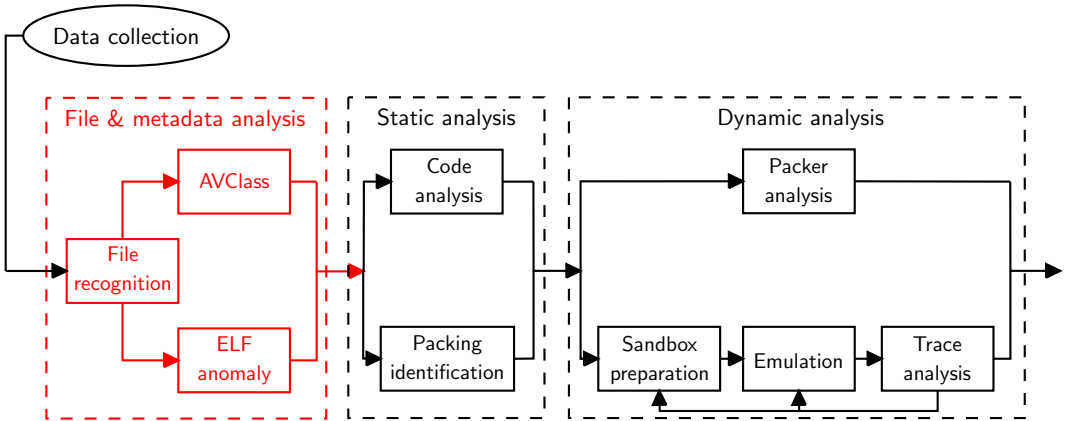
Data collection

From November 2016 to November 2017

200 *candidate* samples per day

Dataset of 10,548 Linux malware

File & metadata analysis



Dataset

Architecture	Samples	Percentage
X86-64	3018	28.61%
MIPS I	2120	20.10%
PowerPC	1569	14.87%
Motorola 68000	1216	11.53%
Sparc	1170	11.09%
Intel 80386	720	6.83%
ARM 32-bit	555	5.26%
Hitachi SH	130	1.23%
AArch64 (ARM 64-bit)	47	0.45%
others	3	0.03%

Distribution of the 10,548 downloaded samples across architectures

Dataset

Architecture	Samples	Percentage
X86-64	3018	28.61%
MIPS I	2120	20.10%
PowerPC	1569	14.87%
Motorola 68000	1216	11.53%
Sparc	1170	11.09%
Intel 80386	720	6.83%
ARM 32-bit	555	5.26%
Hitachi SH	130	1.23%
AArch64 (ARM 64-bit)	47	0.45%
others	3	0.03%

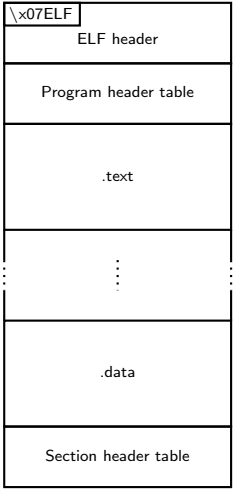
Distribution of the 10,548 downloaded samples across architectures

Dataset

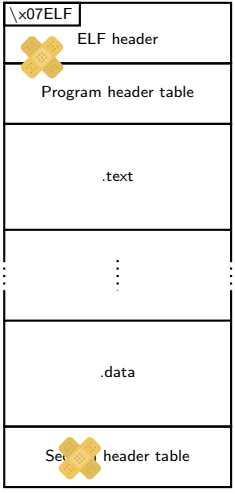
Architecture	Samples	Percentage
X86-64	3018	28.61%
MIPS I	2120	20.10%
PowerPC	1569	14.87%
Motorola 68000	1216	11.53%
Sparc	1170	11.09%
Intel 80386	720	6.83%
ARM 32-bit	555	5.26%
Hitachi SH	130	1.23%
AArch64 (ARM 64-bit)	47	0.45%
others	3	0.03%

Distribution of the 10,548 downloaded samples across architectures

ELF manipulation

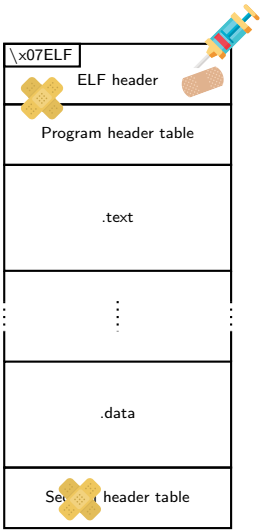


ELF manipulation



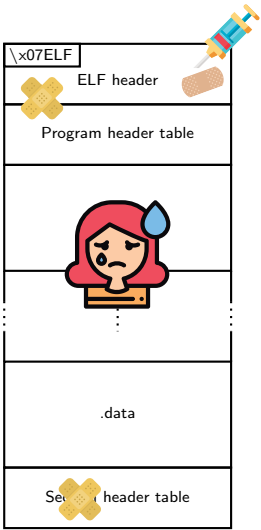
- Anomalous ELF
  - Sections table removed

ELF manipulation



- Anomalous ELF
  - Sections table removed
- Invalid ELF
  - Segments table points beyond file
  - Overlapping header/segment
  - Sections table points beyond file

ELF manipulation



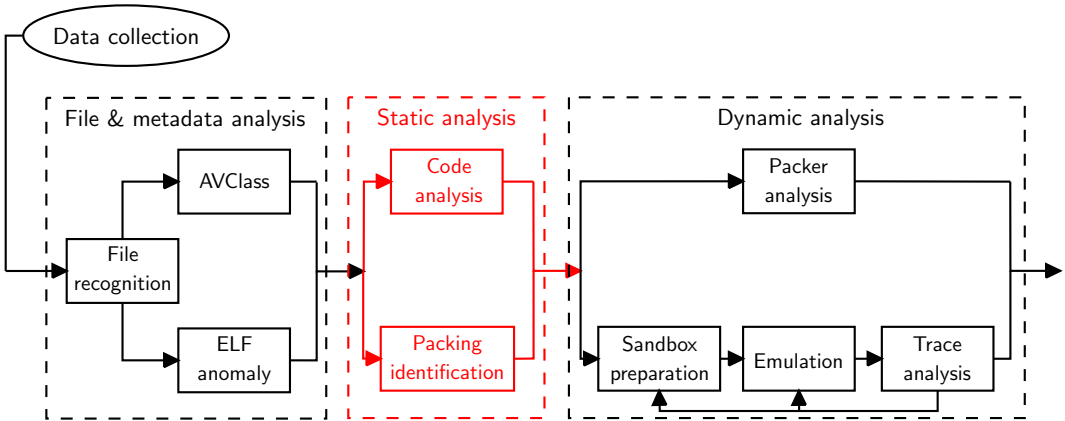
- Anomalous ELF
  - Sections table removed
- Invalid ELF
  - Segments table points beyond file
  - Overlapping header/segment
  - Sections table points beyond file
- Problems with common analysis tools
  - ✘ readelf 2.26.1
  - ✘ GDB 7.11.1
  - ✘ pyelftools 0.24
  - ✔ IDA Pro 7

AVClass[3]

Pymadro	Miner	Ebolachan	Golad	Lady	Connectback	Mirai
Elfpatch	Pomedaj	Liora	Ddostf	Cinarek	Ztorg	Elknot
Shishiga	Aidra	Chinaz	Fysbis	Ganiw	Scanner	Roopre
Mrblack	Equation	Logcleaner	Sniff	Tsunami	Sshbrute	Probe
Znaich	Erebus	Xingyi	Xaynnalc	Gafgyt	Flood	Coinminer
Bassobo	Killdisk	Eicar	Remaiten	Bossabot	Midav	Getshell
Drobur	Webshell	Dcom	Cloudatlas	Luabot	Iroffer	Mayday
Grip	Darkkomet	Prochider	Ircbot	Xhide	Portscan	Xunpes
Diesel	Setag	Raas	Shelma	Shellshock	Nixgi	Wuscan
Cleanlog	Sshdoor	Psybnc	Themoon	Rekoobe	Intfour	Pulse
Sickabs	Hajime	Hijacker	Mumblehard	Darlloz	Sotdas	Ladvix
Pnscan	Ropys	Lightaidra	Moose	Vmsplice	Ddoser	Spyeye

[3]Sebastin et al. "Avclass: A tool for massive malware labeling," International Symposium on Research in Attacks, Intrusions, and Defenses 2016.

Static analysis



Packing

```
oooooo      ooo  oooooooooo.    ooooooo   ooooo
‘888’       ‘8’  ‘888    ‘Y88.   ‘8888     d8’
 888         8    888    .d88’     Y888..8P
 888         8    888ooo88P’        ‘8888’
 888         8    888                .8PY888.
 ‘88.     .8’    888              d8’   ‘888b
   ‘YbodP’      o888o            o888o  o88888o

The Ultimate Packer for eXecutables
```

- Vanilla UPX and custom variants are the prevalent packers (almost 4% of the dataset)

Packing

```
oo          ooo o   oooooo.        oooo   ooooo
‘8          ‘8’ ‘888    ‘Y88.   ‘8888     d8’
 888         8   888    .d88’     Y8 8..
 888         8   8      88P’        ‘8888’
   8         8   888                  Y888.
   8.            88               d8’   ‘88
   ‘YbodP’       88o             o888o   o888

The Ultimate Packer for eXecutables
```

- Vanilla UPX and custom variants are the prevalent packers (almost 4% of the dataset)

Packing

```
oo          ooo o  oo   o.       oooo     ooo
‘8          ‘8’ ‘888    ‘Y8     ‘8888     d8’
 888         8   888    .d8       Y8 8..
  8              8     88P’        ‘88
   8         8   888                 Y
   8.             8               d8’  ‘88
   ‘Yb dP’        88o           o888     888

The Ultimate Packer for eXecutables
```

- Vanilla UPX and custom variants are the prevalent packers (almost 4% of the dataset)

Packing

```
oo            o  o   o     o.       ooo       oo
‘8          ‘8’  ‘88     ‘Y8        88      d8
 8            8    8       .d8        8  8..
                   8        8          ‘88
   8         8     88                    Y
   8.                8                 8’   ‘88
     b dP’           88o              88       88

The Ultimate Packer for eXecutables
```

- Vanilla UPX and custom variants are the prevalent packers (almost 4% of the dataset)
  - modified magic bytes
  - modified strings
  - junk bytes

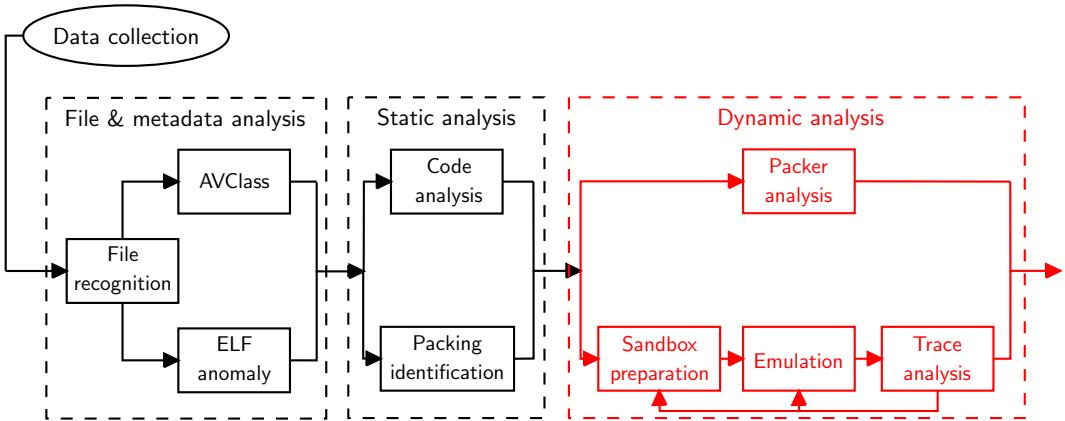
Packing

```
oo            o  o   o     o.        ooo       oo
‘8          ‘8’  ‘88      ‘Y8         88      d8
 8            8    8        .d8         8  8..
                   8         8           ‘88
   8         8     88                     Y
   8.                8                  8’   ‘88
     b dP’           88o              88       88

The Ultimate Packer for eXecutables
```

- Vanilla UPX and custom variants are the prevalent packers (almost 4% of the dataset)
  - modified magic bytes
  - modified strings
  - junk bytes
- *At least* one malware family is using a custom packer

Dynamic analysis



Behaviors

Process interaction

Process injection

Deception

Anti-debugging

Anti-execution

Persistence

Shell commands

Privileges escalation

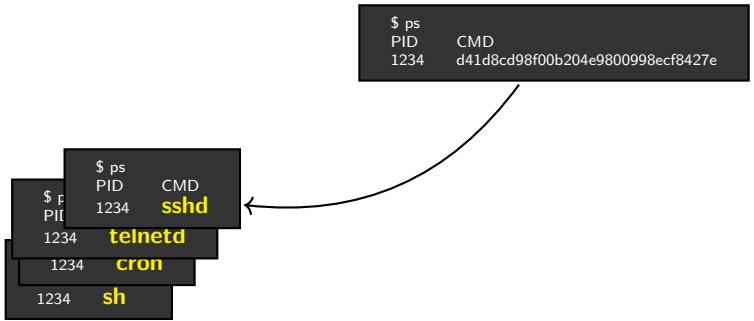
Sandbox detection

Information gathering

Processes enumeration

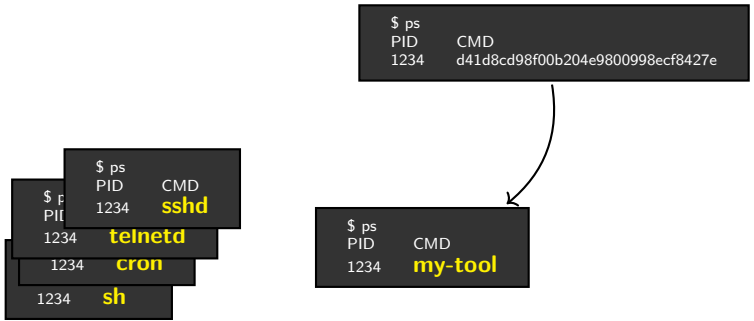
Required privileges

Deception



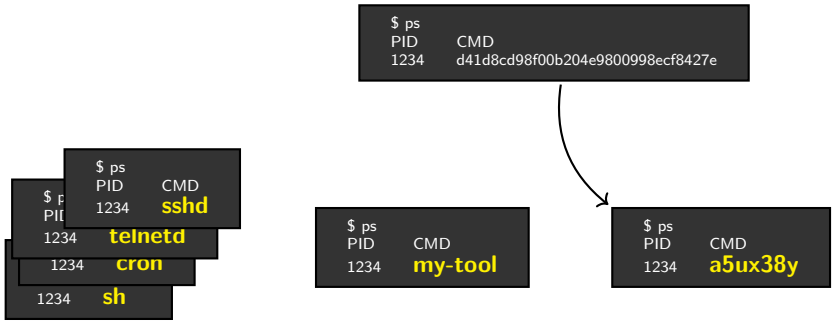
- Malicious processes assume new names to trick process listing tools
- 52% of the samples renamed the process

Deception



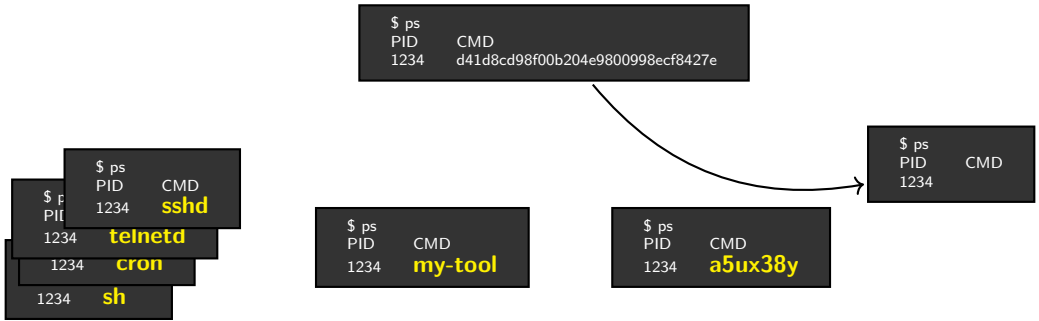
- Malicious processes assume new names to trick process listing tools
- 52% of the samples renamed the process

Deception



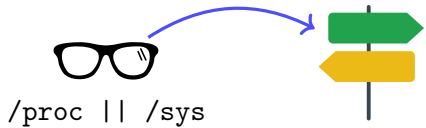
- Malicious processes assume new names to trick process listing tools
- 52% of the samples renamed the process

Deception



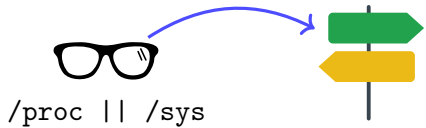
- Malicious processes assume new names to trick process listing tools
- 52% of the samples renamed the process

Evasion



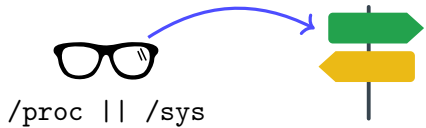
- Detect VMware, VirtualBox, QEMU, KVM but also OpenVZ, XEN or chroot jails

Evasion



- Detect VMware, VirtualBox, QEMU, KVM but also OpenVZ, XEN or chroot jails
- Malware may also check their file name before real execution

Evasion



- Detect VMware, VirtualBox, QEMU, KVM but also OpenVZ, XEN or chroot jails
- Malware may also check their file name before real execution

```
if (!sandbox) {
    //do evil
}
else {
    print("https://lmgtfy.com/q=how+to+@@@@@@@@@@@@@@@@@@@@")
    rm -rf /
}
```

- OS/ABI field in ELF header is not used

- OS/ABI field in ELF header is not used
- Malware executed by root or user

- OS/ABI field in ELF header is not used
- Malware executed by root or user
- Processes enumeration

- OS/ABI field in ELF header is not used
- Malware executed by root or user
- Processes enumeration
- Unstripped symbols (?)

Conclusions

- Linux malware still in its infancy
- Already a broad range of behaviors and tricks
- ELF binaries *could* run anywhere from a thermostat to a large server
- New research needed to overcome the lack of information about the execution environment

Thank you

https://padawan.s3.eurecom.fr/

Emanuele Cozzi
cozzi@eurecom.fr
@invano