Modern Linux Malware Exposed

Emanuele Cozzi @invano

Mariano Graziano @emd3l

RECON

MONTREAL

2018

About us

Emanuele Cozzi

@invano

PhD student at s3@Eurecom

Mariano Graziano

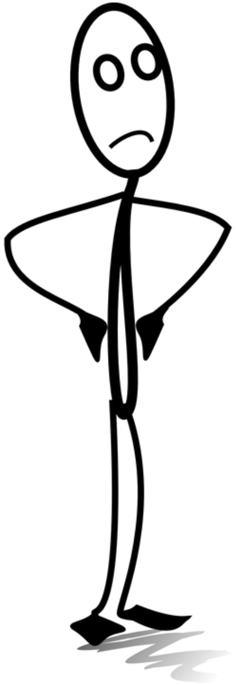
@emd3l

Security Researcher at Cisco Talos

Malware we fight - myth

Windows

Malware we fight - reality



Windows

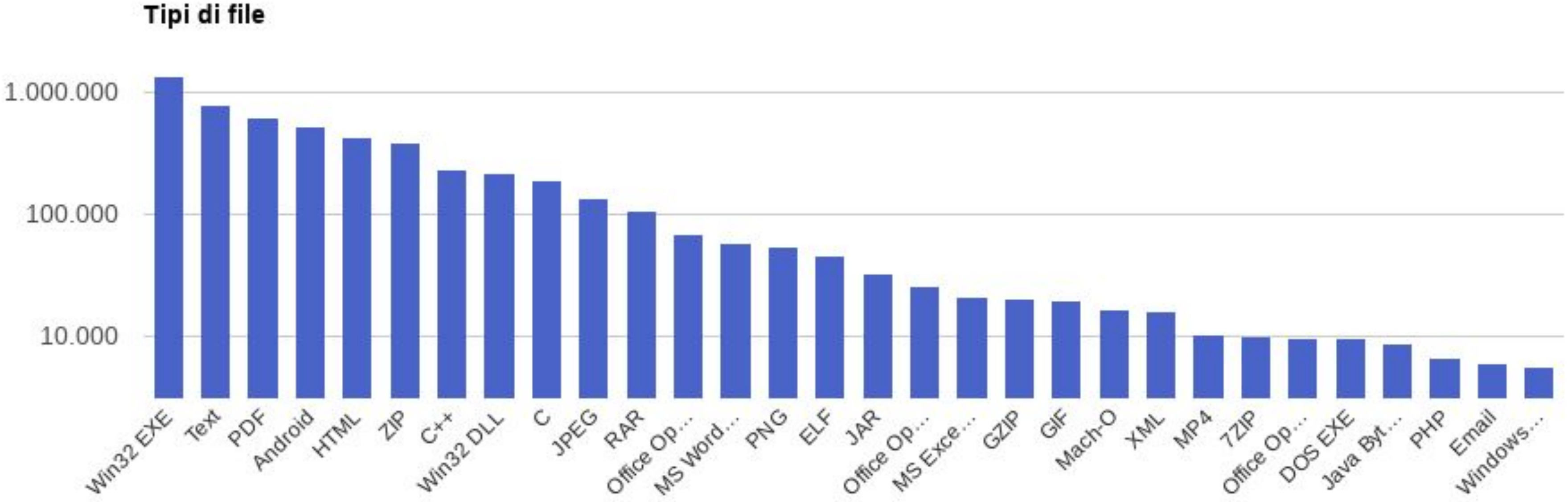
Linux

macOS

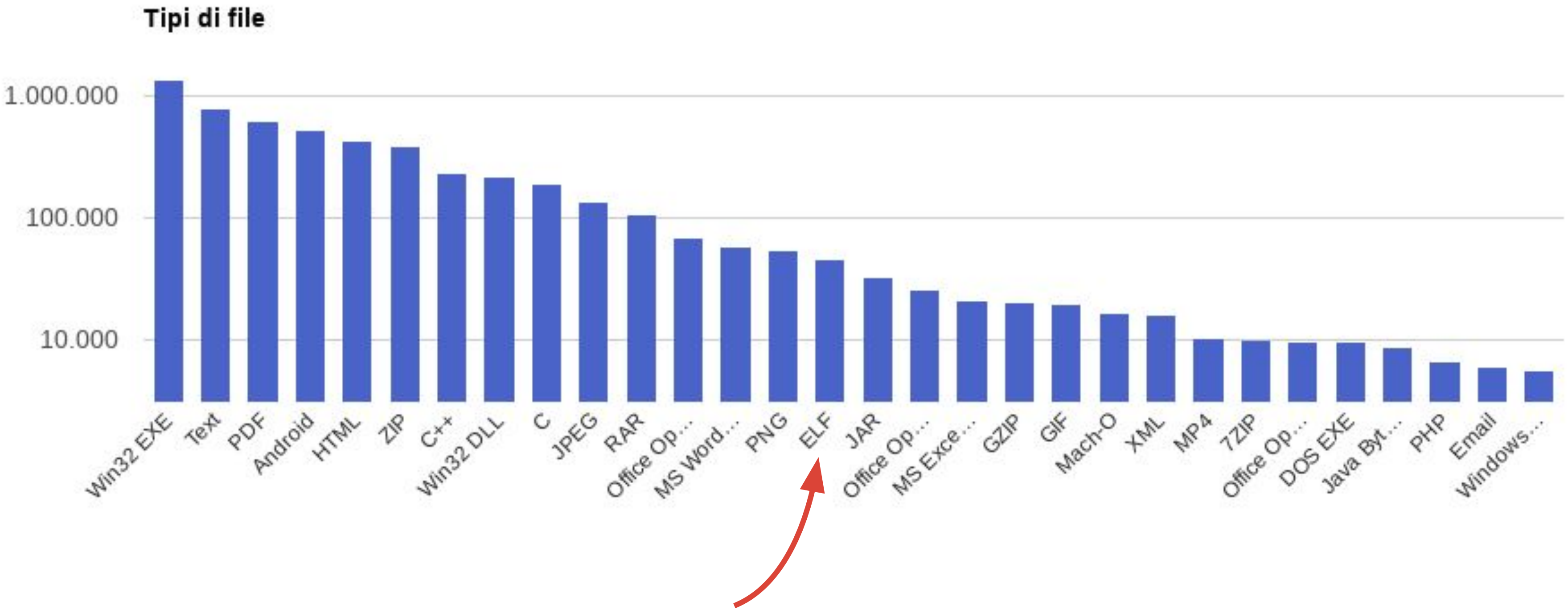
Android

. . .

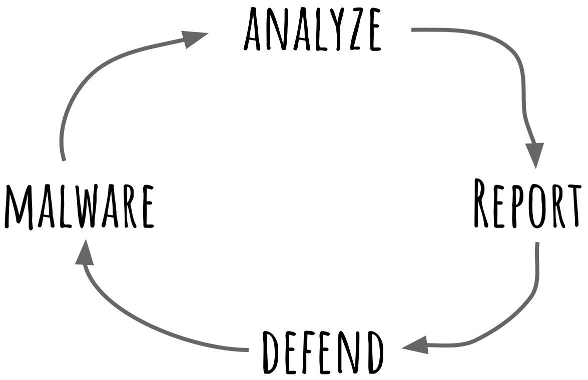
VirusTotal FILE SUBMISSIONS - LAST 7 DAYS



VirusTotal FILE SUBMISSIONS - LAST 7 DAYS



WINDOWS MALWARE



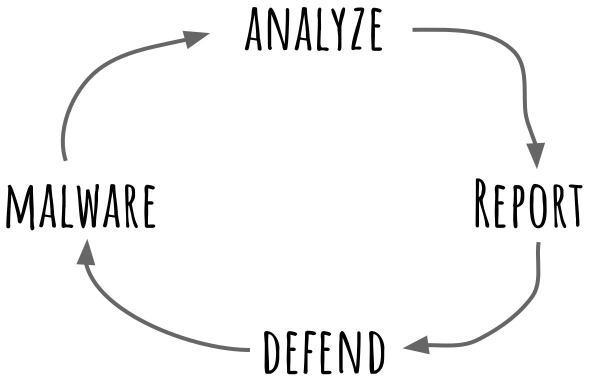
ANALYZE

Report

DEFEND

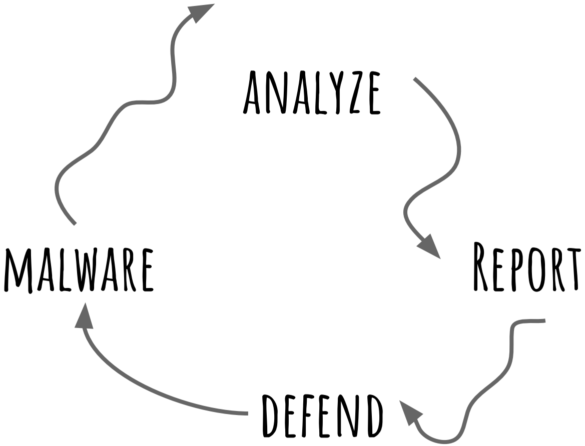
MALWARE

WINDOWS MALWARE



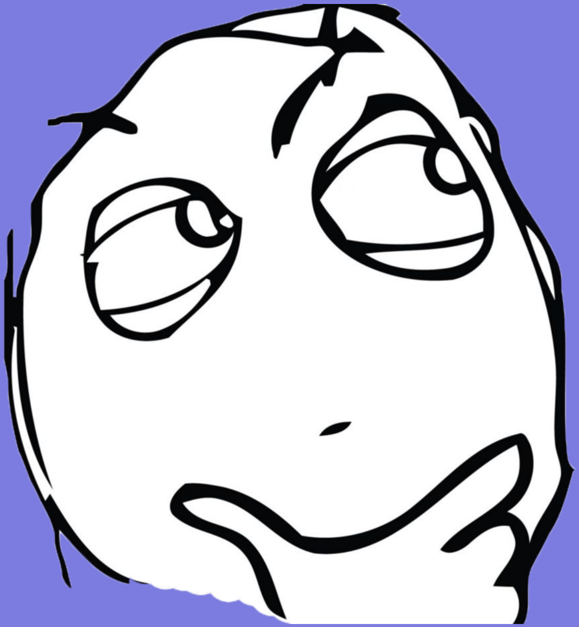
- We know their techniques
- We have tools
- We have sandboxes
- We built expertise
- ...
- We are not done yet

LINUX MALWARE

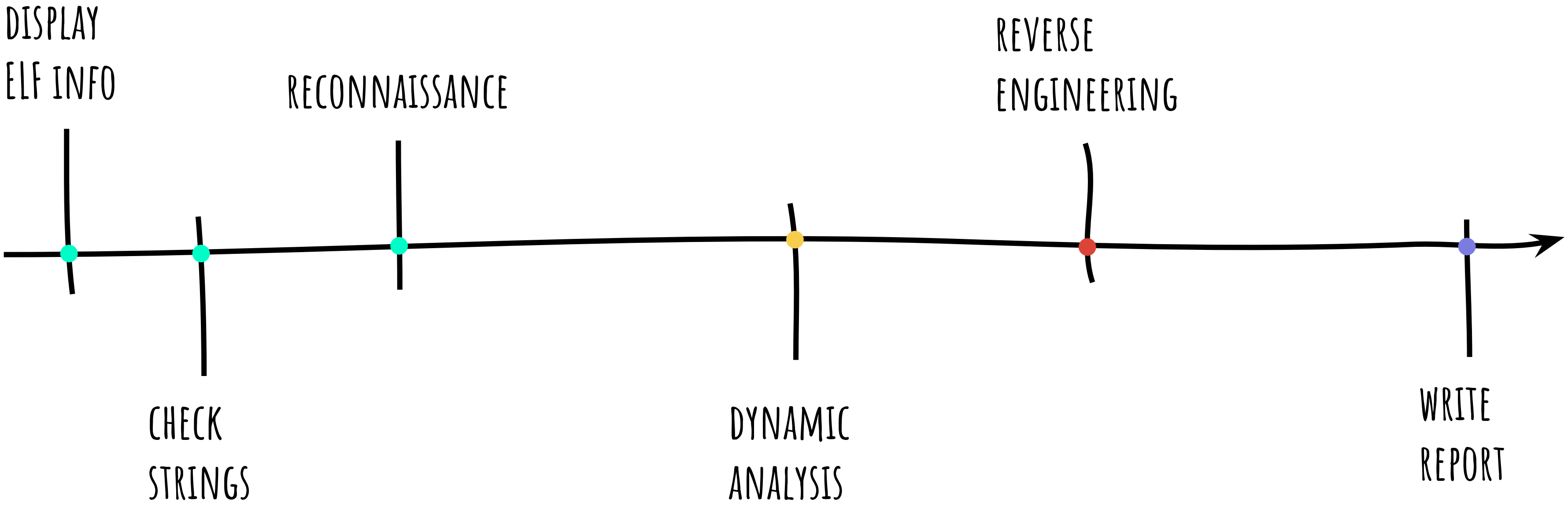


- We know their techniques?
- We have tools?
- We have sandboxes?
- We built expertise?
- ...
- We are not done yet

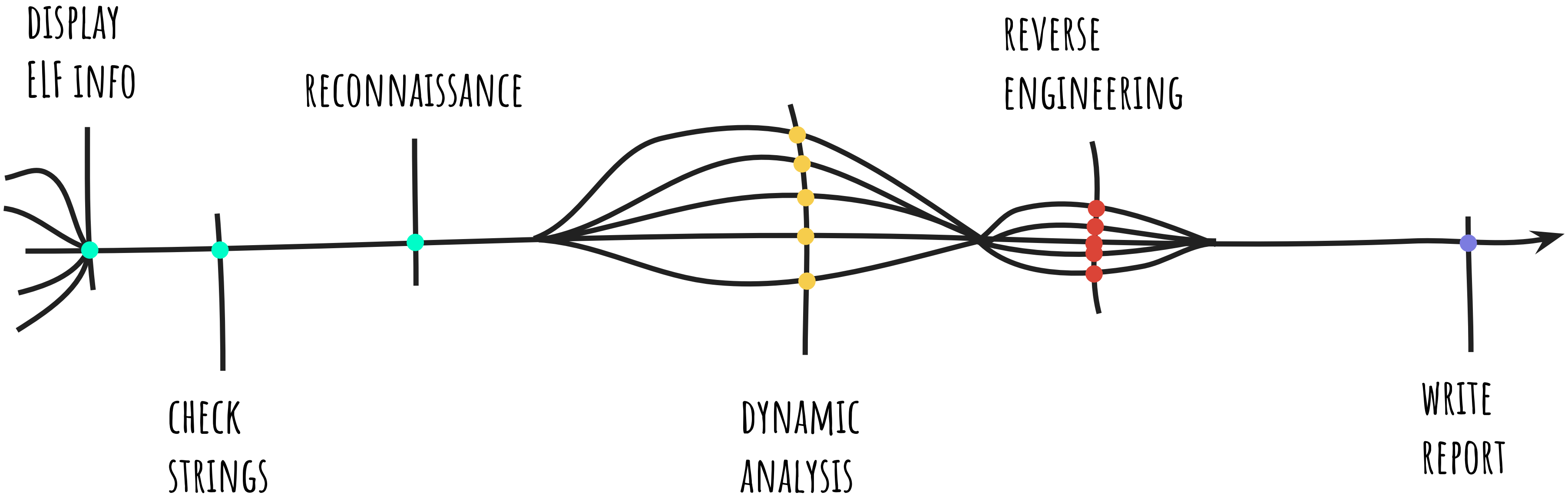
when I get a Linux Malware



Analysis process



ANALYSIS PROCESS



DISPLAY ELF INFO

CHECK STRINGS

RECONNAISSANCE

DYNAMIC ANALYSIS

REVERSE ENGINEERING

WRITE REPORT

Key problem: Diversity

- server, desktop, router, printer, camera

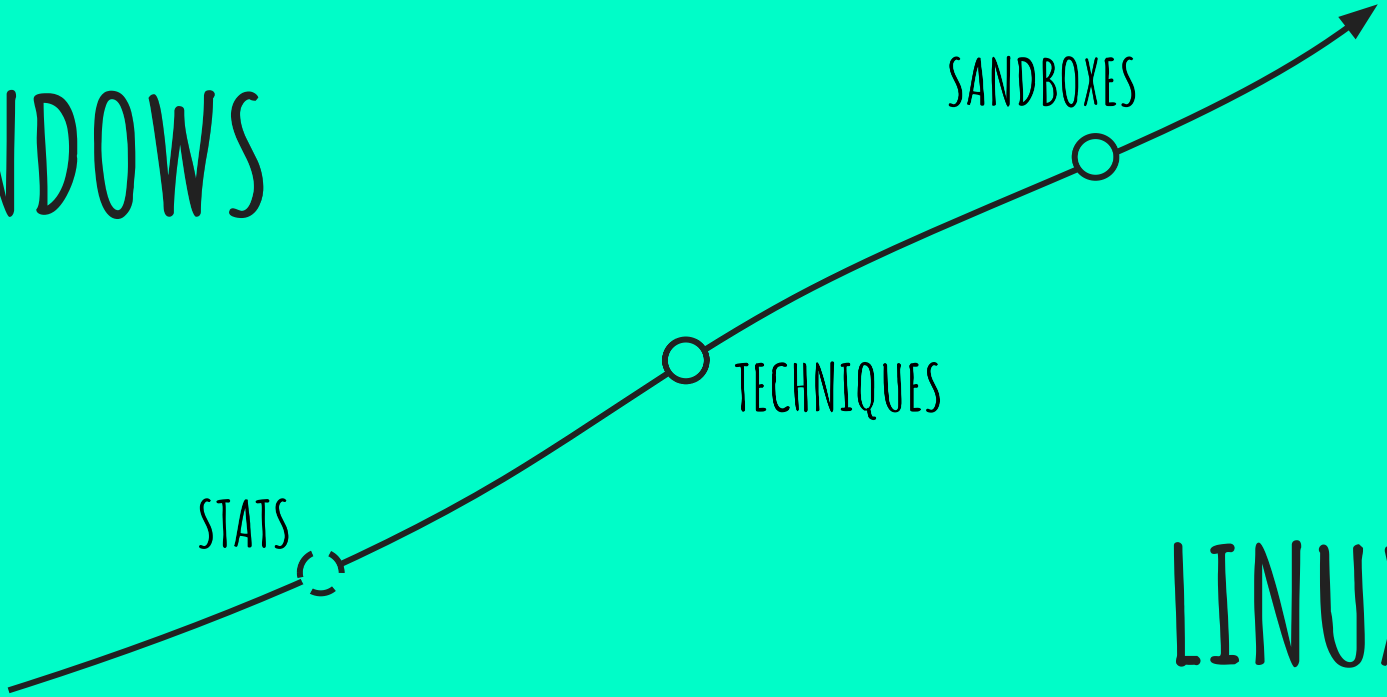
Key problem: Diversity

- server, desktop, router, printer, camera
- Intel, AMD, ARM, MIPS, PowerPC, Motorola, SPARC

Key problem: Diversity

- server, desktop, router, printer, camera
- Intel, AMD, ARM, MIPS, PowerPC, Motorola, SPARC
- Linux, FreeBSD, Android, Solaris, AIX

WINDOWS



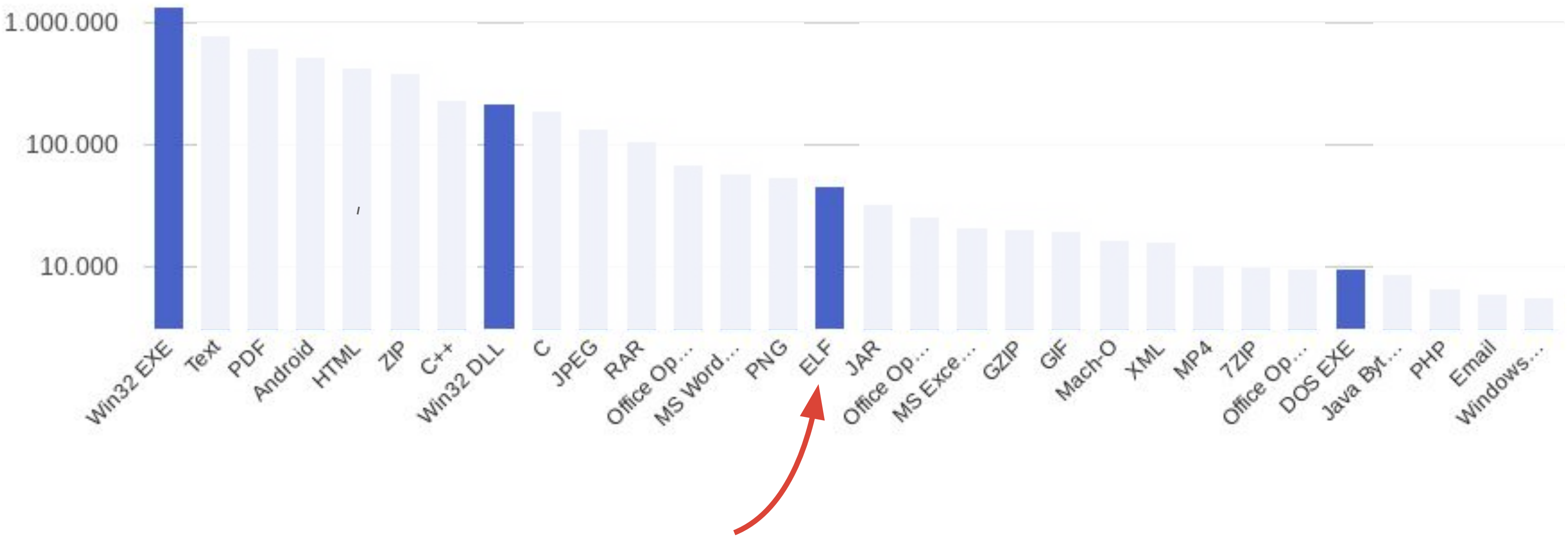
STATS

TECHNIQUES

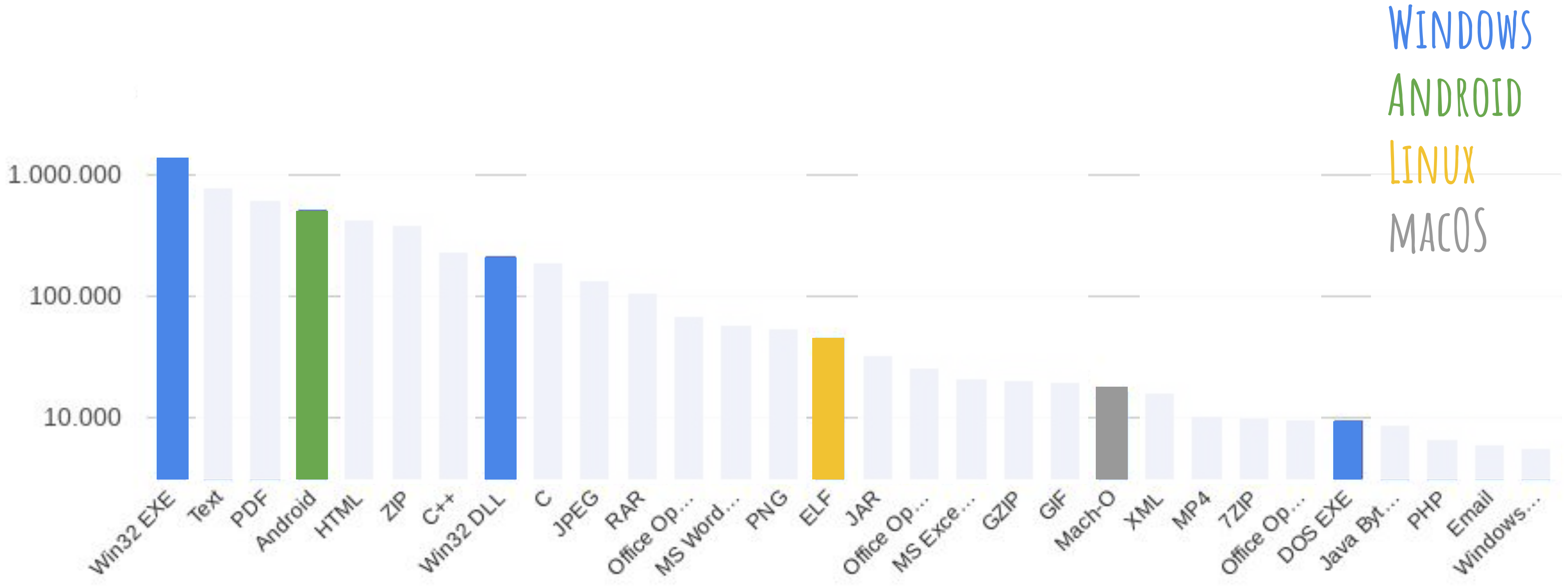
SANDBOXES

LINUX

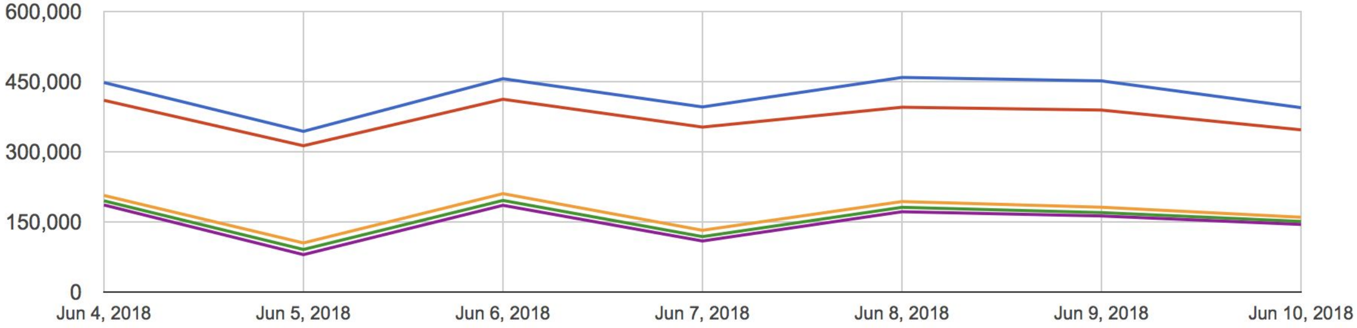
Current situation



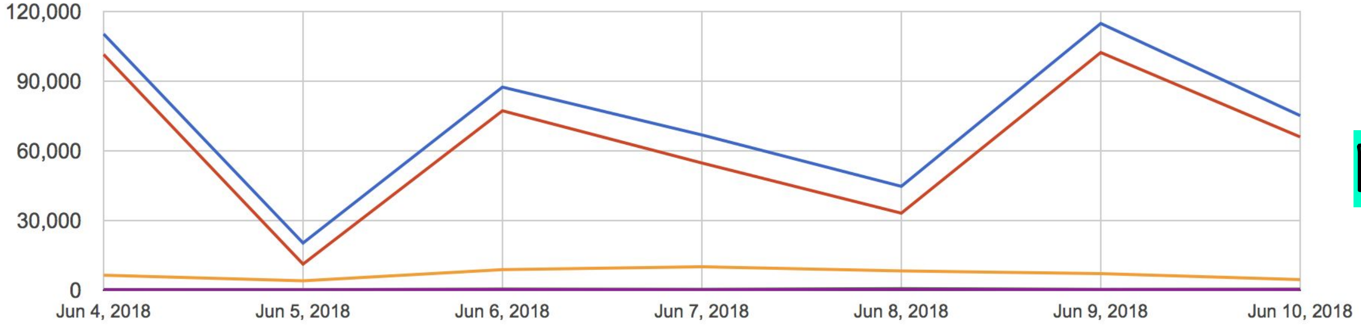
Current situation



VT Submissions

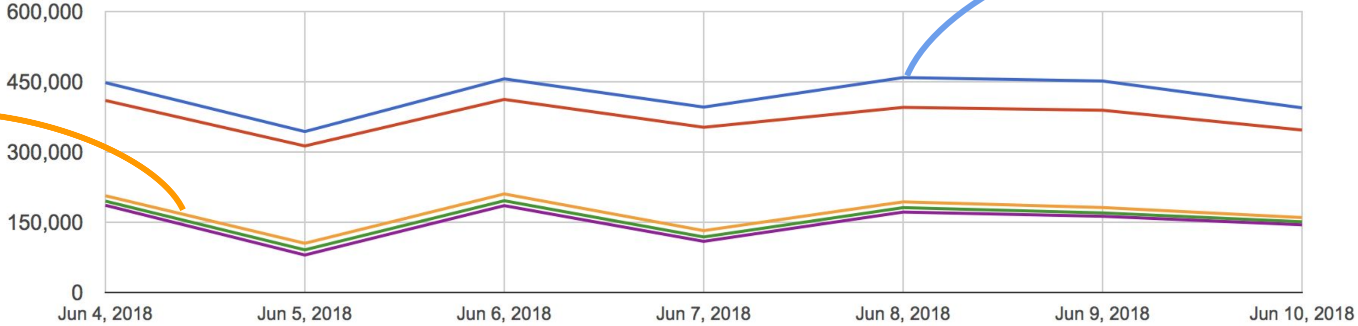


PE FILES



ELF FILES

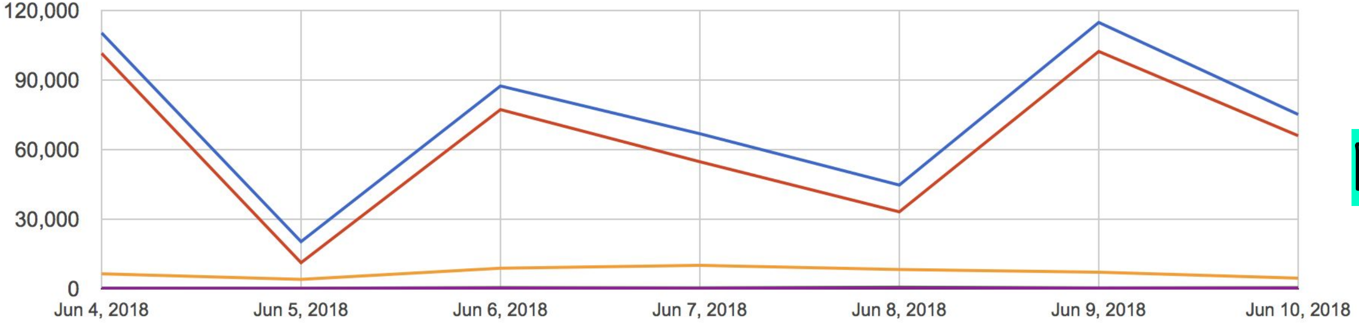
VT Submissions



TOTAL FILES

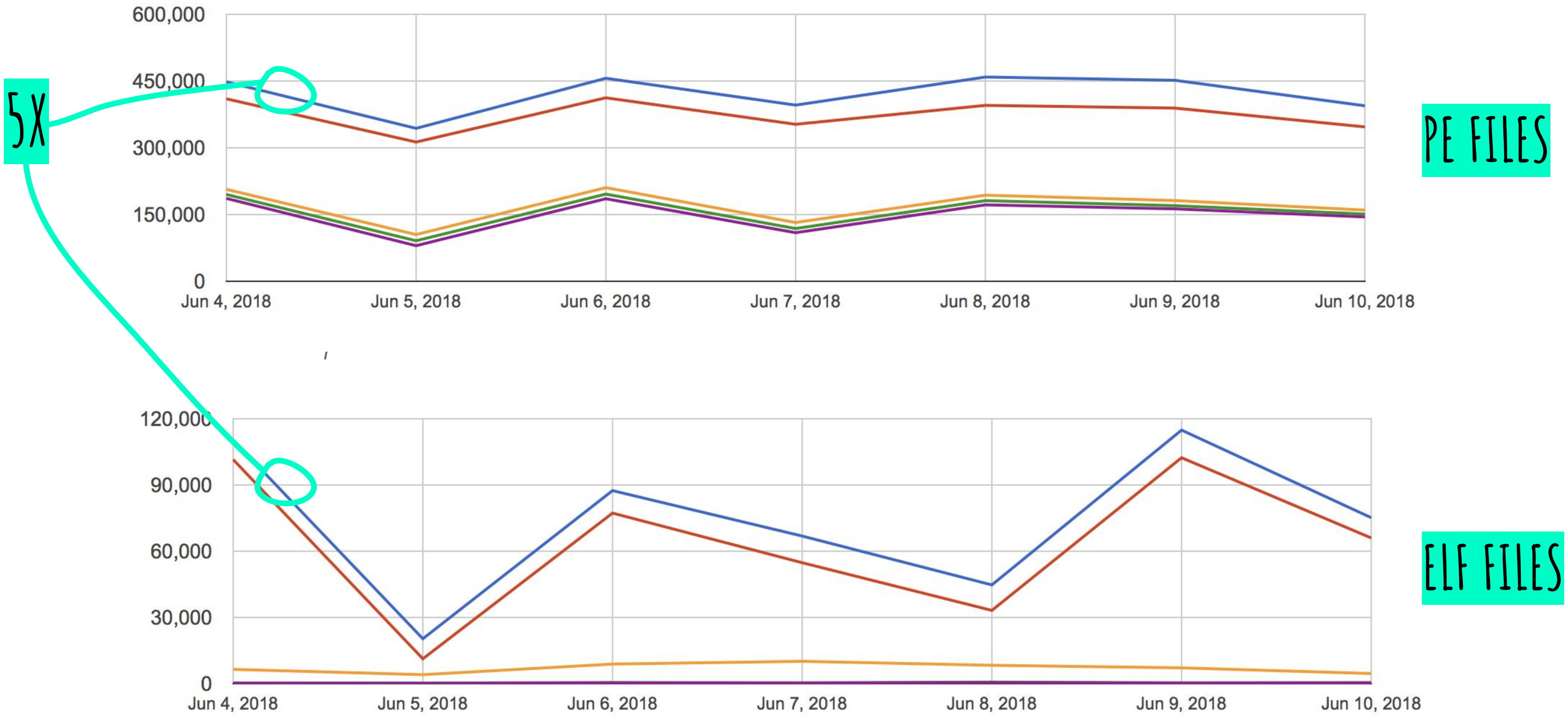
NEW FILES

PE FILES



ELF FILES

VT Submissions

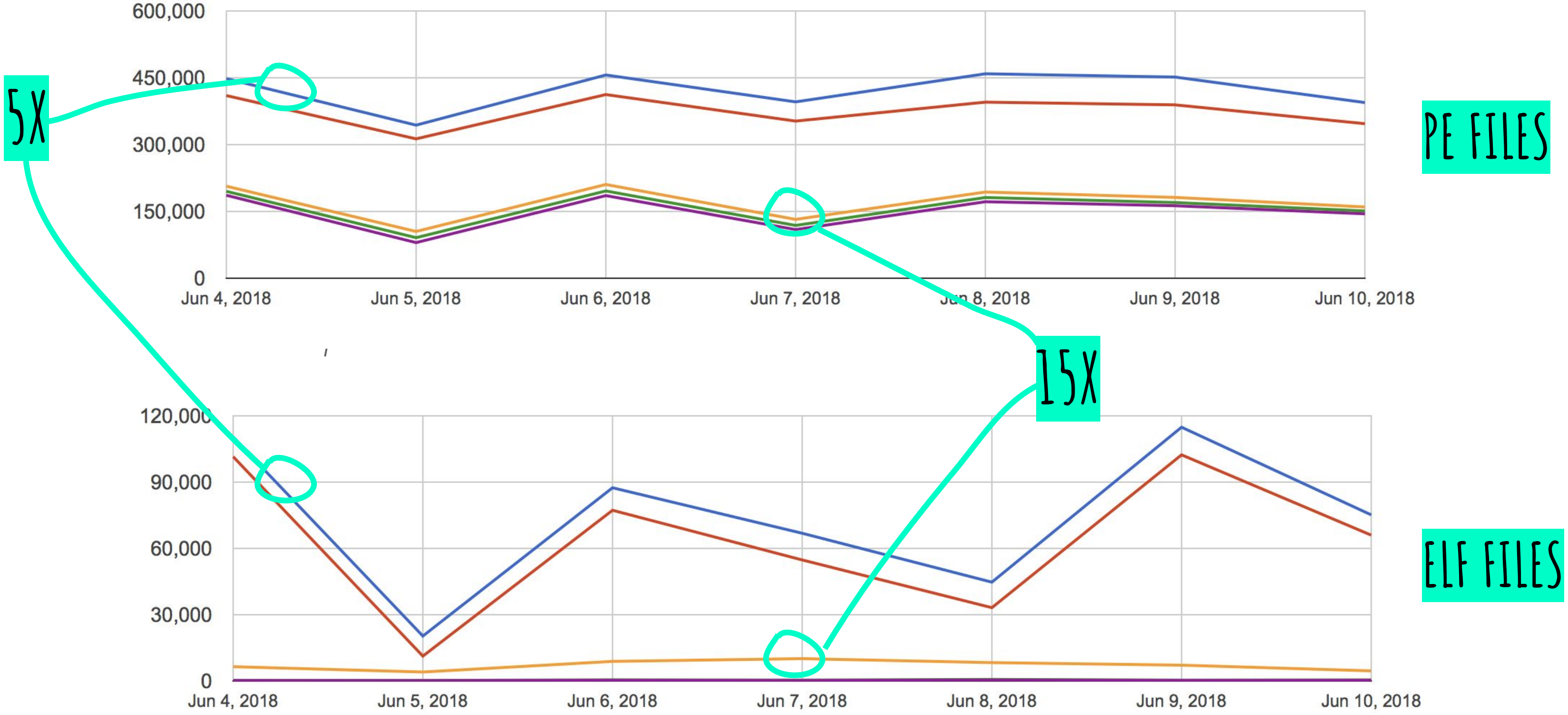


5X

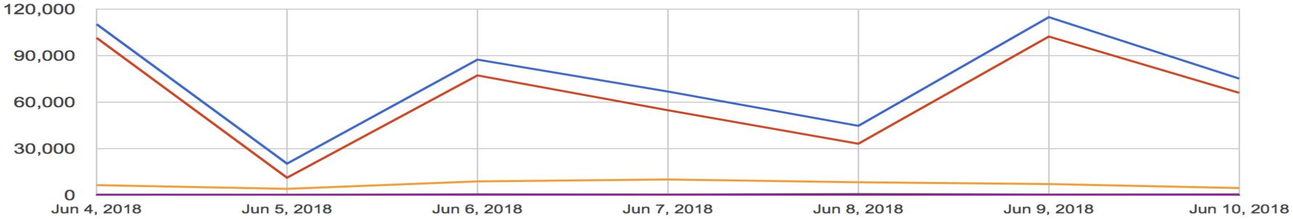
PE FILES

ELF FILES

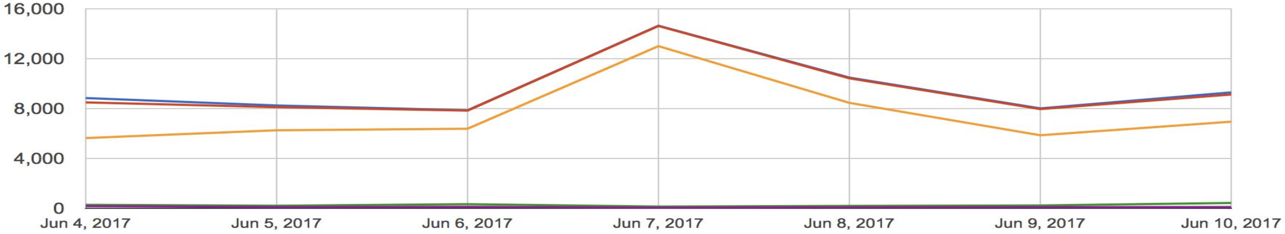
VT Submissions



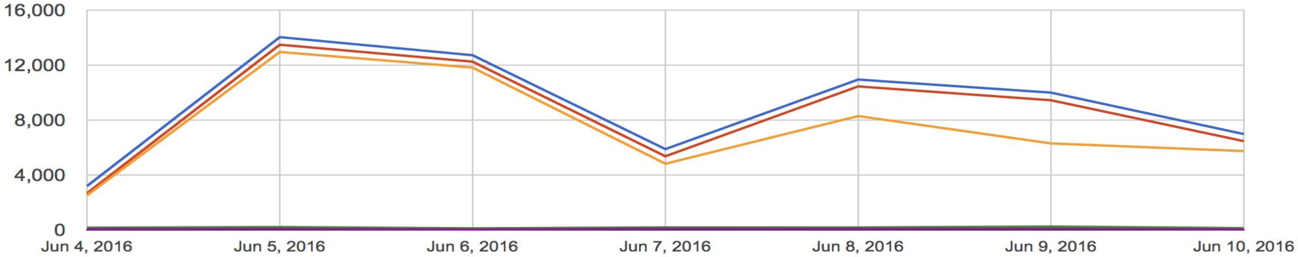
ELF OVER THE YEARS



2018

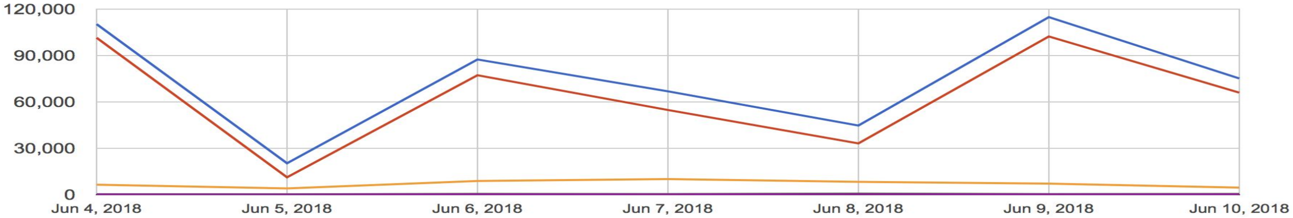


2017



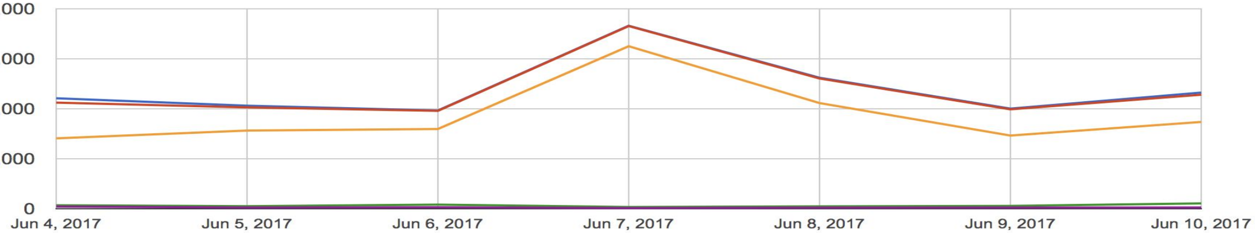
2016

ELF OVER THE YEARS

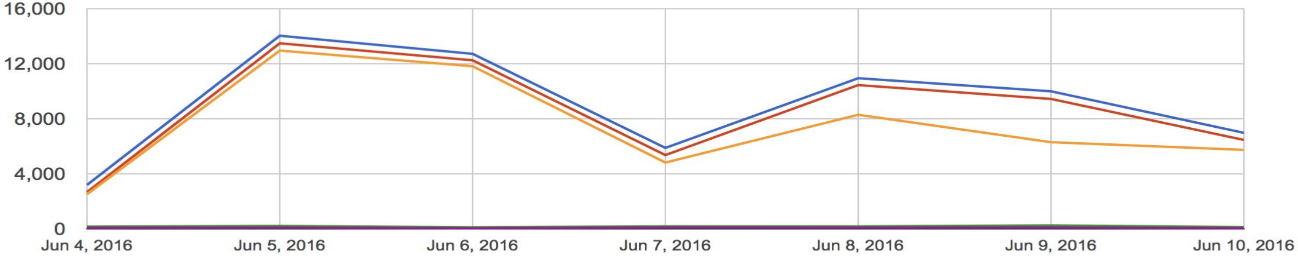


2018

9X



2017

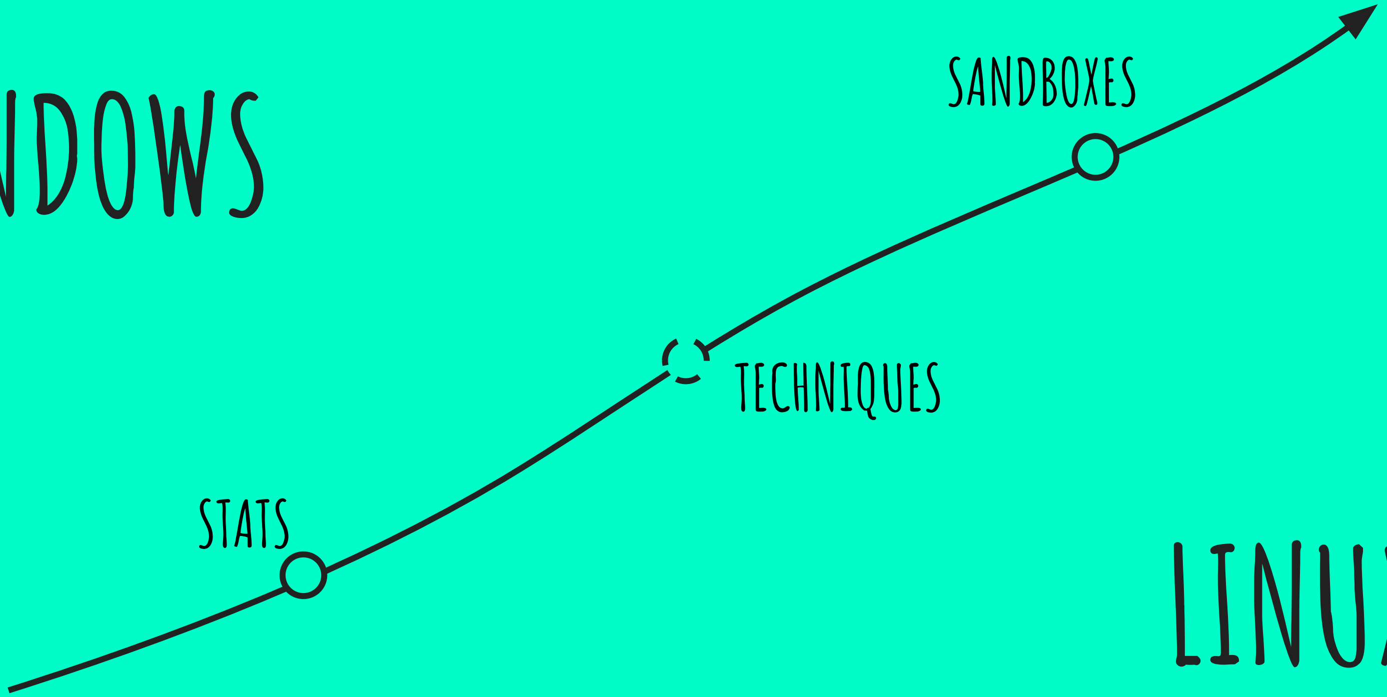


2016

DATASET

- Collected ELF samples for one year
- 200 candidate samples per day
- Final dataset: 10k ELF binaries

WINDOWS



LINUX

Packing

UPX

Y0da cryptor

FSG

NSPack

ASPack

Xtreme

TheMida

Armadillo

VMProtect

PECompact

PEtite

mPack

Obsidium

BackPack

UPolyX

ACProtect

UPX

ELFuck (sd)

DecryFile (@thegrugq)

BurnEye (scut)

Shiva (Mehta, Clowes)

Midgetpack (@aris_ada)

Maya (@ryan_elfmaster)

WINDOWS

LINUX

ELF Packers

NAME	SAMPLES	PERCENTAGE
VANILLA UPX	189	1.79%
CUSTOM UPX VARIANT:	188	1.78%
- DIFFERENT MAGIC	129	
- DIFF UPX STRINGS	55	
- JUNK BYTES	126	
- ALL OF THEM	16	

Rootkits

Rustock.C

Zeroaccess

Uroburos

FU

FUto

Sinowal

TLD3/4

Gapz

Carberp

adore-ng

mood-nt

enyelkm

override

reptile

suterusu

Vogl’s ROP rootkit

WINDOWS

LINUX

Injection techniques

CreateRemoteThread:

- LoadLibrary/WriteProcessMemory

Hijacking:

- Thread/COM

Process Hollowing

APC

SetWindowsHookEx

Atoms

Registry keys

LD_PRELOAD

ptrace

process_vm_writev

WINDOWS

LINUX

Anti-debugging

IsDebuggerPresent/IsDebugged

NtGlobalFlags

Debug registers

TLS callbacks

Heap

Trap flag

NtQueryInformationProcess

SEH/VEH

...

/proc/pid/status

ptrace

ENV (“_”)

WINDOWS

LINUX

Persistence

Windows registry:

  Run/RunOnce

  Winlogon

  AppInit_DLLs

  …

Browser Helper Objects

DLL Search Order Hijacking

...

cron

.bashrc

init.d

rc.d

systemd

X desktop autostart

WINDOWS

LINUX

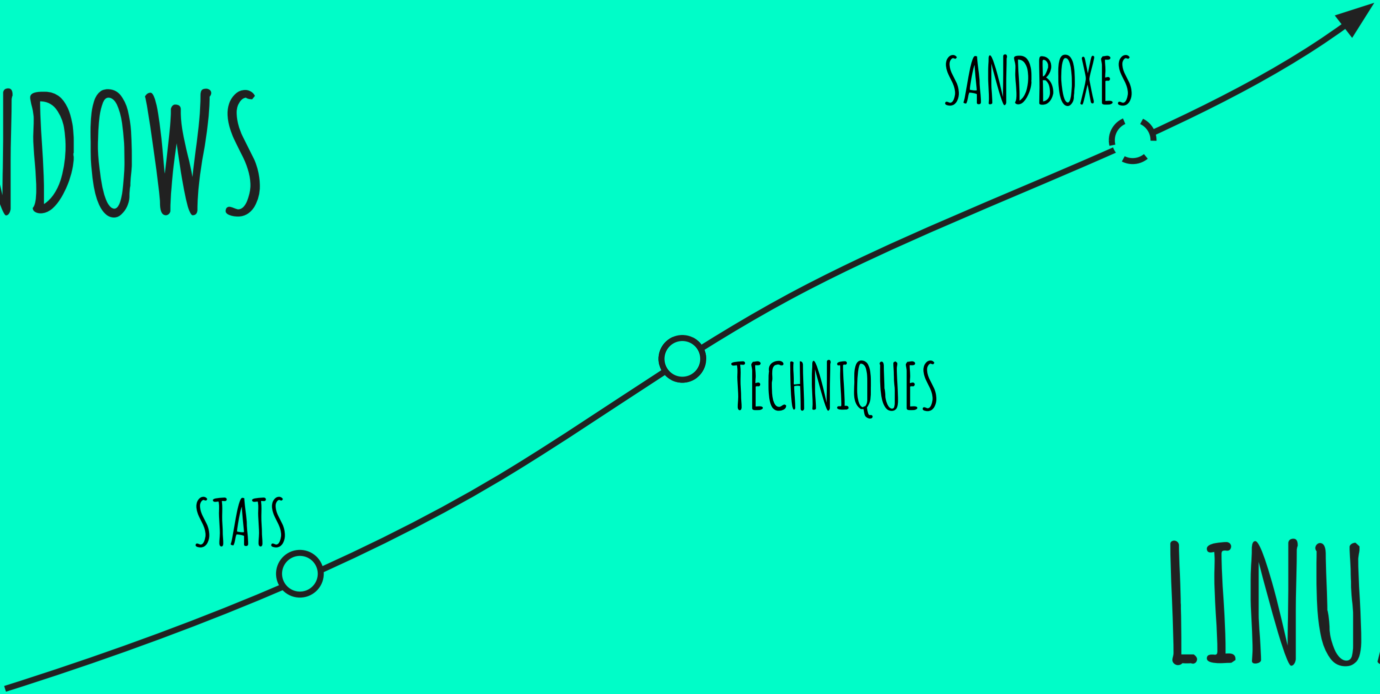
ELF Persistence

TECHNIQUE	USER	ROOT
/ETC/RC.D/RC.LOCAL	-	1393
/ETC/RC.CONF	-	1236
/ETC/INIT.D	-	210
/ETC/RCX.D	-	212
/ETC/RC.LOCAL	-	11
SYSTEMD SERVICE	-	2
~/.BASHRC	19	8
~/.BASHRC_PROFILE	18	8
X DESKTOP AUTOSTART	3	1
/ETC/CRON.HOURLY	-	70
/ETC/CRONTAB	-	70
/ETC/CRON.DAILY	-	26
CRONTAB UTILITY	6	6

ELF Persistence → 21.10%

TECHNIQUE	USER	ROOT
/ETC/RC.D/RC.LOCAL	-	1393
/ETC/RC.CONF	-	1236
/ETC/INIT.D	-	210
/ETC/RCX.D	-	212
/ETC/RC.LOCAL	-	11
SYSTEMD SERVICE	-	2
~/.BASHRC	19	8
~/.BASHRC_PROFILE	18	8
X DESKTOP AUTOSTART	3	1
/ETC/CRON.HOURLY	-	70
/ETC/CRONTAB	-	70
/ETC/CRON.DAILY	-	26
CRONTAB UTILITY	6	6

WINDOWS



STATS

TECHNIQUES

SANDBOXES

LINUX

Sandboxes

Malwr

~~Anubis~~

Cuckoo

ThreatExpert

Malbox

TotalHash

Joebox

ThreatGrid

Vicheck

DeepViz

anlyz.io

hybrid-analysis

VMRay

SecondWrite

ThreatTrack

Joebox (NEW!)

Limon

Cuckoo

hybrid-analysis (online)

detux (online)

Tencent Habo

WINDOWS

LINUX

Architectures

Architecture	Samples	Percentage
x86_64	3018	28.61%
MIPS	2120	20.10%
PowerPC	1569	14.87%
Motorola	1216	11.53%
Sparc	1170	11.09%
Intel 80386	720	6.83%
ARM 32-bit	555	5.26%
Hitachi SH	130	1.23%
AArch64	47	0.45%
Others	3	0.03%

ARCHITECTURES

ARCHITECTURE	SAMPLES	PERCENTAGE
x86_64	3018	28.61%
MIPS	2120	20.10%
POWERPC	1569	14.87%
MOTOROLA	1216	11.53%
SPARC	1170	11.09%
INTEL 80386	720	6.83%
ARM 32-BIT	555	5.26%
HITACHI SH	130	1.23%
AARCH64	47	0.45%
OTHERS	3	0.03%

35.44%

Architectures

Architecture	Samples	Percentage
x86_64	3018	28.61%
MIPS	2120	20.10%
PowerPC	1569	14.87%
Motorola	1216	11.53%
SPARC	1170	11.09%
Intel 80386	720	6.83%
ARM 32-bit	555	5.26%
Hitachi SH	130	1.23%
AArch64	47	0.45%
Others	3	0.03%

63.58%

35.44%

Architectures

Architecture	Samples	Percentage
x86_64	3018	28.61%
MIPS	2120	20.10%
PowerPC	1569	14.87%
Motorola	1216	11.53%
SPARC	1170	11.09%
Intel 80386	720	6.83%
ARM 32-bit	555	5.26%
Hitachi SH	130	1.23%
AArch64	47	0.45%
Others	3	0.03%

63.58%

70.41%

35.44%

Evasive samples

Evasion	Samples	Percentage
Process enumeration	259	3.32%
Anti-debugging	63	0.81%
Sandbox detection	19	0.24%
Anti-execution	3	0.04%
Stalling code	0	0%

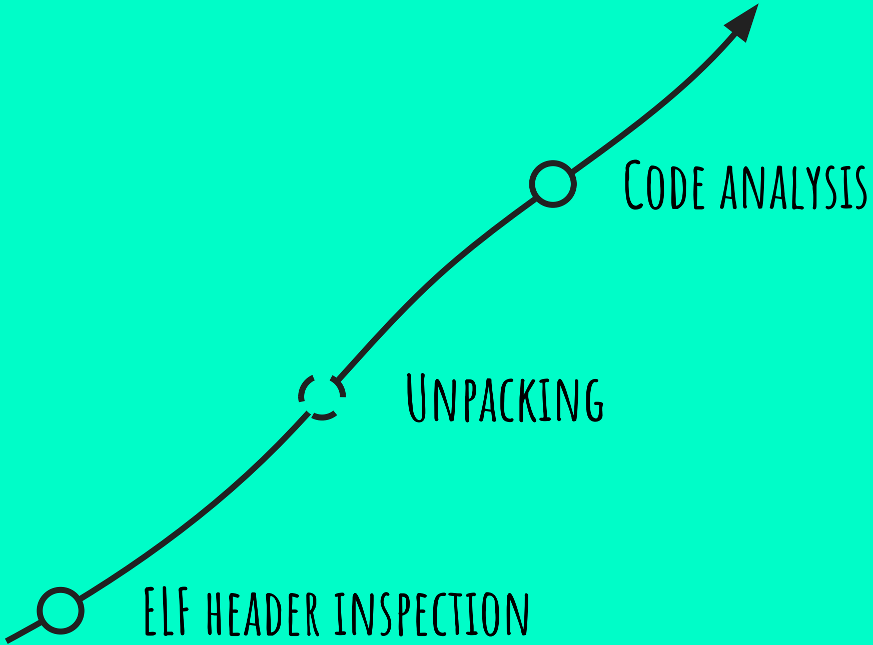
SANDBOX DETECTION

PATH	# SAMPLES
/SYS/CLASS/DMI/ID/PRODUCT_NAME	18
/SYS/CLASS/DMI/ID/SYS_VENDOR	18
/PROC/CPUINFO	1
/PROC/SYSINFO	1
/PROC/SCSI/SCSI	1
/PROC/VZ/ & /PROC/BC	1
/PROC/XEN/CAPABILITIES	1
/PROC/<PID>/MOUNTINFO	1

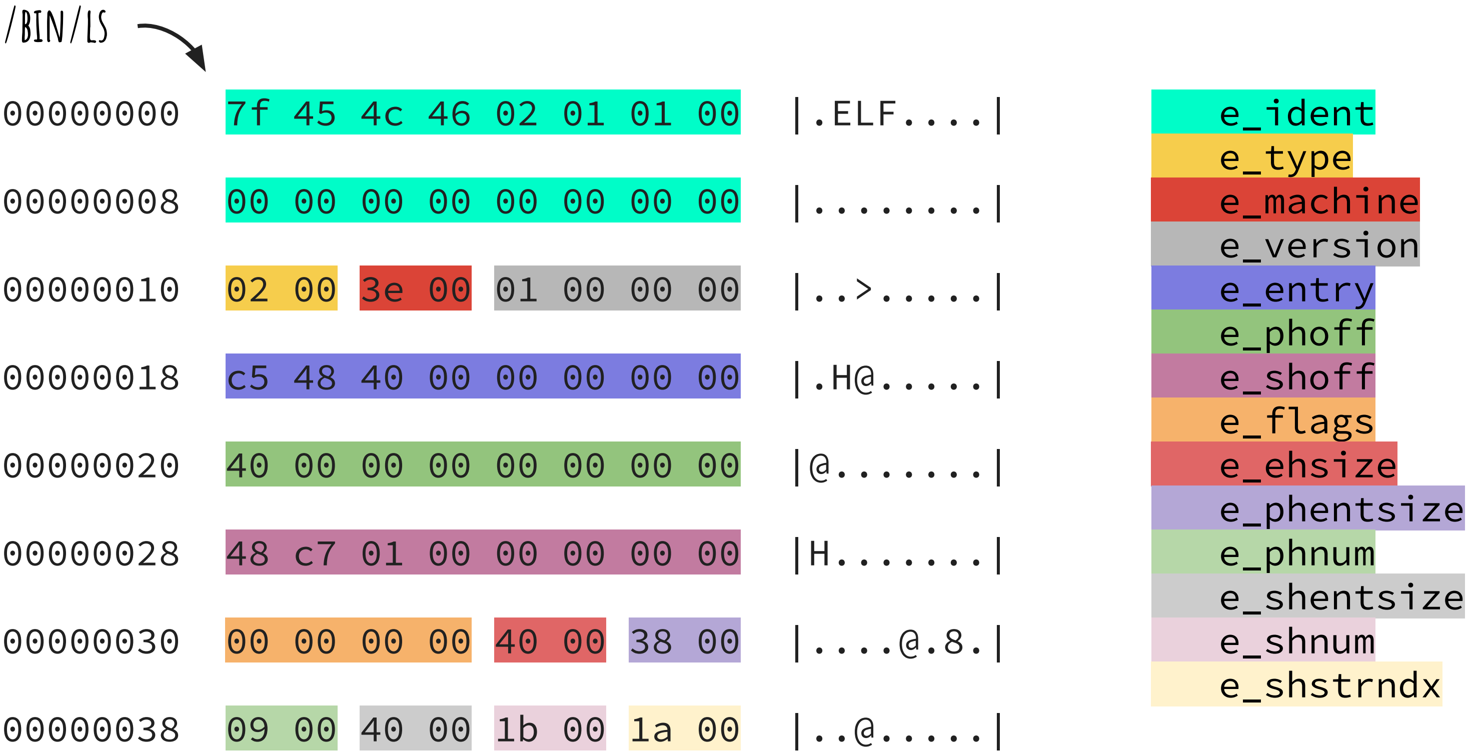
SANDBOX DETECTION

	PATH	# SAMPLES
VMWARE/VBOX	/SYS/CLASS/DMI/ID/PRODUCT_NAME	18
QEMU	/SYS/CLASS/DMI/ID/SYS_VENDOR	18
	/PROC/CPUINFO	1
KVM	/PROC/SYSINFO	1
	/PROC/SCSI/SCSI	1
OPENVZ	/PROC/VZ/ & /PROC/BC	1
	/PROC/XEN/CAPABILITIES	1
CHROOT JAIL	/PROC/<PID>/MOUNTINFO	1

Static Analysis



ELF HEADER



/BIN/LS

```
00000000  7f 45 4c 46 02 01 01 00  |.ELF....|
00000008  00 00 00 00 00 00 00 00  |........|
00000010  02 00 3e 00 01 00 00 00  |..>.....|
00000018  c5 48 40 00 00 00 00 00  |.H@.....|
00000020  40 00 00 00 00 00 00 00  |@.......|
00000028  48 c7 01 00 00 00 00 00  |H.......|
00000030  00 00 00 00 40 00 38 00  |....@.8.|
00000038  09 00 40 00 1b 00 1a 00  |..@.....|
```

- e_ident
- e_type
- e_machine
- e_version
- e_entry
- e_phoff
- e_shoff
- e_flags
- e_ehsize
- e_phentsize
- e_phnum
- e_shentsize
- e_shnum
- e_shstrndx

UNTRUSTABLE SECTIONS

/BIN/LS

```
00000000  7f 45 4c 46 02 01 01 00  |.ELF....|

00000008  00 00 00 00 00 00 00 00  |........|

00000010  02 00 3e 00 01 00 00 00  |..>.....|

00000018  c5 48 40 00 00 00 00 00  |.H@.....|

00000020  40 00 00 00 00 00 00 00  |@.......|

00000028  00 00 00 00 00 00 00 00  |........|

00000030  00 00 00 00 40 00 38 00  |....@.8.|

00000038  09 00 44 44 ff ff 00 00  |..DD....|
```

- Sections are useful for linking, relocation and debugging
- Not needed at run-time

Untrustable sections

/BIN/LS

```
00000000  7f 45 4c 46 02 01 01 00  |.ELF....|
00000008  00 00 00 00 00 00 00 00  |........|
00000010  02 00 3e 00 01 00 00 00  |..>.....|
00000018  c5 48 40 00 00 00 00 00  |.H@.....|
00000020  40 00 00 00 00 00 00 00  |@.......|
00000028  00 00 00 00 00 00 00 00  |........|
00000030  00 00 00 00 40 00 38 00  |....@.8.|
00000038  09 00 44 44 ff ff 00 00  |..DD....|
```

```
e_shoff       == 0 ||
e_shentsize   == 0 ||
e_shnum       == 0 ||
e_shstrndx    == 0 ||
ALL INVALID
```

- Sections are useful for linking, relocation and debugging
- Not needed at run-time

UNTRUSTABLE SECTIONS

/BIN/LS →

```
00000000  7f 45 4c 46 02 01 01 00  |.ELF....|
00000008  00 00 00 00 00 00 00 00  |........|
00000010  02 00 3e 00 01 00 00 00  |..>.....|
00000018  c5 48 40 00 00 00 00 00  |.H@.....|
00000020  40 00 00 00 00 00 00 00  |@.......|
00000028  00 00 00 00 00 00 00 00  |........|
00000030  00 00 00 00 40 00 38 00  |....@.8.|
00000038  09 00 44 44 ff ff 00 00  |..DD....|
```

```
e_shoff      == 0 ||
e_shentsize  == 0 ||
e_shnum      == 0 ||
e_shstrndx   == 0 ||
ALL INVALID
```

- Sections are useful for linking, relocation and debugging
- Not needed at run-time

DO NOT RELY ON THE SECTION HEADER TABLE

EXAMPLE - GDB

/BIN/LS

```
00000000  7f 45 4c 46 02 01 01 00  |.ELF....|
00000008  00 00 00 00 00 00 00 00  |........|
00000010  02 00 3e 00 01 00 00 00  |..>.....|
00000018  c5 48 40 00 00 00 00 00  |.H@.....|
00000020  40 00 00 00 00 00 00 00  |@.......|
00000028  48 c7 01 00 00 00 00 00  |H.......|
00000030  00 00 00 00 40 00 38 00  |....@.8.|
00000038  09 00 50 00 1b 00 1a 00  |..P.....|
```

e_shentsize != 0x40

EXAMPLE - GDB

/BIN/LS

```
00000000  7f 45 4c 46 02 01 01 00  |.ELF....|
00000008  00 00 00 00 00 00 00 00  |........|
00000010  02 00 3e 00 01 00 00 00  |..>.....|
00000018  c5 48 40 00 00 00 00 00  |.H@.....|
00000020  40 00 00 00 00 00 00 00  |@.......|
00000028  48 c7 01 00 00 00 00 00  |H.......|
00000030  00 00 00 00 40 00 38 00  |....@.8.|
00000038  09 00 50 00 1b 00 1a 00  |..P.....|
```

`e_shentsize  != 0x40`

```
$ gdb ./ls
"/home/aaa/ls": not in
executable format: File
format not recognized
(gdb) r
Starting program:
No executable file
specified.
(gdb) q
```

Malware - Mumblehard

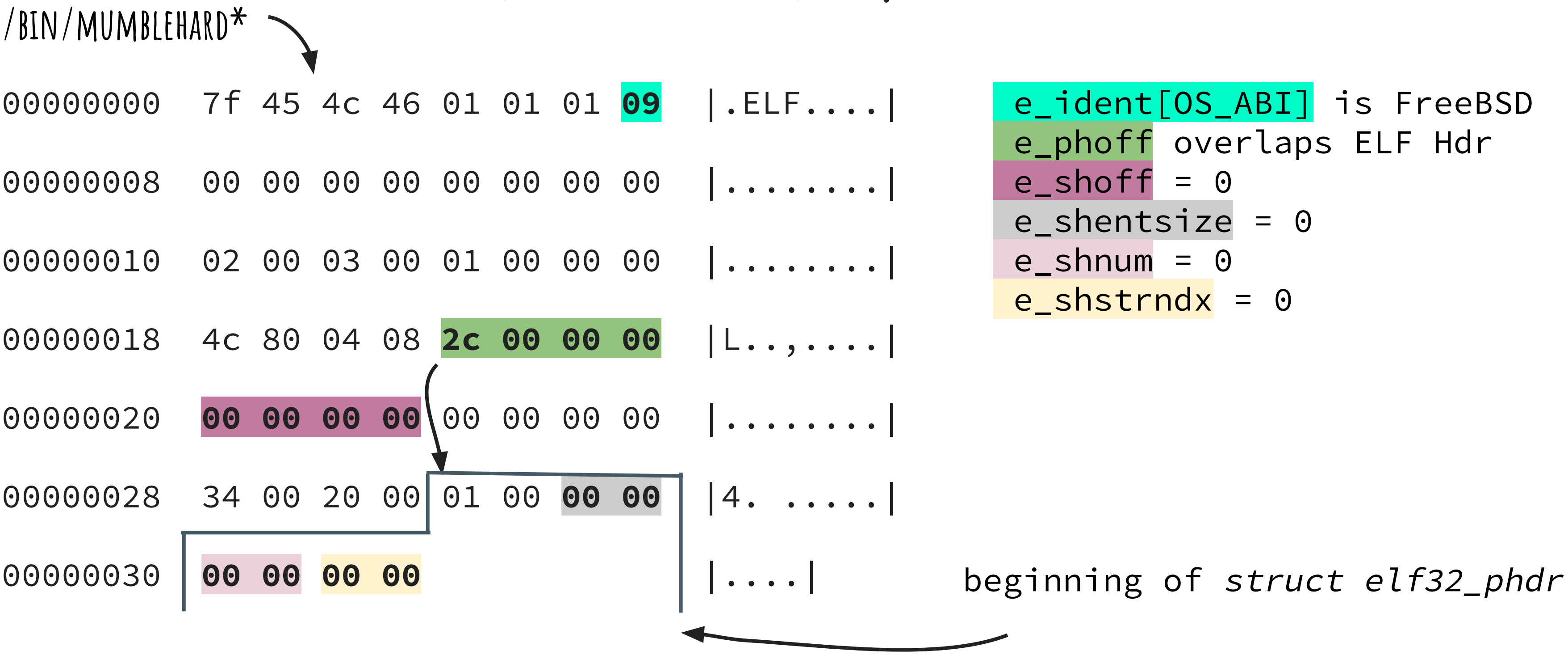
/bin/mumblehard*

```
00000000  7f 45 4c 46 01 01 01 09  |.ELF....|

00000008  00 00 00 00 00 00 00 00  |........|

00000010  02 00 03 00 01 00 00 00  |........|

00000018  4c 80 04 08 2c 00 00 00  |L..,....|

00000020  00 00 00 00 00 00 00 00  |........|

00000028  34 00 20 00 01 00 00 00  |4. .....|

00000030  00 00 00 00              |....|
```

e_ident[OS_ABI] is FreeBSD
e_phoff overlaps ELF Hdr
e_shoff = 0
e_shentsize = 0
e_shnum = 0
e_shstrndx = 0

SHA256: 2db2064458255a9e882392e3878fb19a6b0f3d779779cce77ae84835172d923

Malware - Mumblehard



/bin/mumblehard*

```
00000000  7f 45 4c 46 01 01 01 09  |.ELF....|
00000008  00 00 00 00 00 00 00 00  |........|
00000010  02 00 03 00 01 00 00 00  |........|
00000018  4c 80 04 08 2c 00 00 00  |L..,....|
00000020  00 00 00 00 00 00 00 00  |........|
00000028  34 00 20 00 01 00 00 00  |4. .....|
00000030  00 00 00 00              |....|
```

e_ident[OS_ABI] is FreeBSD

e_phoff overlaps ELF Hdr

e_shoff = 0

e_shentsize = 0

e_shnum = 0

e_shstrndx = 0

beginning of *struct elf32_phdr*

SHA256: 2db2064458255a9e882392e3878fb19a6b0f3d779779cce77ae84835172d923

E_IDENT[OS/ABI]

LINUX/V4.17/SOURCE/FS/BINFMT_ELF.C

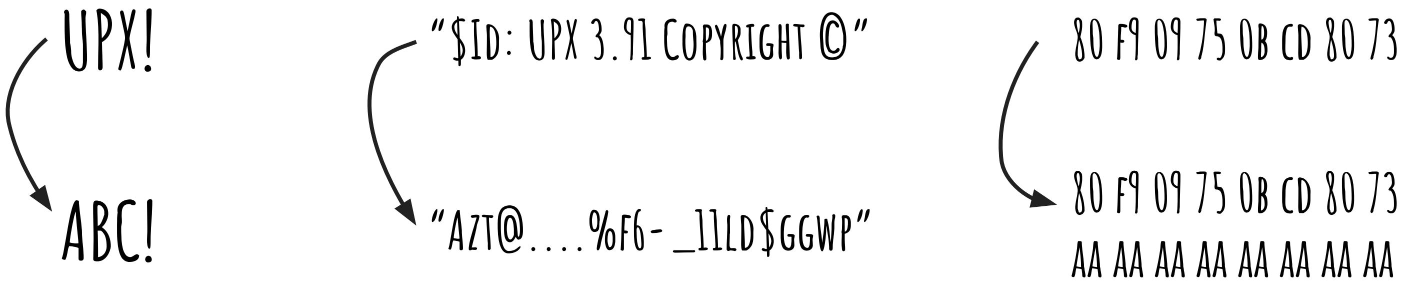
```
static int load_elf_binary(struct linux_binprm *bprm)
{
    ...

    /* Get the exec-header */
    loc->elf_ex = *((struct elfhdr *)bprm->buf);

    retval = -ENOEXEC;
    /* First of all, some simple consistency checks */
    if (memcmp(loc->elf_ex.e_ident, ELFMAG, SELFMAG) != 0)
        goto out;

    if (loc->elf_ex.e_type != ET_EXEC && loc->elf_ex.e_type != ET_DYN)
        goto out;
    if (!elf_check_arch(&loc->elf_ex))
        goto out;
    if (elf_check_fdpic(&loc->elf_ex))
        goto out;
```

CHECKS ON: MAGIC, TYPE, ARCH

OS/ABI NOT ENFORCED BY THE LINUX KERNEL

Executable unpacking

- Mostly underground and private packers on linux
- UPX is the top choice
    - binary mods to break "upx -d"
    - still easy to unpack manually



UPX!

ABC!

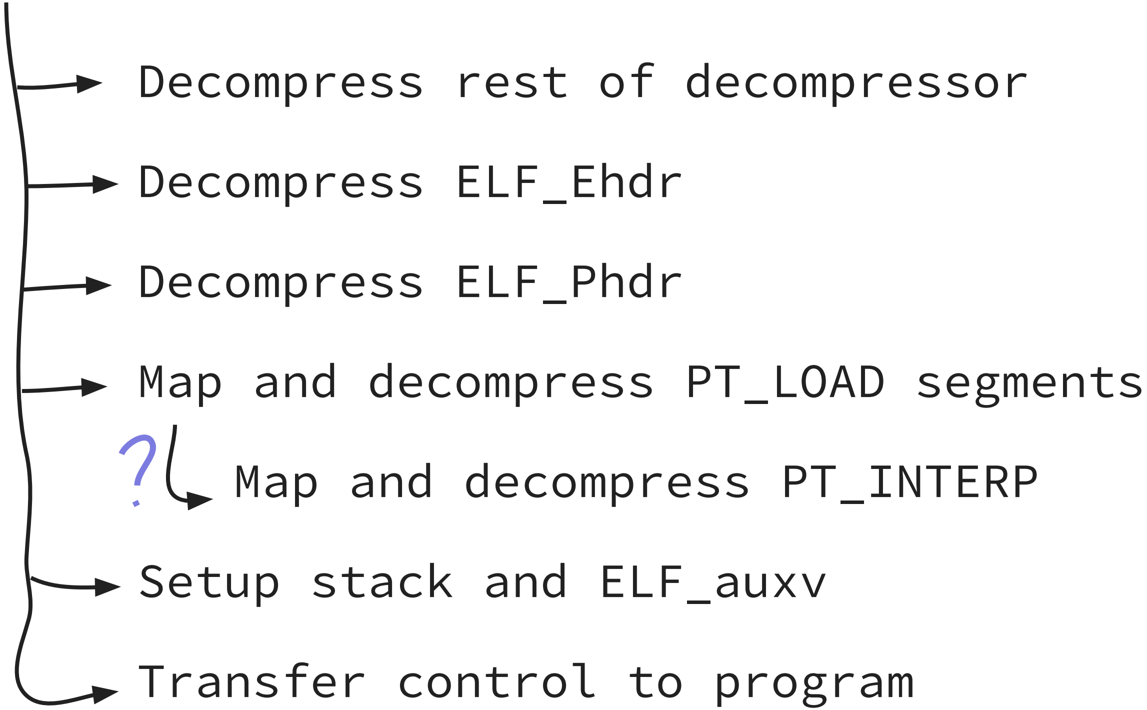
"$Id: UPX 3.91 Copyright ©"

"Azt@.....%f6-_11ld$ggwp"

80 f9 09 75 0b cd 80 73

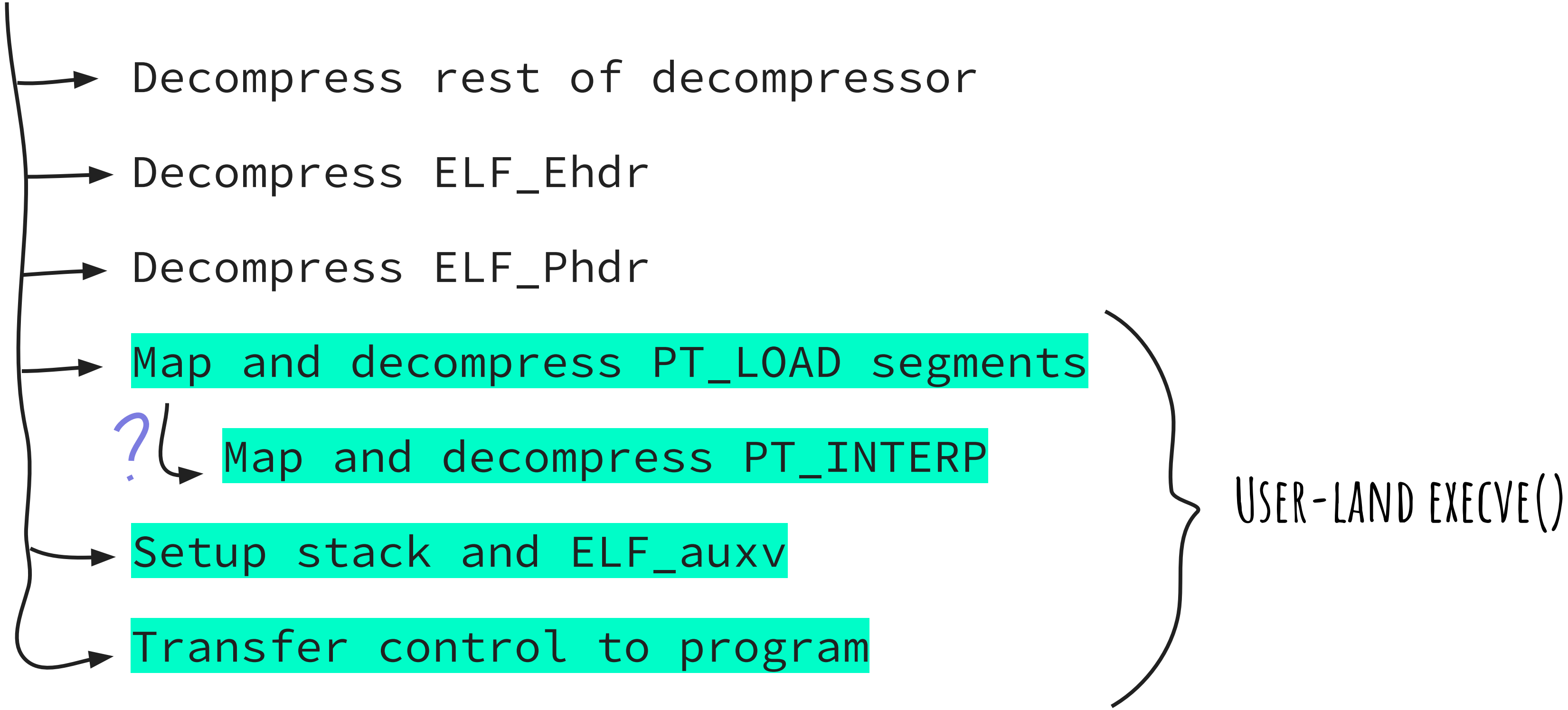
80 f9 09 75 0b cd 80 73
AA AA AA AA AA AA AA AA

UPX stub behavior



- Decompress rest of decompressor
- Decompress ELF_Ehdr
- Decompress ELF_Phdr
- Map and decompress PT_LOAD segments
  - ? Map and decompress PT_INTERP
- Setup stack and ELF_auxv
- Transfer control to program

UPX STUB BEHAVIOR



Decompress rest of decompressor

Decompress ELF_Ehdr

Decompress ELF_Phdr

Map and decompress PT_LOAD segments

Map and decompress PT_INTERP

Setup stack and ELF_auxv

Transfer control to program

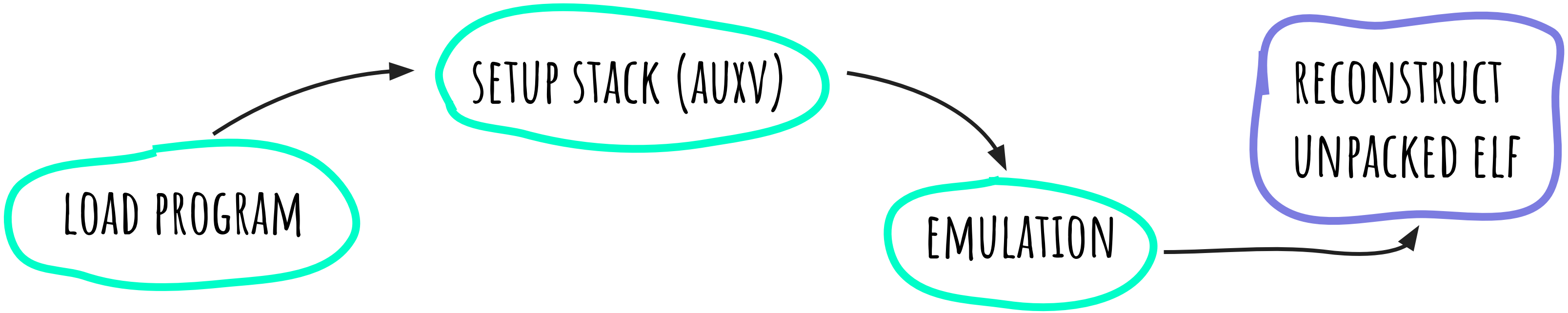
USER-LAND EXECVE()

UnPXer

- Based on Unicorn Engine
- Supports x86, x64, ARM, ARM-EABI, ARM64, MIPS
- Tiny kernel to run upx stub
  - `read, write, open, close, mmap, mprotect, munmap, brk, readlink, exit`

UnPXer

- Based on Unicorn Engine
- Supports x86, x64, ARM, ARM-EABI, ARM64, MIPS
- Tiny kernel to run UPX stub
  - read, write, open, close, mmap, mprotect, munmap, brk, readlink, exit



Function recognition

- crucial for human reverse engineering
- signature matching approach is not scalable

Function recognition

- crucial for human reverse engineering
- signature matching approach is not scalable

IDA Pro → recursive CF discovery

Function recognition

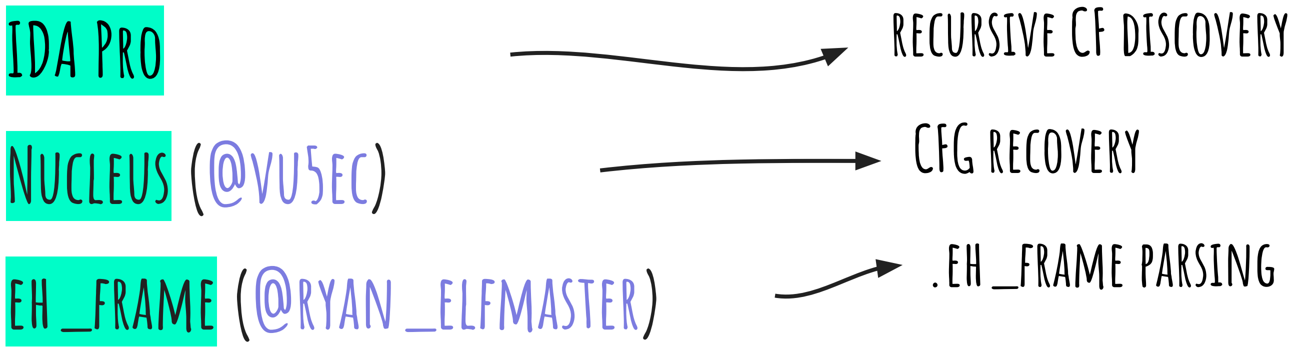
- crucial for human reverse engineering
- signature matching approach is not scalable

IDA Pro → recursive CF discovery

Nucleus (@vu5ec) → CFG recovery

Function recognition

- Crucial for human reverse engineering
- Signature matching approach is not scalable

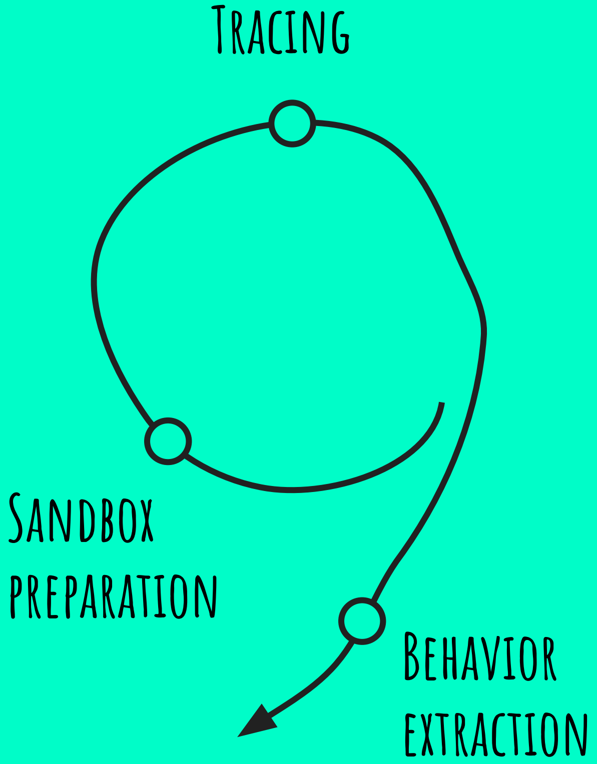


IDA Pro → recursive CF discovery

Nucleus (@vu5ec) → CFG recovery

eh_frame (@ryan_elfmaster) → .eh_frame parsing

Dynamic Analysis

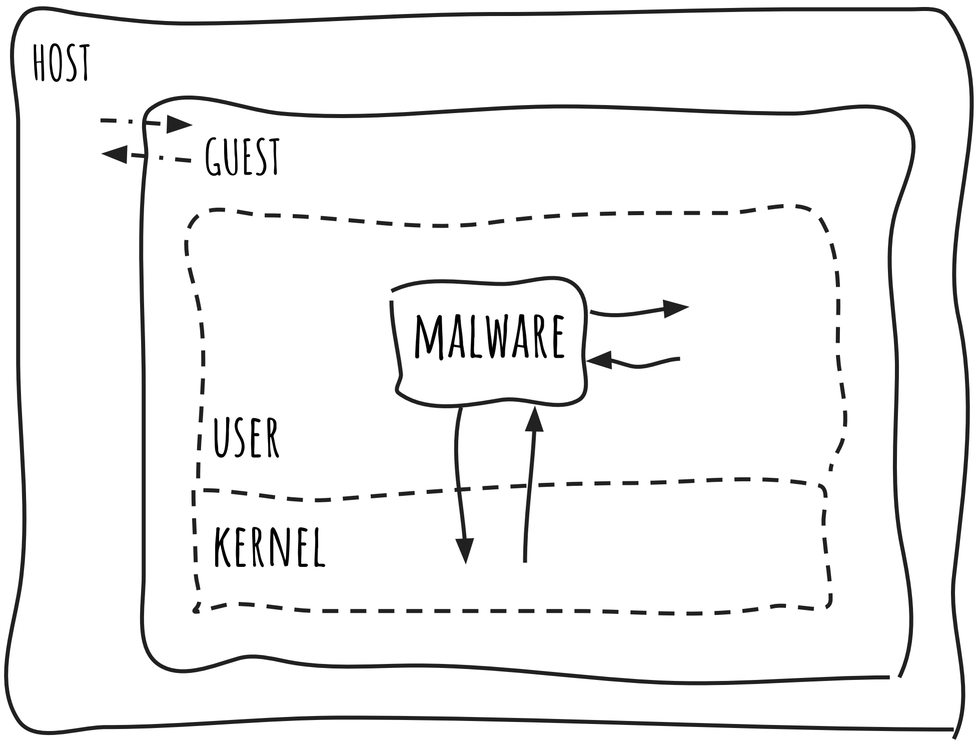


Tracing

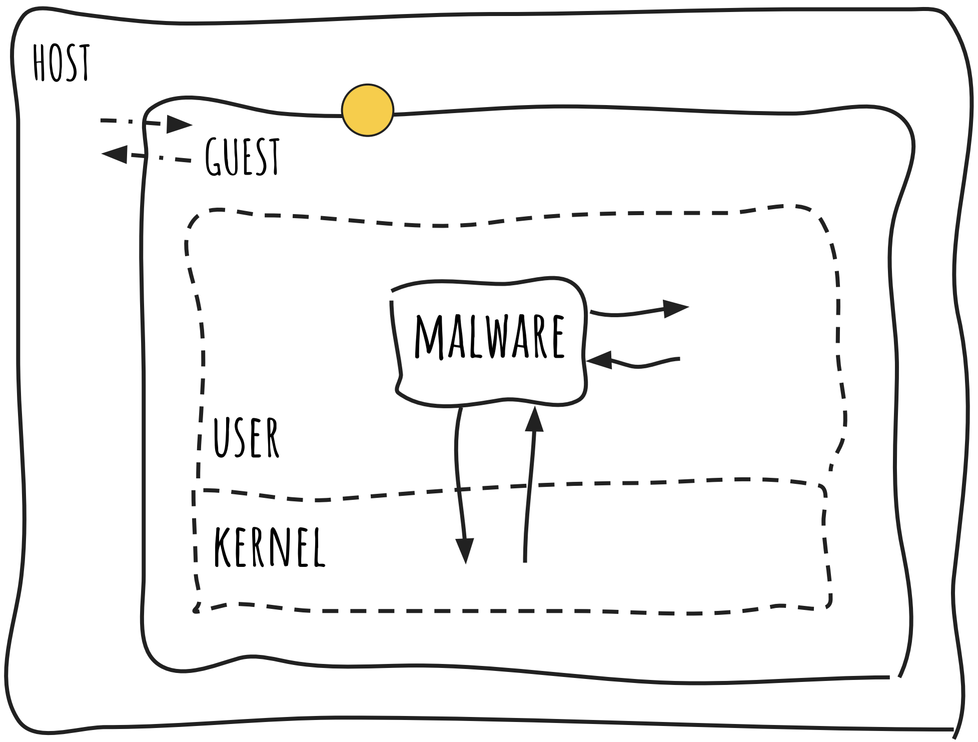
Sandbox preparation

Behavior extraction

Sandboxing linux malware

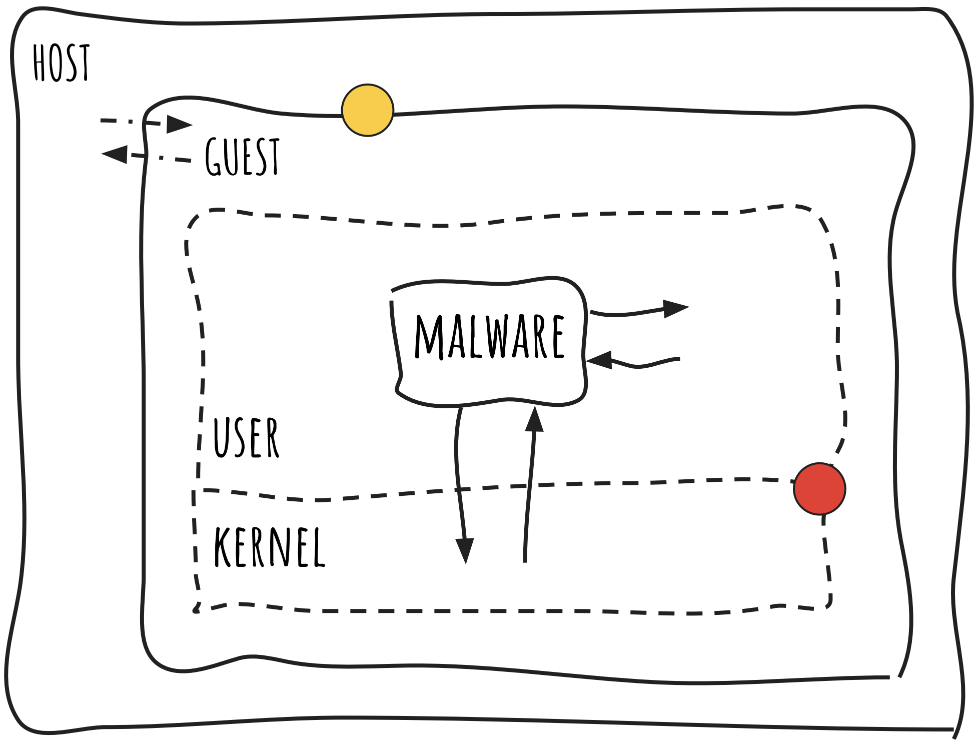


SANDBOXING LINUX MALWARE



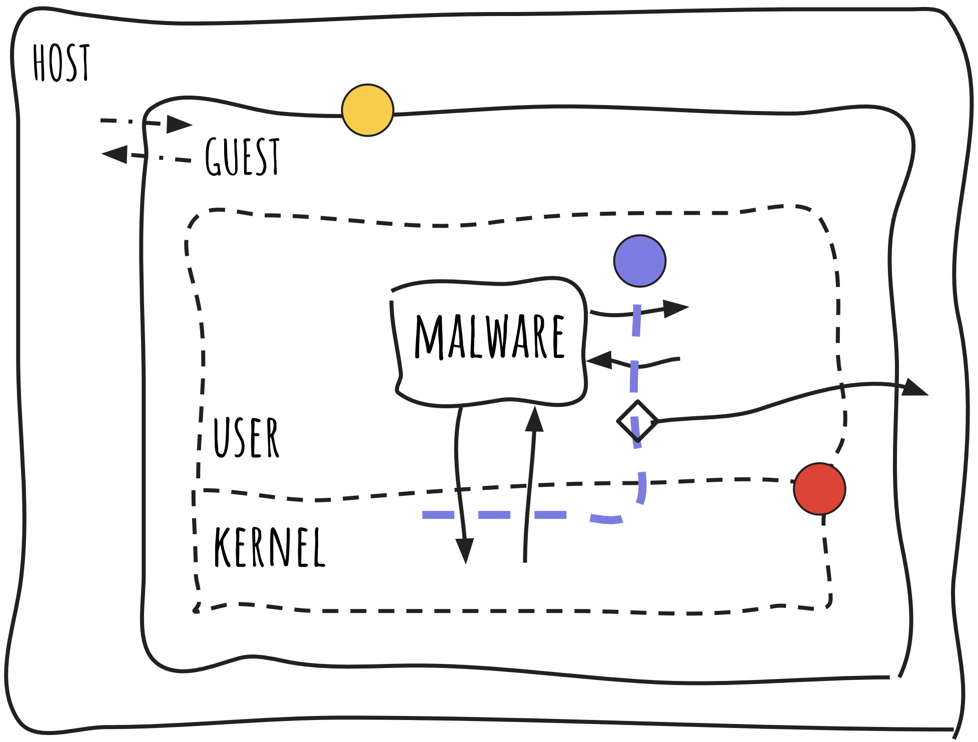
- QEMU TO SUPPORT DIFFERENT ARCHITECTURES AS MUCH AS POSSIBLE

Sandboxing linux malware



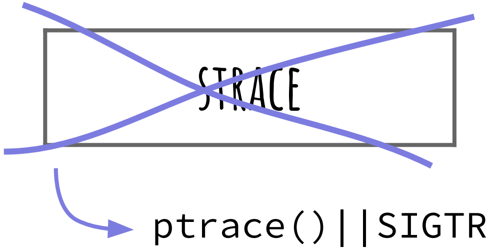
- 🟡 Qemu to support different architectures as much as possible
- 🔴 Full OS/environment

Sandboxing linux malware



- ● Qemu to support different architectures as much as possible
- ● Full OS/environment
- ● Syscalls and user functions tracing
- ● Behavioral report back to host

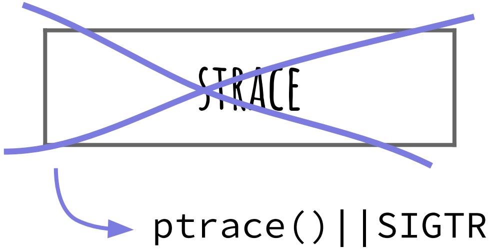
Linux tracing systems



~~strace~~

```
ptrace()||SIGTRAP||/proc/PID/cmdline||/proc/PID/status||...
```

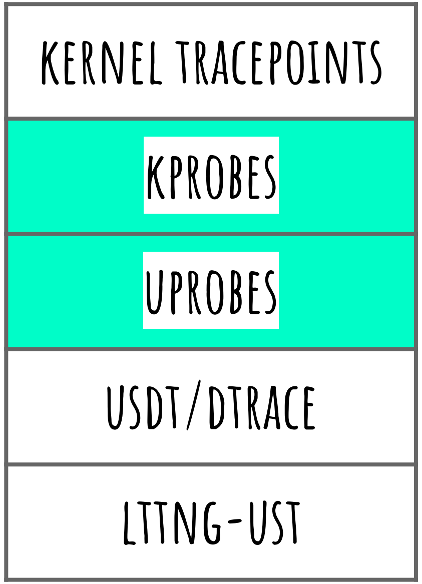
Linux tracing systems



~~STRACE~~

ptrace()||SIGTRAP||/proc/PID/cmdline||/proc/PID/status||...

BACKEND



KERNEL TRACEPOINTS

KPROBES

UPROBES

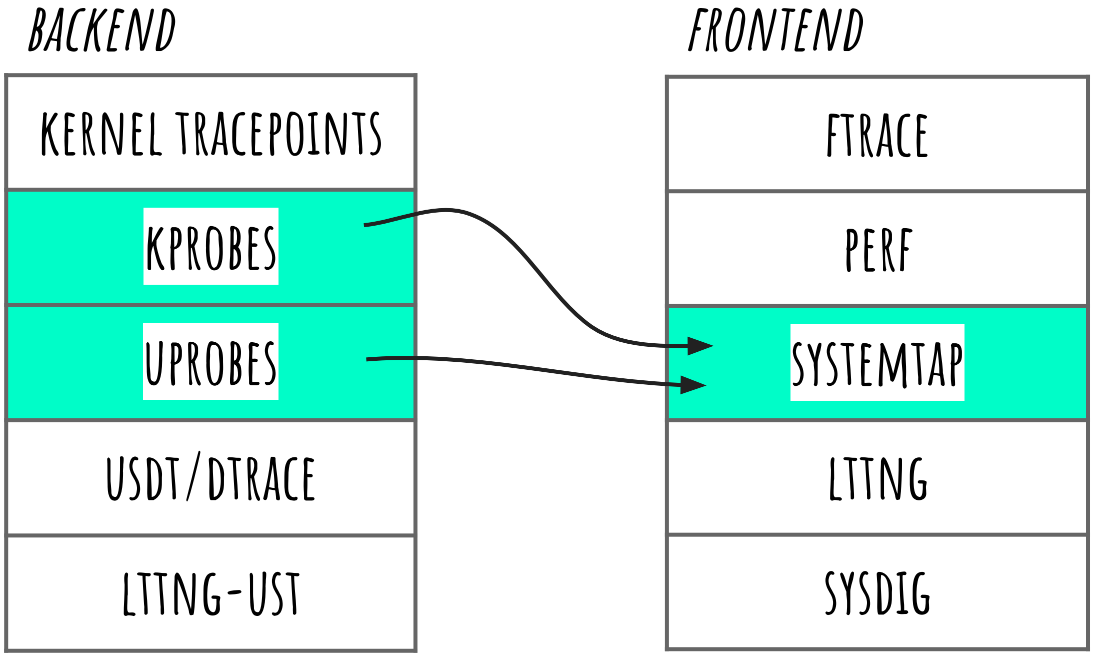
USDT/DTRACE

LTTNG-UST

Linux tracing systems

~~STRACE~~

ptrace()||SIGTRAP||/proc/PID/cmdline||/proc/PID/status||...



BACKEND

KERNEL TRACEPOINTS

KPROBES

UPROBES

USDT/DTRACE

LTTNG-UST

FRONTEND

FTRACE

PERF

SYSTEMTAP

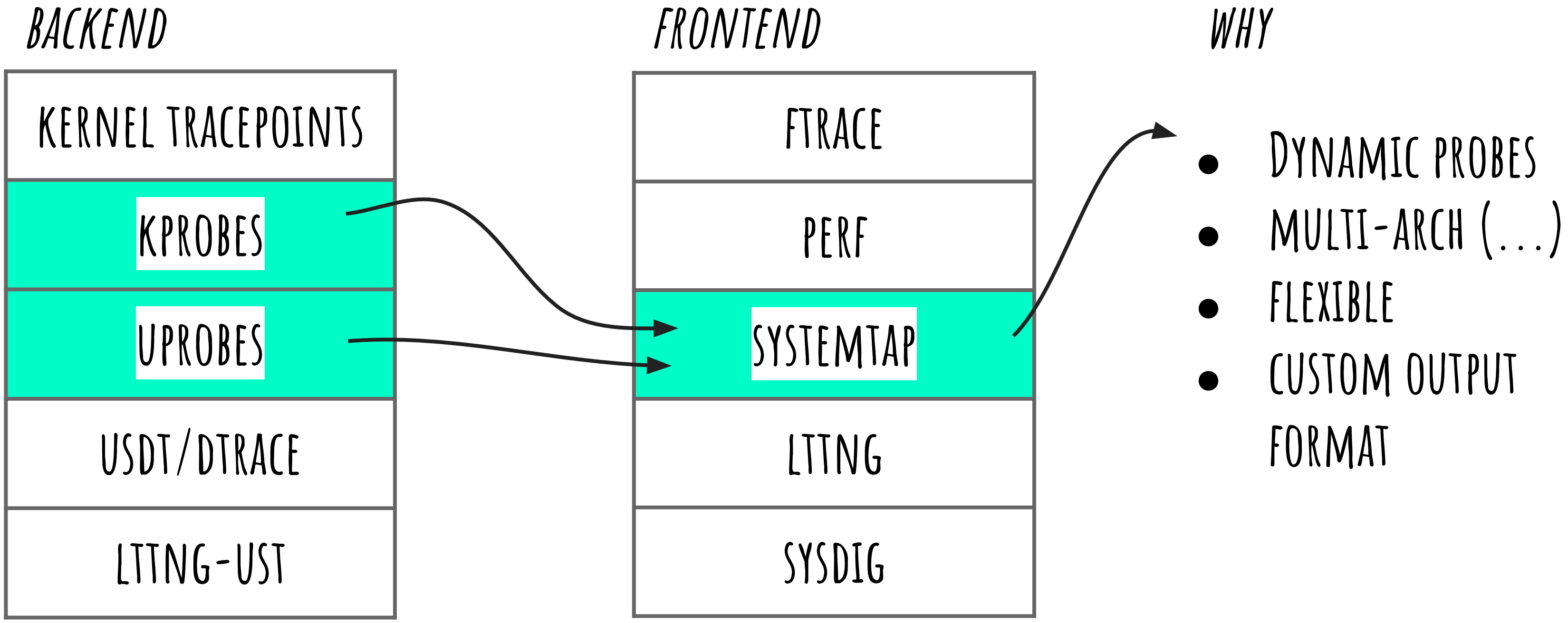
LTTNG

SYSDIG

Linux tracing systems

~~STRACE~~

ptrace()||SIGTRAP||/proc/PID/cmdline||/proc/PID/status||...



BACKEND

- KERNEL TRACEPOINTS
- KPROBES
- UPROBES
- USDT/DTRACE
- LTTNG-UST

FRONTEND

- FTRACE
- PERF
- SYSTEMTAP
- LTTNG
- SYSDIG

WHY

- DYNAMIC PROBES
- MULTI-ARCH (...)
- FLEXIBLE
- CUSTOM OUTPUT FORMAT

Kprobes -- Uprobes

1. COPY PROBED INSTRUCTION
2. REPLACE FIRST BYTE WITH *INT3*
3. ON HIT, EXECUTE PRE_HANDLER
4. SINGLE-STEP PROBED INSTRUCTION
5. EXECUTE POST_HANDLER

1. COPY PROBED INSTRUCTION
2. REPLACE FIRST BYTE WITH *INT3*
3. ON HIT, EXECUTE HANDLER
4. SINGLE-STEP PROBED INSTRUCTION

Kprobes -- Uprobes

1. COPY PROBED INSTRUCTION
2. REPLACE FIRST BYTE WITH *INT3*
3. ON HIT, EXECUTE PRE_HANDLER
4. SINGLE-STEP PROBED INSTRUCTION
5. EXECUTE POST_HANDLER

1. COPY PROBED INSTRUCTION
2. REPLACE FIRST BYTE WITH *INT3*
3. ON HIT, EXECUTE HANDLER
4. SINGLE-STEP PROBED INSTRUCTION

PROBE CREATION

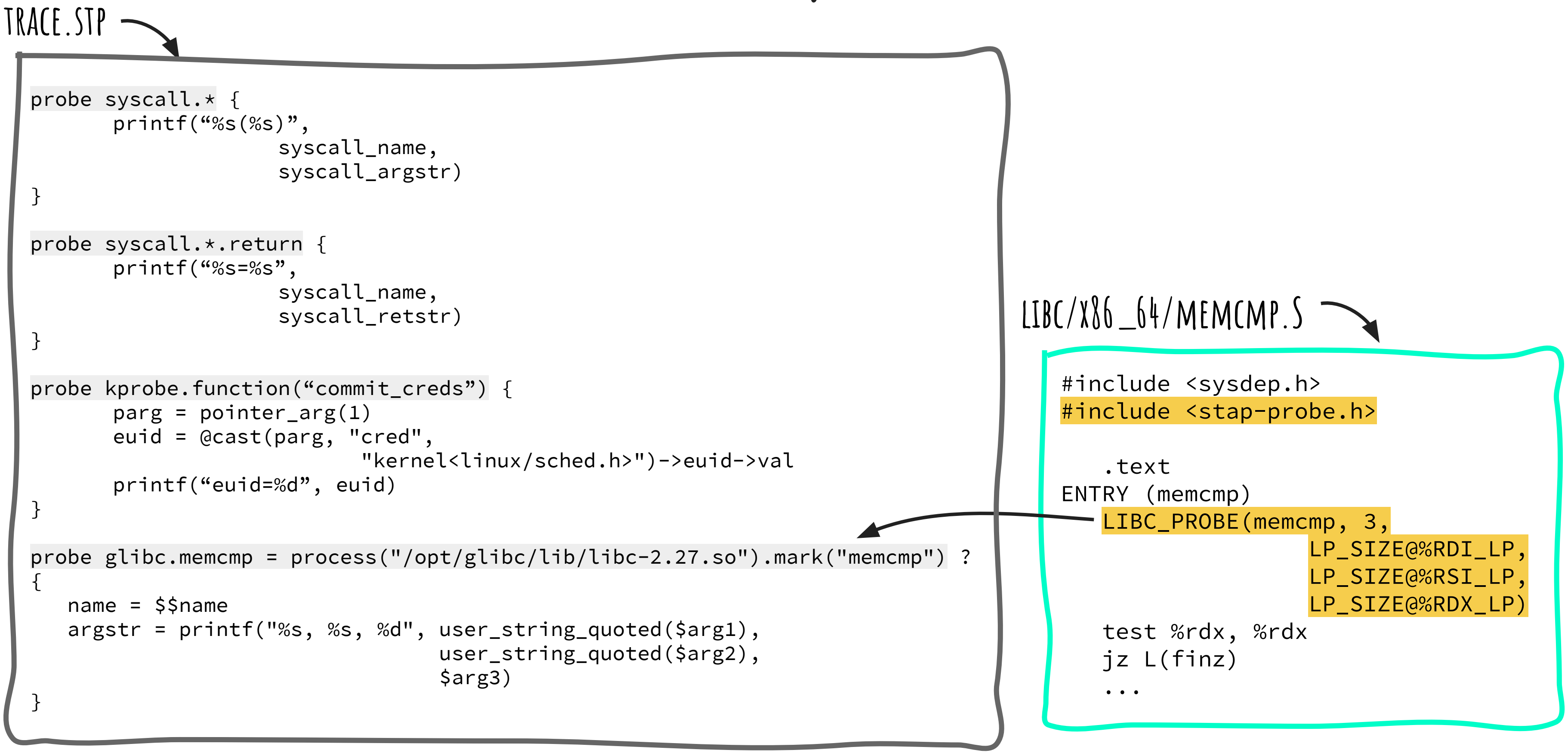
1. COMPILER EMITS *NOP* INSTRUCTION ON ADDRESS TO PROBE
2. ADD *NT_STAPSDT* DESCRIPTOR TO ELF FOR EACH ADDRESS TO PROBE

SystemTap

trace.stp

```
probe syscall.* {
       printf(“%s(%s)”,
                    syscall_name,
                    syscall_argstr)
}

probe syscall.*.return {
       printf(“%s=%s”,
                    syscall_name,
                    syscall_retstr)
}

probe kprobe.function(“commit_creds”) {
       parg = pointer_arg(1)
       euid = @cast(parg, "cred",
                           "kernel<linux/sched.h>")->euid->val
       printf(“euid=%d”, euid)
}

probe glibc.memcmp = process("/opt/glibc/lib/libc-2.27.so").mark("memcmp") ?
{
   name = $$name
   argstr = printf("%s, %s, %d", user_string_quoted($arg1),
                                 user_string_quoted($arg2),
                                 $arg3)
}
```

SystemTap



TRACE.STP

```
probe syscall.* {
      printf(“%s(%s)”,
                  syscall_name,
                  syscall_argstr)
}

probe syscall.*.return {
      printf(“%s=%s”,
                  syscall_name,
                  syscall_retstr)
}

probe kprobe.function(“commit_creds”) {
      parg = pointer_arg(1)
      euid = @cast(parg, "cred",
                   "kernel<linux/sched.h>")->euid->val
      printf(“euid=%d”, euid)
}

probe glibc.memcmp = process("/opt/glibc/lib/libc-2.27.so").mark("memcmp") ?
{
   name = $$name
   argstr = printf("%s, %s, %d", user_string_quoted($arg1),
                                 user_string_quoted($arg2),
                                 $arg3)
}
```

LIBC/X86_64/MEMCMP.S

```
#include <sysdep.h>
#include <stap-probe.h>

   .text
ENTRY (memcmp)
   LIBC_PROBE(memcmp, 3,
                  LP_SIZE@%RDI_LP,
                  LP_SIZE@%RSI_LP,
                  LP_SIZE@%RDX_LP)
   test %rdx, %rdx
   jz L(finz)
   ...
```

SYSTEMTAP

TRACE.STP

```
probe syscall.* {
        printf(“%s(%s)”,
                        syscall_name,
                        syscall_argstr)
}

probe syscall.*.return {
        printf(“%s=%s”,
                        syscall_name,
                        syscall_retstr)
}

probe kprobe.function(“commit_creds”) {
        parg = pointer_arg(1)
        euid = @cast(parg, "cred",
                                "kernel<linux/sched.h>")->euid->val
        printf(“euid=%d”, euid)
}

probe glibc.memcmp = process("/opt/glibc/lib/libc-2.27.so").mark("memcmp") ?
{
    name = $$name
    argstr = printf("%s, %s, %d", user_string_quoted($arg1),
                                  user_string_quoted($arg2),
                                  $arg3)
}
```

WORKS OUT-OF-THE-BOX ON I386/X86_64

SYSTEMTAP PATCHES NEEDED FOR OTHER ARCHS

(SYSCALL ABI SUPPORT, ARM, MIPS O32,...)

LIBC/X86_64/MEMCMP.S

```
#include <sysdep.h>
#include <stap-probe.h>

    .text
ENTRY (memcmp)
    LIBC_PROBE(memcmp, 3,
                    LP_SIZE@%RDI_LP,
                    LP_SIZE@%RSI_LP,
                    LP_SIZE@%RDX_LP)
    test %rdx, %rdx
    jz L(finz)
    ...
```

Malware - Amnesia

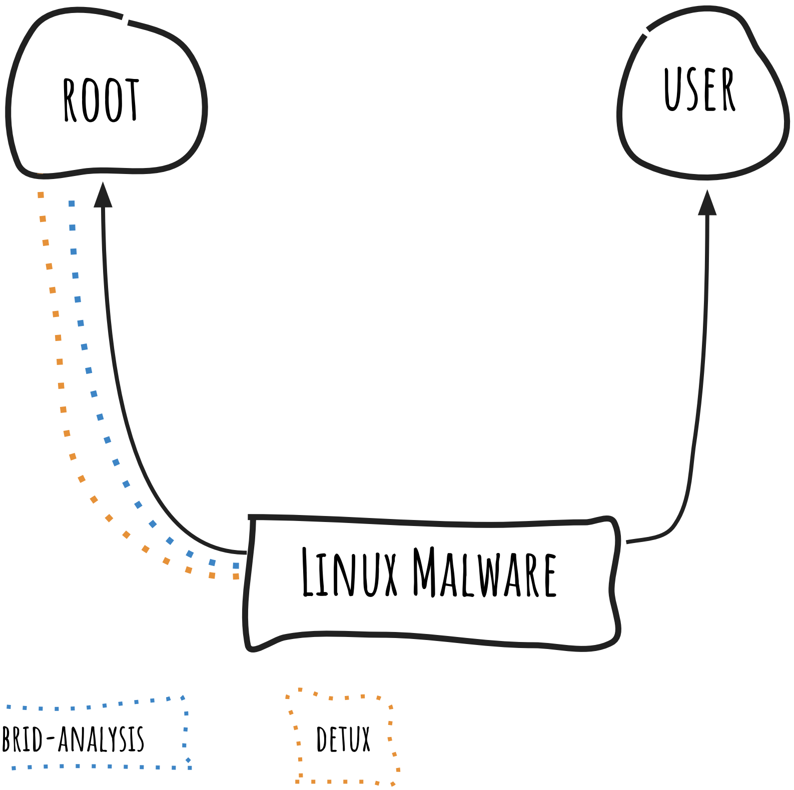
/BIN/AMNESIA*

EXECUTION TRACE WITH KPROBES AND UPROBES

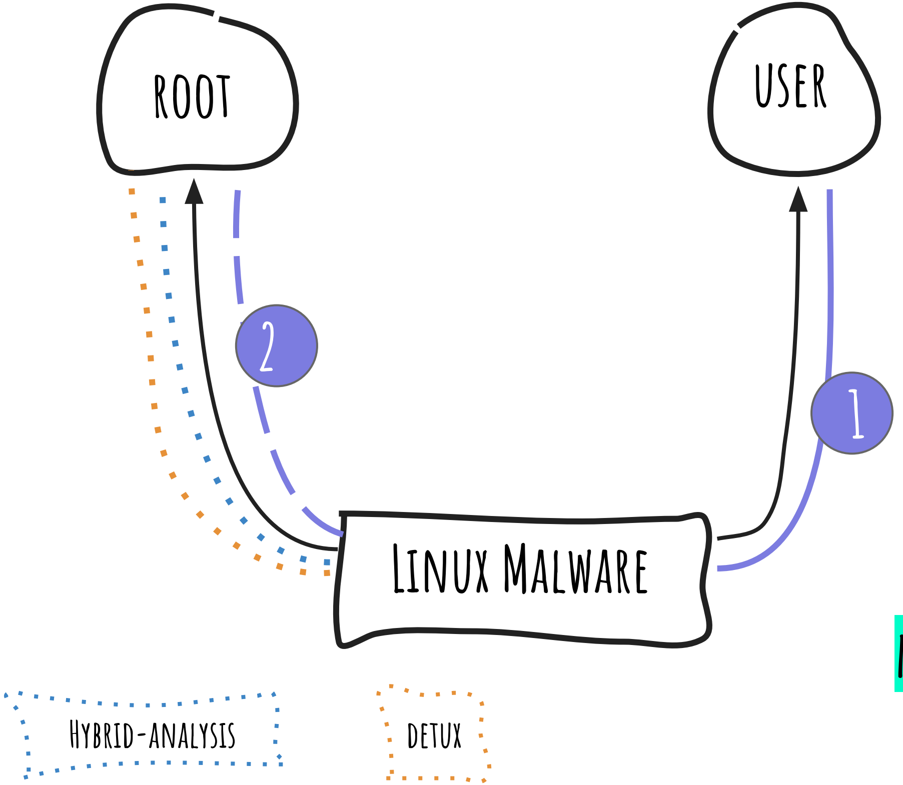
```
...
6b2885a4f8c9d84@0xb7747c31[911-1001] brk(0x0) = 146305024
6b2885a4f8c9d84@0xb7747c31[911-1001] brk(0x8ba8000) = 146440192
6b2885a4f8c9d84@0xb7747c31[911-1001] open("/sys/class/dmi/id/product_name", O_RDONLY) = 3
6b2885a4f8c9d84@0xb7747c31[911-1001] fstat(3, 0xbfafa800) = 0
6b2885a4f8c9d84@0xb7747c31[911-1001] read(3, 0x8b87168, 4096) = 34
6b2885a4f8c9d84@0xb75fbb7e[911-1001] strstr("Standard PC (i440FX + PIIX, 1996)\n", "VirtualBox") = 0
6b2885a4f8c9d84@0xb75fbb7e[911-1001] strstr("Standard PC (i440FX + PIIX, 1996)\n", "VMware") = 0
6b2885a4f8c9d84@0xb7747c31[911-1001] read(3, 0x8b87168, 4096) = 0
6b2885a4f8c9d84@0xb7747c31[911-1001] close(3) = 0
6b2885a4f8c9d84@0xb7747c31[911-1001] open("/sys/class/dmi/id/sys_vendor", O_RDONLY) = 3
6b2885a4f8c9d84@0xb7747c31[911-1001] fstat(3, 0xbfafa800) = 0
6b2885a4f8c9d84@0xb7747c31[911-1001] read(3, 0x8b87168, 4096) = 5
6b2885a4f8c9d84@0xb75fbb7e[911-1001] strstr("QEMU\nard PC (i440FX + PIIX, 1996)\n", "QEMU") = 0x8b87168
6b2885a4f8c9d84@0xb7747c31[911-1001] fstat(1, 0xbfafa800) = 0
6b2885a4f8c9d84@0xb7747c31[911-1001] write(1, "https://lmgtfy.com/?q=how+to+suck+your+own+di"..., 48) = 48
6b2885a4f8c9d84@0xb7747c31[911-1001] lstat("/tmp", 0xbfafa8d0) = 0
...
```

SHA256: 6b2885a4f8c9d84e5dc49830abf7b1edbf1b458d8b9d2bafb680370106f93bc3

Execution privilege



Execution privilege

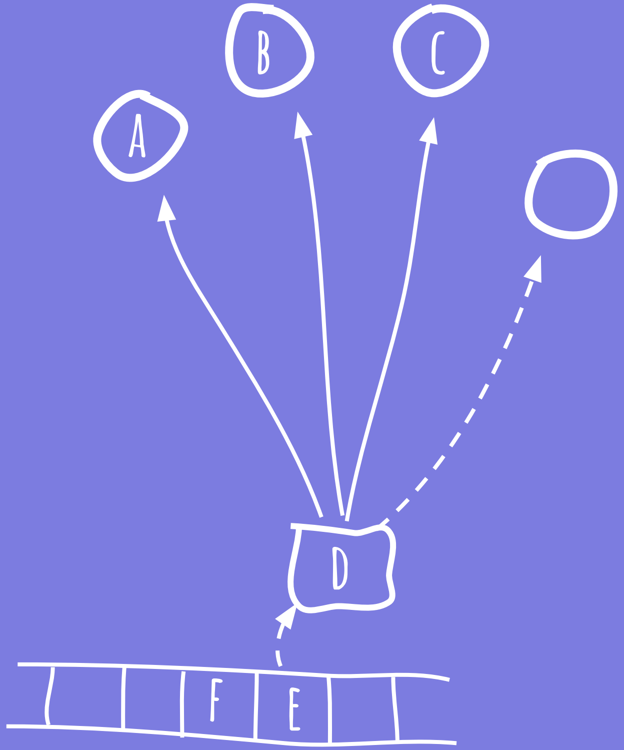


1. USER EXECUTION
   IF -EPERM OR -EACCES GOTO2
   IF CHECK UID OR GID GOTO2
   ELSE FINISH
2. ROOT EXECUTION

MORE RESOURCES BUT MAY SHOW OFF NEW BEHAVIORS

AUTOMATED ANALYSIS WITH

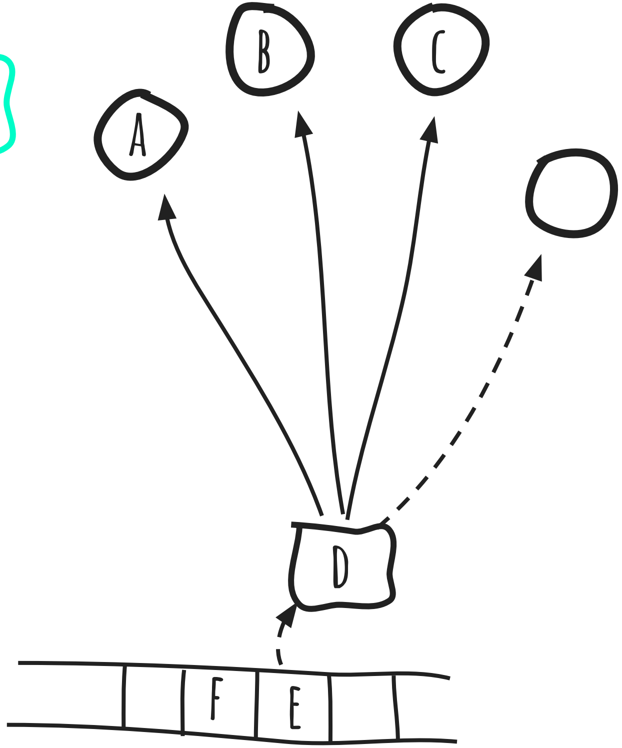
PADAWAN



Padawan

Framework for parallel data processing and data visualization

RE few samples is affordable - *thousands* would be a nightmare

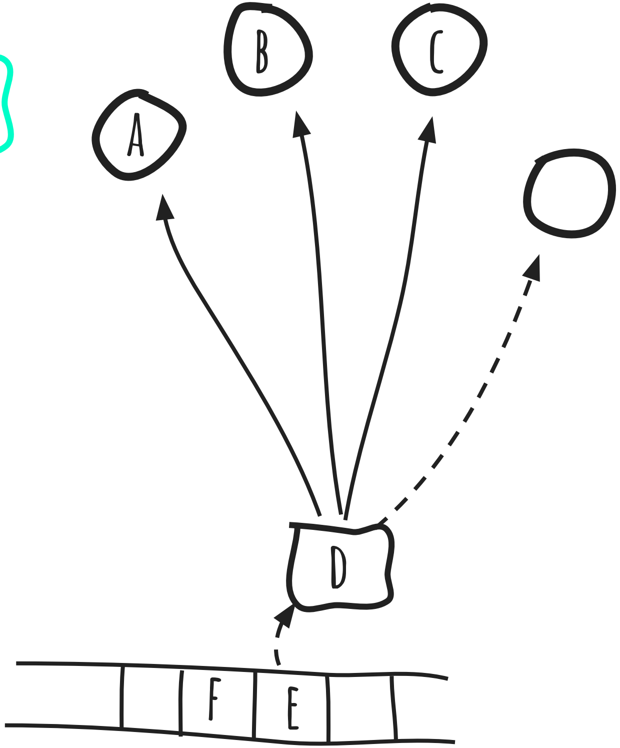


Padawan

Framework for parallel data processing and data visualization

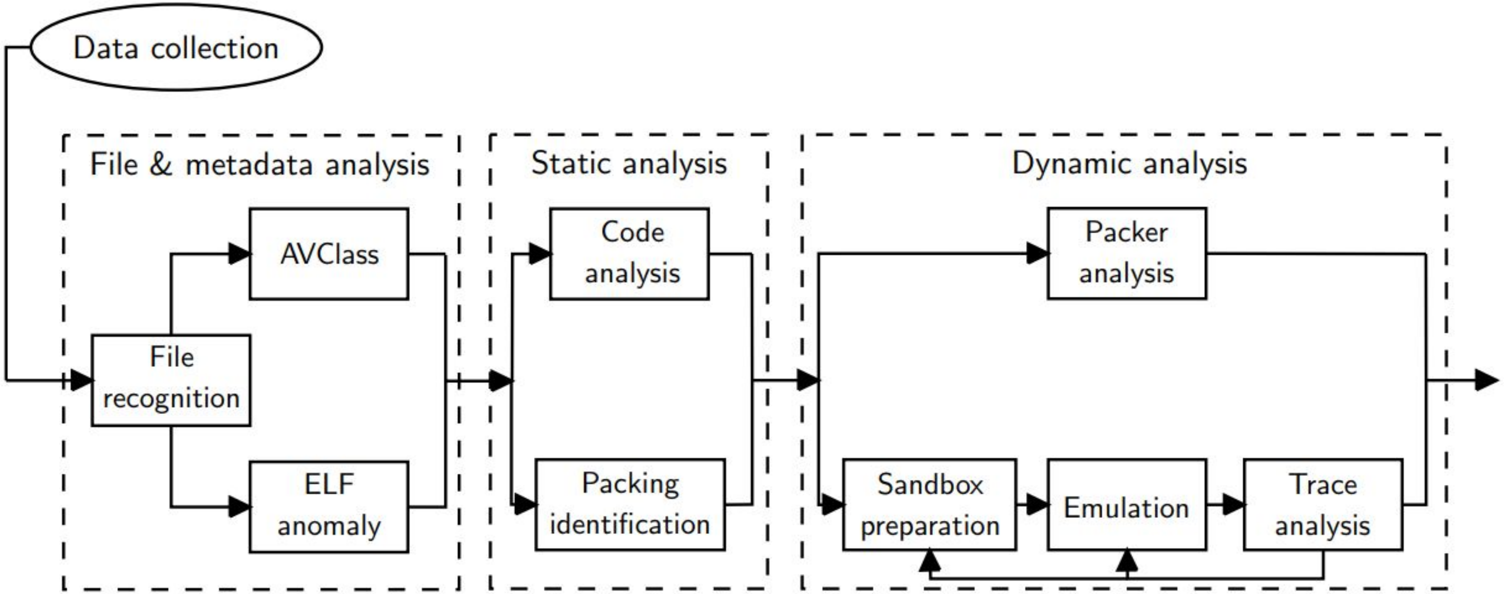
RE few samples is affordable - *thousands* would be a nightmare

- Padawan core handle data and dispatch analysis jobs
- Analysis jobs executed on worker machines
- Jobs instantiated from analysis modules



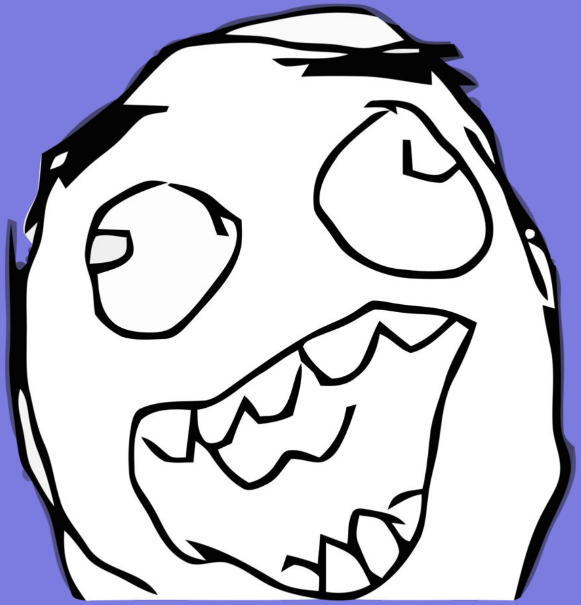
Analysis pipeline

for automated large-scale analysis



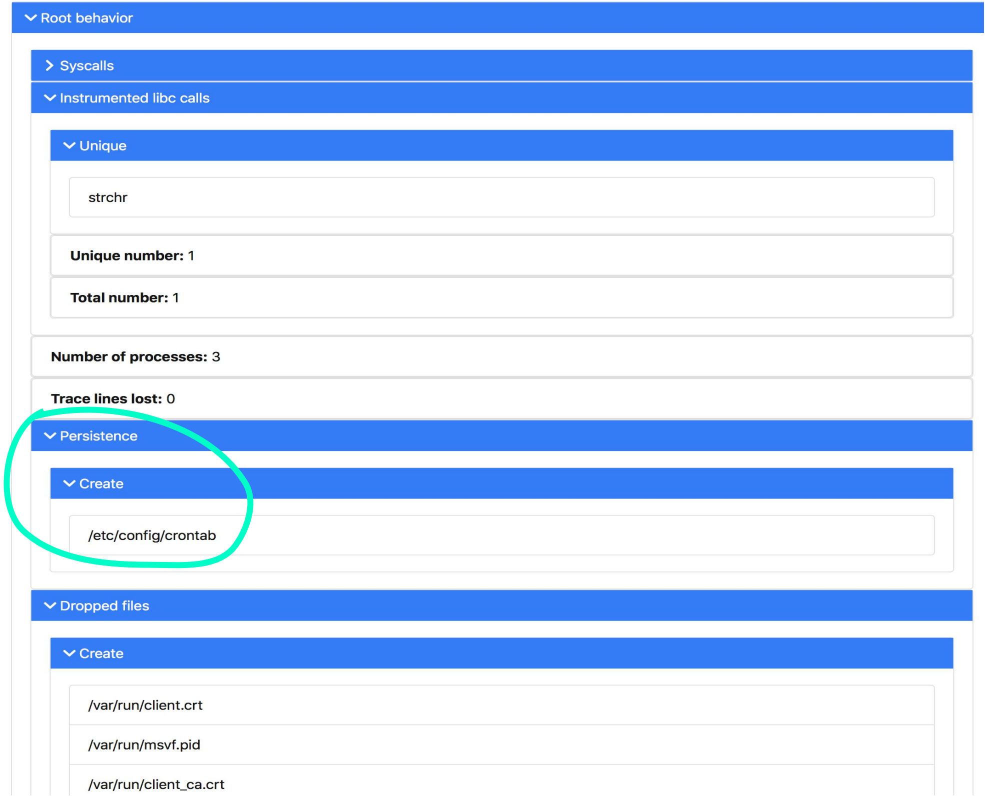
Padawan as a service

-WIP-



VPNFilter first stage

Persistence



Root behavior

Syscalls

Instrumented libc calls

Unique

strchr

Unique number: 1

Total number: 1

Number of processes: 3

Trace lines lost: 0

Persistence

Create

/etc/config/crontab

Dropped files

Create

/var/run/client.crt

/var/run/msvf.pid

/var/run/client_ca.crt

SHA256: 0e0094d9bd396a6594da8e21911a3982cd737b445f591581560d766755097d92

https://blog.talosintelligence.com/2018/05/VPNFilter.html

VPNFilter first stage

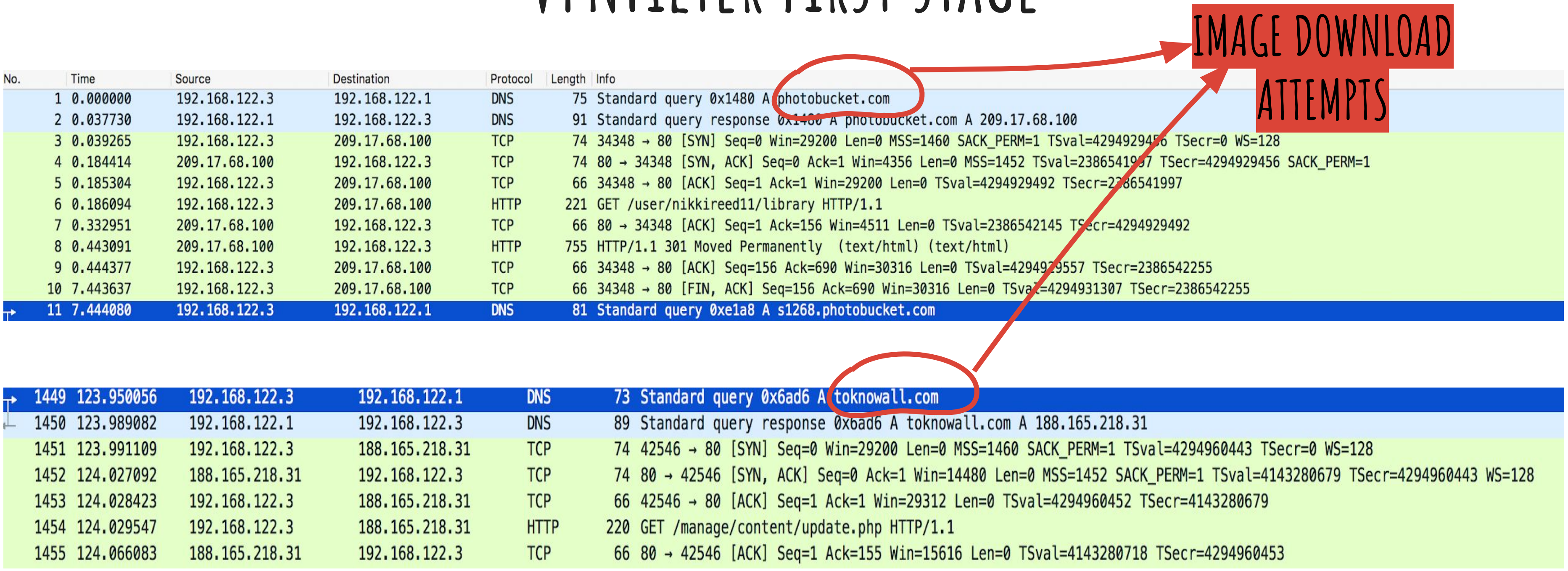


IMAGE DOWNLOAD ATTEMPTS

```
No.   Time        Source           Destination      Protocol  Length  Info
    1 0.000000    192.168.122.3    192.168.122.1    DNS          75  Standard query 0x1480 A photobucket.com
    2 0.037730    192.168.122.1    192.168.122.3    DNS          91  Standard query response 0x1480 A photobucket.com A 209.17.68.100
    3 0.039265    192.168.122.3    209.17.68.100    TCP          74  34348 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=4294929456 TSecr=0 WS=128
    4 0.184414    209.17.68.100    192.168.122.3    TCP          74  80 → 34348 [SYN, ACK] Seq=0 Ack=1 Win=4356 Len=0 MSS=1452 TSval=2386541997 TSecr=4294929456 SACK_PERM=1
    5 0.185304    192.168.122.3    209.17.68.100    TCP          66  34348 → 80 [ACK] Seq=1 Ack=1 Win=29200 Len=0 TSval=4294929492 TSecr=2386541997
    6 0.186094    192.168.122.3    209.17.68.100    HTTP        221  GET /user/nikkireed11/library HTTP/1.1
    7 0.332951    209.17.68.100    192.168.122.3    TCP          66  80 → 34348 [ACK] Seq=1 Ack=156 Win=4511 Len=0 TSval=2386542145 TSecr=4294929492
    8 0.443091    209.17.68.100    192.168.122.3    HTTP        755  HTTP/1.1 301 Moved Permanently  (text/html) (text/html)
    9 0.444377    192.168.122.3    209.17.68.100    TCP          66  34348 → 80 [ACK] Seq=156 Ack=690 Win=30316 Len=0 TSval=4294929557 TSecr=2386542255
   10 7.443637    192.168.122.3    209.17.68.100    TCP          66  34348 → 80 [FIN, ACK] Seq=156 Ack=690 Win=30316 Len=0 TSval=4294931307 TSecr=2386542255
   11 7.444080    192.168.122.3    192.168.122.1    DNS          81  Standard query 0xe1a8 A s1268.photobucket.com

 1449 123.950056  192.168.122.3    192.168.122.1    DNS          73  Standard query 0x6ad6 A toknowall.com
 1450 123.989082  192.168.122.1    192.168.122.3    DNS          89  Standard query response 0x6ad6 A toknowall.com A 188.165.218.31
 1451 123.991109  192.168.122.3    188.165.218.31   TCP          74  42546 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=4294960443 TSecr=0 WS=128
 1452 124.027092  188.165.218.31   192.168.122.3    TCP          74  80 → 42546 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 MSS=1452 SACK_PERM=1 TSval=4143280679 TSecr=4294960443 WS=128
 1453 124.028423  192.168.122.3    188.165.218.31   TCP          66  42546 → 80 [ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=4294960452 TSecr=4143280679
 1454 124.029547  192.168.122.3    188.165.218.31   HTTP        220  GET /manage/content/update.php HTTP/1.1
 1455 124.066083  188.165.218.31   192.168.122.3    TCP          66  80 → 42546 [ACK] Seq=1 Ack=155 Win=15616 Len=0 TSval=4143280718 TSecr=4294960453
```

SHA256: 0e0094d9bd396a6594da8e21911a3982cd737b445f591581560d766755097d92

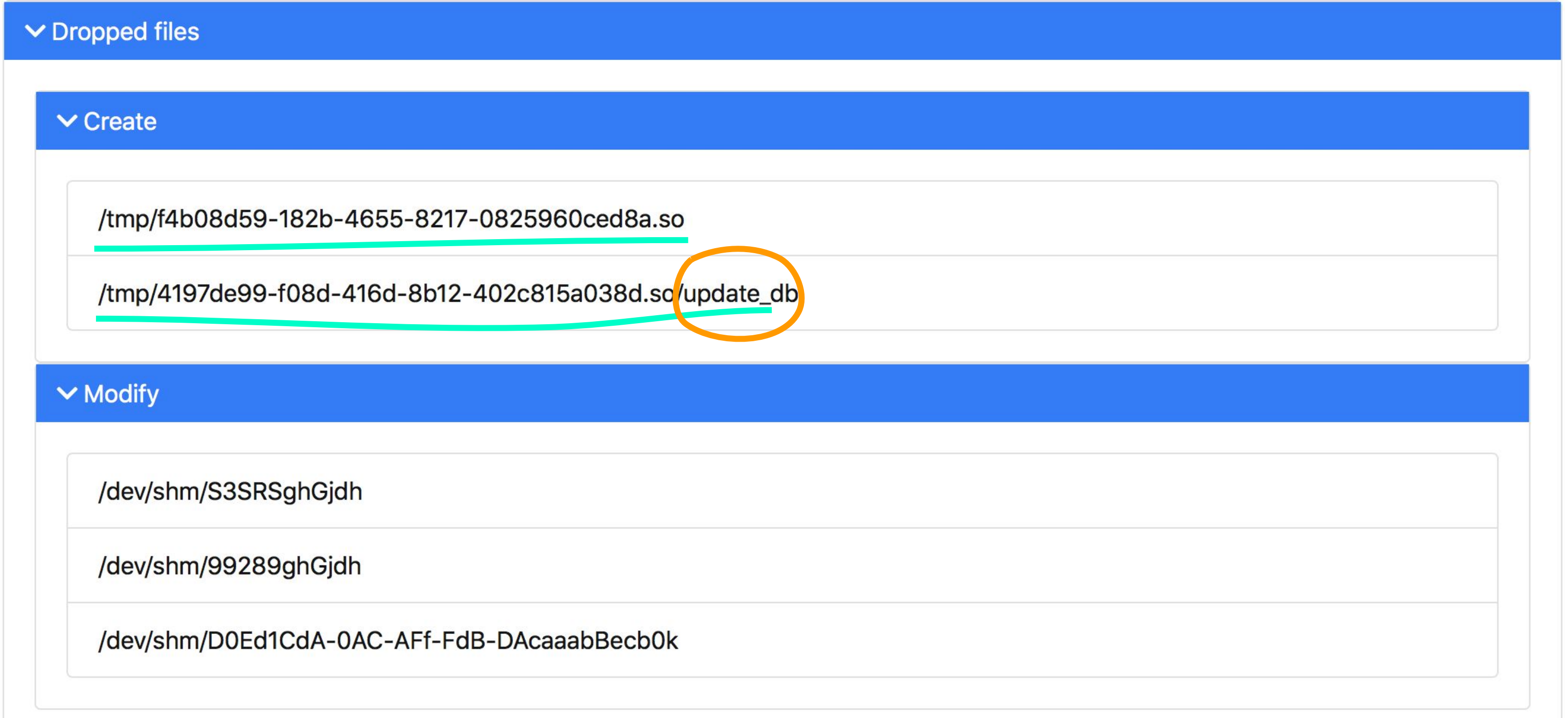
https://blog.talosintelligence.com/2018/05/VPNFilter.html

VPNFilter second stage

```
mkdir("/var/run/d6097e942dd0fdc1fb28ec1814780e6ecc169ec6d24f9954e71954eedbc4c70em", 0770) = 0
mkdir("/var/run/d6097e942dd0fdc1fb28ec1814780e6ecc169ec6d24f9954e71954eedbc4c70ew", 0770) = 0

open("/proc/mtd", O_RDONLY) = -2 (ENOENT)

connect(3, {AF_INET, 127.0.0.1, 9050}, 16) = -111 (ECONNREFUSED)
```

SHA256: d6097e942dd0fdc1fb28ec1814780e6ecc169ec6d24f9954e71954eedbc4c70
https://blog.talosintelligence.com/2018/05/VPNFilter.html

HAND OF THIEF



Dropped files

Create

/tmp/f4b08d59-182b-4655-8217-0825960ced8a.so

/tmp/4197de99-f08d-416d-8b12-402c815a038d.so/update_db

Modify

/dev/shm/S3SRSghGjdh

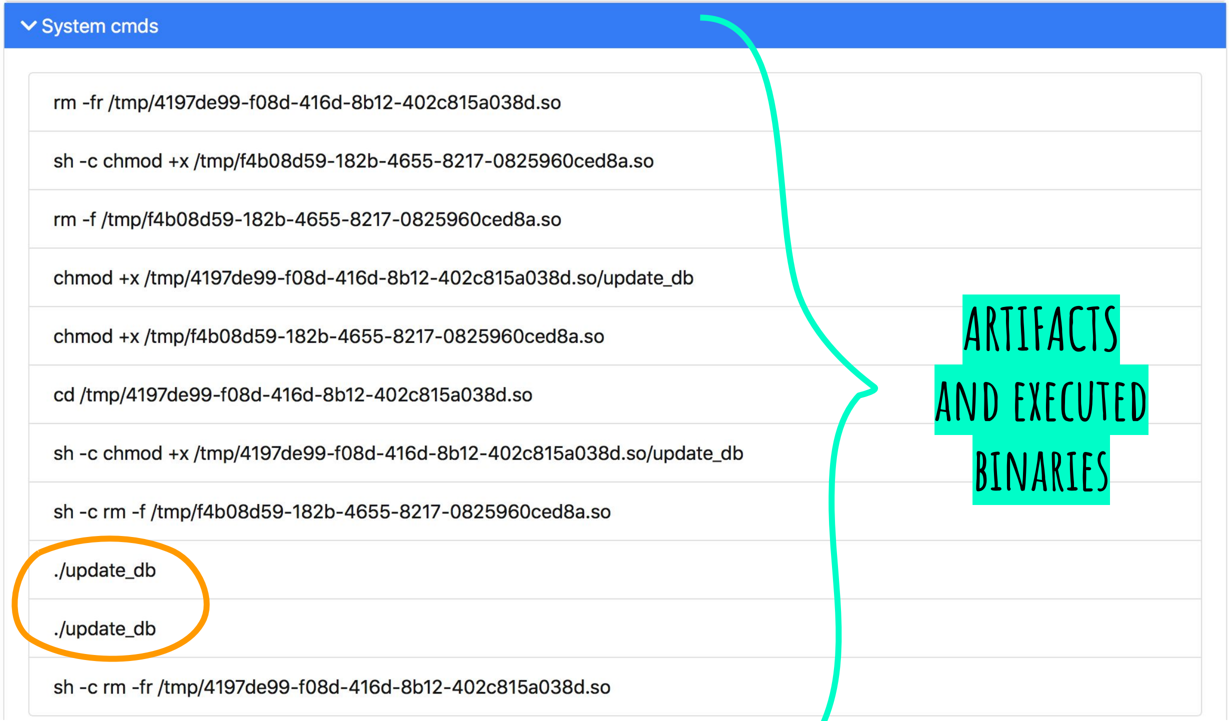
/dev/shm/99289ghGjdh

/dev/shm/D0Ed1CdA-0AC-AFf-FdB-DAcaaabBecb0k

SHA256: bd92ce74844b1ddfdd1b61eac86abe7140d38eedf9c1b06fb7fbf446f6830391

https://blog.avast.com/2013/08/27/linux-trojan-hand-of-thief-ungloved/

HAND OF THIEF



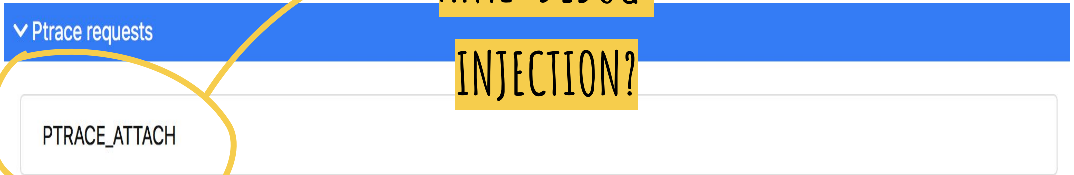
```
rm -fr /tmp/4197de99-f08d-416d-8b12-402c815a038d.so
sh -c chmod +x /tmp/f4b08d59-182b-4655-8217-0825960ced8a.so
rm -f /tmp/f4b08d59-182b-4655-8217-0825960ced8a.so
chmod +x /tmp/4197de99-f08d-416d-8b12-402c815a038d.so/update_db
chmod +x /tmp/f4b08d59-182b-4655-8217-0825960ced8a.so
cd /tmp/4197de99-f08d-416d-8b12-402c815a038d.so
sh -c chmod +x /tmp/4197de99-f08d-416d-8b12-402c815a038d.so/update_db
sh -c rm -f /tmp/f4b08d59-182b-4655-8217-0825960ced8a.so
./update_db
./update_db
sh -c rm -fr /tmp/4197de99-f08d-416d-8b12-402c815a038d.so
```

ARTIFACTS AND EXECUTED BINARIES

SHA256: bd92ce74844b1ddfdd1b61eac86abe7140d38eedf9c1b06fb7fbf446f6830391

https://blog.avast.com/2013/08/27/linux-trojan-hand-of-thief-ungloved/

HAND OF THIEF



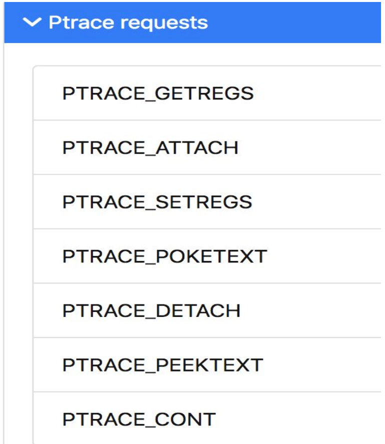
Ptrace requests

PTRACE_ATTACH

ANTI-DEBUG

INJECTION?

USER BEHAVIOR



Ptrace requests

- PTRACE_GETREGS
- PTRACE_ATTACH
- PTRACE_SETREGS
- PTRACE_POKETEXT
- PTRACE_DETACH
- PTRACE_PEEKTEXT
- PTRACE_CONT

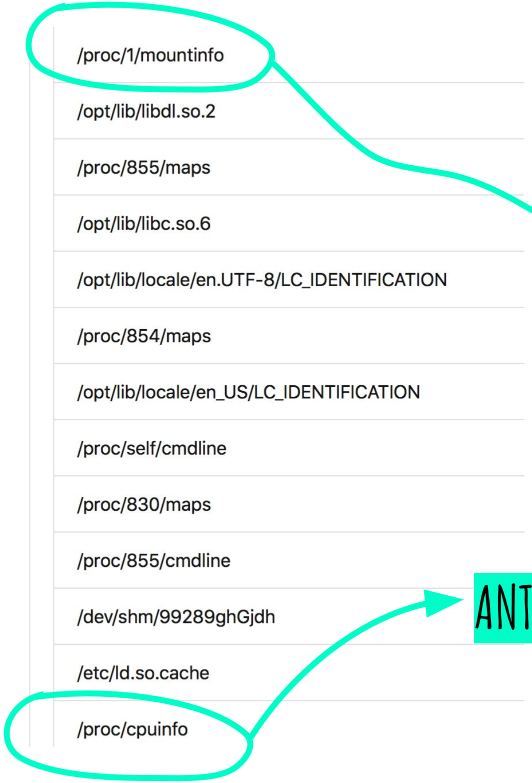
INJECTION

ROOT BEHAVIOR

SHA256: bd92ce74844b1ddfdd1b61eac86abe7140d38eedf9c1b06fb7fbf446f6830391

https://blog.avast.com/2013/08/27/linux-trojan-hand-of-thief-ungloved/

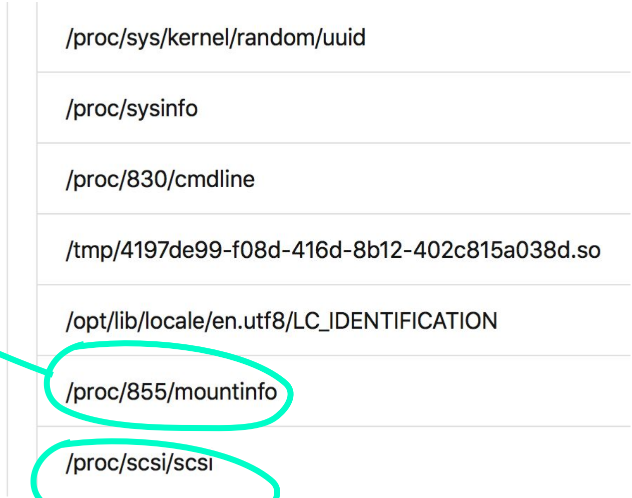
HAND OF THIEF



```
/proc/1/mountinfo
/opt/lib/libdl.so.2
/proc/855/maps
/opt/lib/libc.so.6
/opt/lib/locale/en.UTF-8/LC_IDENTIFICATION
/proc/854/maps
/opt/lib/locale/en_US/LC_IDENTIFICATION
/proc/self/cmdline
/proc/830/maps
/proc/855/cmdline
/dev/shm/99289ghGjdh
/etc/ld.so.cache
/proc/cpuinfo
```

ANTI CHROOT JAIL

ANTI-VM



```
/proc/sys/kernel/random/uuid
/proc/sysinfo
/proc/830/cmdline
/tmp/4197de99-f08d-416d-8b12-402c815a038d.so
/opt/lib/locale/en.utf8/LC_IDENTIFICATION
/proc/855/mountinfo
/proc/scsi/scsi
```

ANTI-VMWARE AND ANTI-VBOX

SHA256: bd92ce74844b1ddfdd1b61eac86abe7140d38eedf9c1b06fb7fbf446f6830391

https://blog.avast.com/2013/08/27/linux-trojan-hand-of-thief-ungloved/

Conclusions

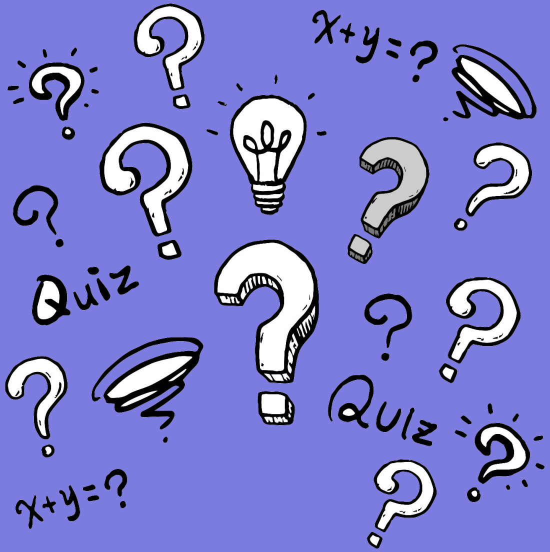
- Shed light on modern Linux malware
- Design a pipeline to cope with ELF binaries
- Release the dataset:

  https://padawan.s3.eurecom.fr/static/data/dataset_sha256.txt

- Open the pipeline to the public:

  https://padawan.s3.eurecom.fr

THANKS!
QUESTIONS?



Emanuele Cozzi @invano
Mariano Graziano @emd3l