Defeating static signatures in blackbox antivirus engines

Insomni'hack 2022

Vladimir Meier

SCRT Information Security

# whoami

- # Security researcher @ SCRT
- # Working on antivirus software since 2015
- # Author of https://github.com/scrt/avcleaner
- # https://blog.scrt.ch «Antivirus Bypass» category

Contents

Demo

- https://github.com/scrt/avdebugger
- ~ 3000 python LoC
- Powered by radare2/rizin, lief and keystone
- Application:
  - Meterpreter's main DLL is detected by Windows Defender
  - Antivirus' verdict is SLFPER:Win32/Meterpreter!ApiRetrieval
  - You have 4 hours

~~1~~3 reasons why antivirus bypass research

- # Security software from a pentester's perspective
- # False sense of security
- # Really legit use case, I swear!

Security software vs pentesters

Pwnable servers / workstations

AD

Meterpreter

Trigger the AV == game over

~~1~~3 reasons why antivirus bypass research

- # Security software from a pentester's perspective
- # False sense of security
- # Really legit use case, I swear!

Why: overpromoted IT security guy

Your next task is to get DA on my infra with my 150 servers. I bought shiny new EDR btw. Oh and there were 3 pentests before. About the SOC team, they might unpatch your Ethernet plug if you're detected. Won't be there this afternoon to fix it though. I think half a day is enough for the test.

MY SECURE INFRA IS SECURE.
WHAT MS17-010? COME ON, THE EDR BLOCKED EVERYTHING, THERE IS NO VULN.
imgflip.com

~~1~~3 reasons why

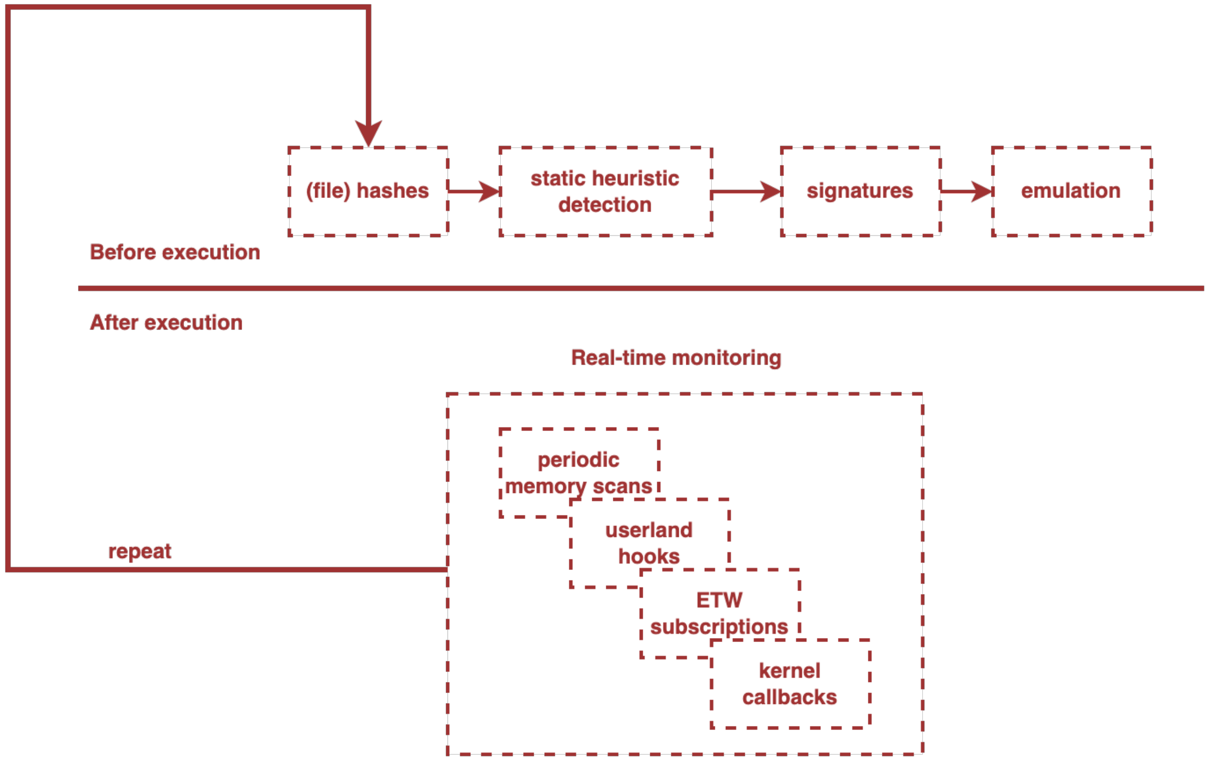
- # Security software from a pentester's perspective
- # False sense of security
- # Really legit use case
  - # Company X sells a software
  - # 39 different antivirus flag it as malware, every new release.
  - # Company X actually worried its own product contains a virus.

~~1~~3 reasons why

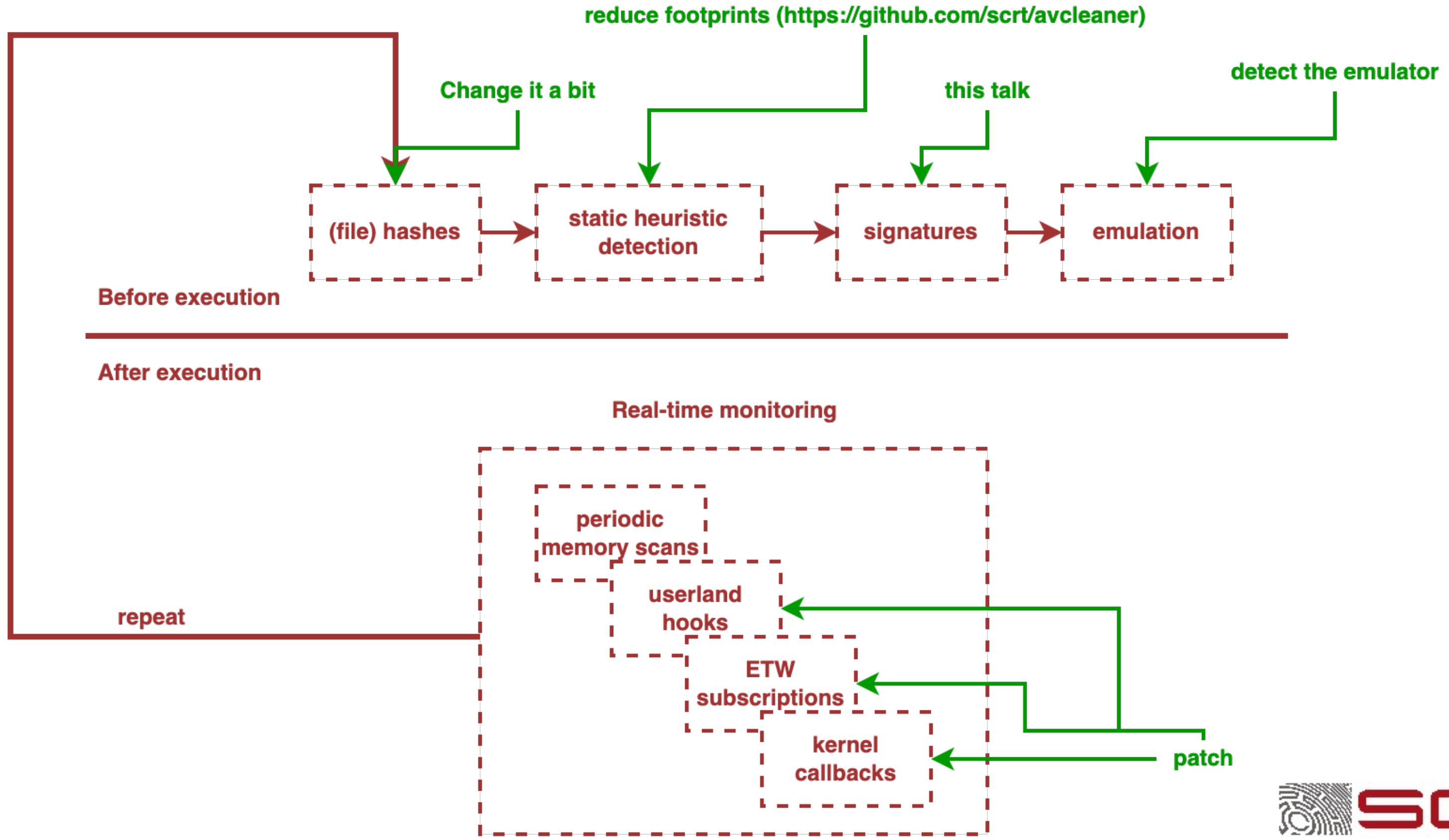
- # Really legit use case
  - # Company X sells a software
  - # 39 different antivirus flag it as malware, every new release.
  - # Company X actually worried its own product contains a virus.
- # **Obviously a false positive, but:**
  - # How do you prove it?
  - # How do you fix it?

=> Call ~~ghosbusters~~ SCRT

Antivirus detection pipeline



Antivirus detection pipeline: bypass



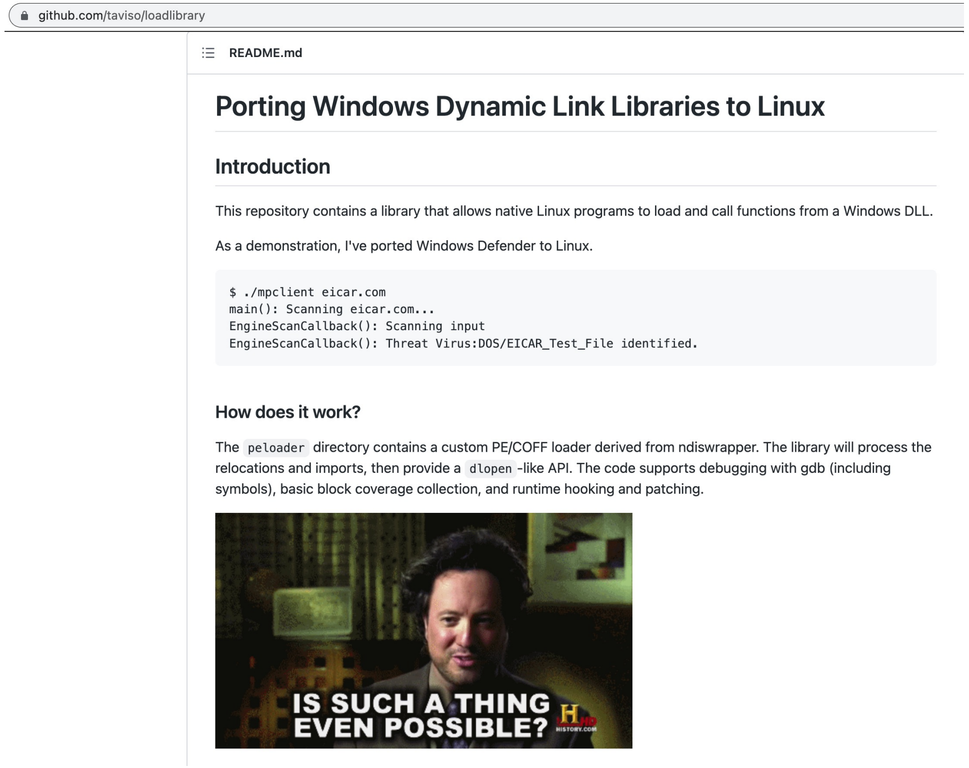
Extracting signatures: Main steps

- Scan automation
- Mutations / search algorithms
  - Prior works
  - Improvements
- Binary patching
- Filtering results
- Encrypting strings directly in the binary

Antivirus scan automation

- Why
  - CI/CD pipeline
  - …or applying mutations until the sample comes out clean => need to scan every sample
- How
  - VirusTotal?
  - Build your own

Taviso's loadlibrary



github.com/taviso/loadlibrary

README.md

Porting Windows Dynamic Link Libraries to Linux

Introduction

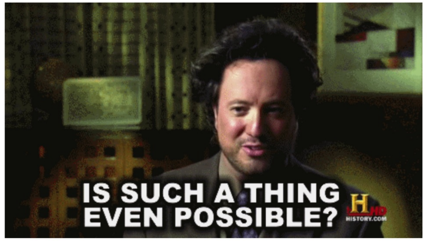
This repository contains a library that allows native Linux programs to load and call functions from a Windows DLL.

As a demonstration, I've ported Windows Defender to Linux.

```
$ ./mpclient eicar.com
main(): Scanning eicar.com...
EngineScanCallback(): Scanning input
EngineScanCallback(): Threat Virus:DOS/EICAR_Test_File identified.
```

How does it work?

The `peloader` directory contains a custom PE/COFF loader derived from ndiswrapper. The library will process the relocations and imports, then provide a `dlopen` -like API. The code supports debugging with gdb (including symbols), basic block coverage collection, and runtime hooking and patching.



Taviso's loadlibrary

# Windows Defender's scan engine is *mpclient.dll*
# "loadlibrary" is able to run it
# Perfect for automation

Problem: other antivirus engines

- A bit more complicated…
- Antivirus with / without command line interface
- Some only run on Windows

Scan automation: VMWare's vmrun

$ vmrun –h

```
vmrun version 1.17.0 build-17964953

Usage: vmrun [AUTHENTICATION-FLAGS] COMMAND [PARAMETERS]

AUTHENTICATION-FLAGS

--------------------

These must appear before the command and any command parameters.


   -T <hostType> (wslfusion)

   -vp <password for encrypted virtual machine>

   -gu <userName in guest OS>

   -gp <password in guest OS>
```

```
Example commands:
CMD              PARAMETERS         DESCRIPTION
--------------   ----------         -----------
start            Path to vmx file   Start a VM
```

Scan automation: VMWare's vmrun

```
$ vmrun –h
```

Command	Use case
CopyFileFromHostToGuest	Upload the sample to the VM
runProgramInGuest	Invoke a scan and get the result

Scan automation: VMWare's vmrun

$ vmrun –h

Command	Use case
CopyFileFromHostToGuest	Upload the sample to the VM
runProgramInGuest	Invoke a scan and get the result

Complete example

arguments

```
vmrun  -T  ws  -gu  <user>  -gp  <password> runProgramInGuest kasp.vmx 'C:\\Program Files (x86)\\Kaspersky Lab\\Kaspersky Anti-Virus 21.3\\avp.exe' SCAN a.exe
```

Path to .vmx

Command line agent

vmrun command

Scan automation: VMWare's vmrun

$ vmrun –h

Command	Use case
CopyFileFromHostToGuest	Upload the sample to the VM
fileExistsInGuest	Some AV scan files when they're written to disk.
runProgramInGuest	Execute the sample. Some AV only scan files upon execution.
fileExistsInGuest	Re-check if the file is deleted. If yes, sample is a malware.

Windows-only AV with no command line agent (Avast, DeepInstinct…)

Extracting signatures: Main steps

- Scan automation
- Mutations / search algorithms
  - Prior works
  - Improvements
- Binary patching
- Filtering results
- Encrypting strings directly in the binary

The needle and the haystack

```
$ ls -lh ext_server_kiwi.x64.dll
-rwxr-xr-x  1 vladimir  staff   994K Mar 21 15:45 ext_server_kiwi.x64.dll
```

Which parts are seen as malicious by the AV?

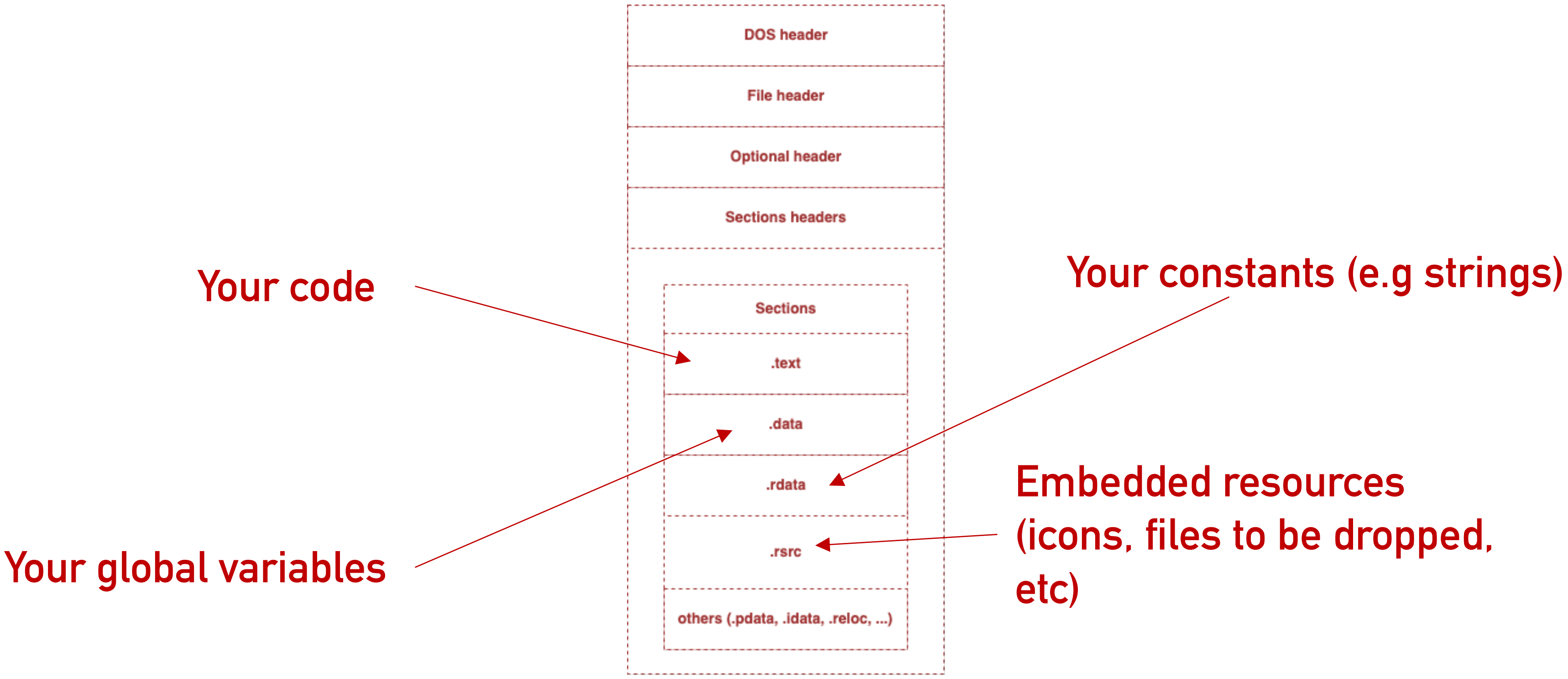
- Public projects
  - Dsplit (2006)
  - DefenderCheck.exe (April, 2019)

Idea: split a binary into smaller parts to see which one triggers the AV

Problems and solutions

- Problem
  - Splitting an executable into chunks
    - Corrupted Portable Executable structure
    - Granularity (more on that later)
- Solution
  - PE format-aware targeted mutations

PE format 101


DOS header
File header
Optional header
Sections headers
Sections
.text
.data
.rdata
.rsrc
others (.pdata, .idata, .reloc, ...)
Your code
Your global variables
Your constants (e.g strings)
Embedded resources (icons, files to be dropped, etc)

All kinds of mutations

- If you were a lazy engineer implementing an antivirus, what would you do?
  - Search sequences of bytes
  - Search strings

All kinds of mutations

- If you were a lazy engineer implementing an antivirus, what would you do?
  - Search sequences of bytes
  - Search strings

- Where would you do it?

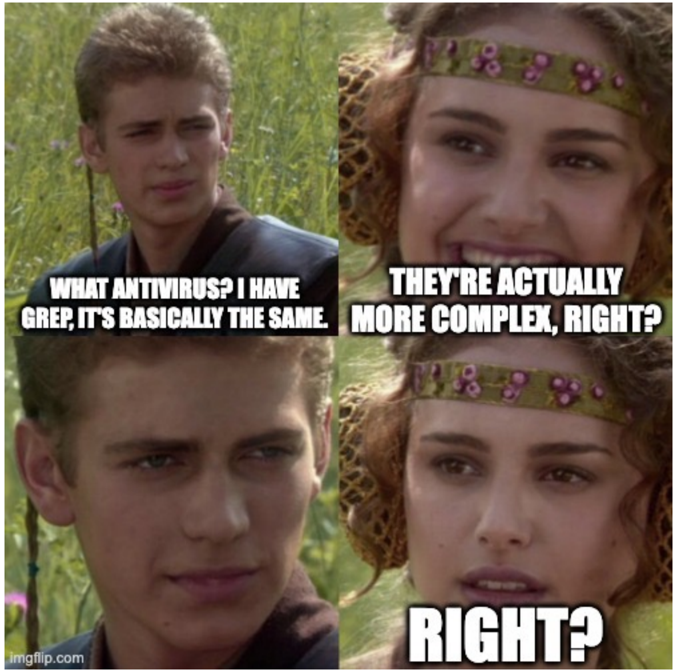
All kinds of mutations

# Where would you do it?

What	Where
Sequence of bytes to find hashes / inlined constants	.text section
Sequence of bytes to find (big) shellcodes	.data section
Strings	.rdata section
Embeddeds files with known hashes	.rsrc section

All kinds of mutations

Is it that simple?


WHAT ANTIVIRUS? I HAVE GREP, IT'S BASICALLY THE SAME.
THEY'RE ACTUALLY MORE COMPLEX, RIGHT?
RIGHT?
imgflip.com

Hypothetize and verify

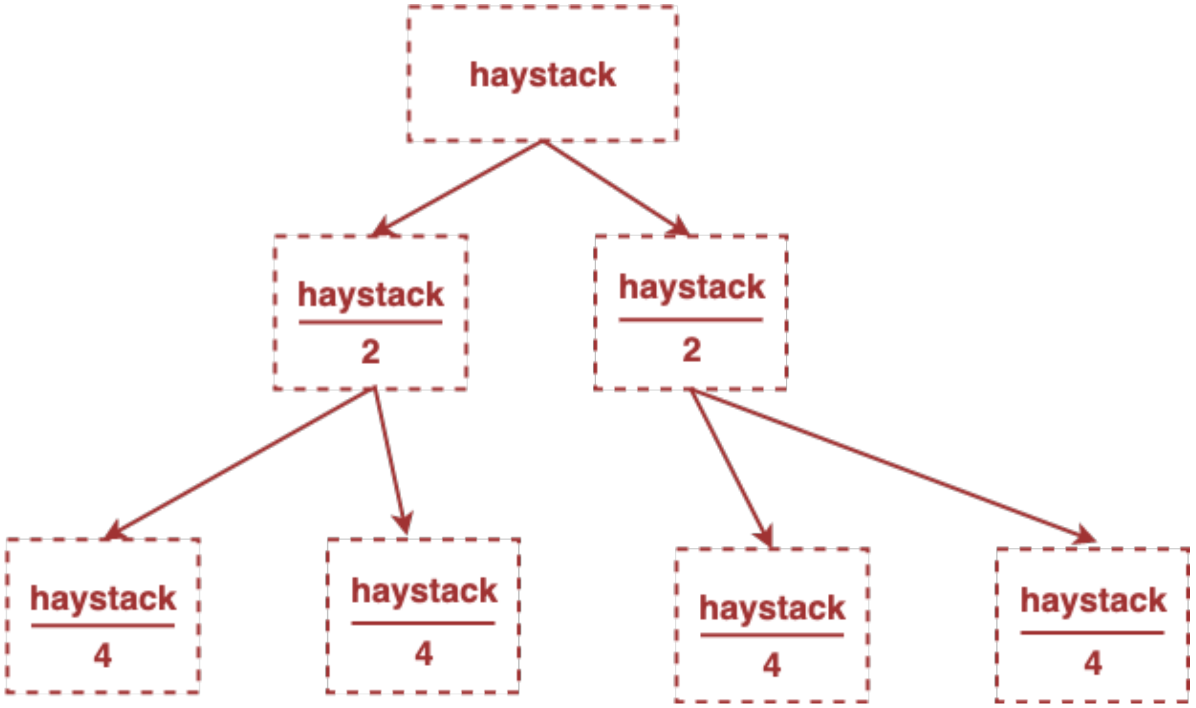
# Demo

```
gem fetch metasploit-payloads
gem unpack metasploit-payloads
ls -lht metasploit-payloads-2.0.66/data/meterpreter/
-rw-r--r--  1 vladimir  wheel   199K Mar 23 20:19 screenshot.x86.dll
-rw-r--r--  1 vladimir  wheel   199K Mar 23 20:19 screenshot.x64.dll
-rw-r--r--  1 vladimir  wheel   170K Mar 23 20:19 metsrv.x86.dll
-rw-r--r--  1 vladimir  wheel   195K Mar 23 20:19 metsrv.x64.dll
-rw-r--r--  1 vladimir  wheel   364K Mar 23 20:19 ext_server_stdapi.x86.dll
-rw-r--r--  1 vladimir  wheel   400K Mar 23 20:19 ext_server_stdapi.x64.dll
-rw-r--r--  1 vladimir  wheel   106K Mar 23 20:19 ext_server_priv.x86.dll
-rw-r--r--  1 vladimir  wheel   127K Mar 23 20:19 ext_server_priv.x64.dll
-rw-r--r--  1 vladimir  wheel   1.1M Mar 23 20:19 ext_server_kiwi.x86.dll
-rw-r--r--  1 vladimir  wheel   1.4M Mar 23 20:19 ext_server_kiwi.x64.dll
...
```

Fantastic signatures and where to find them

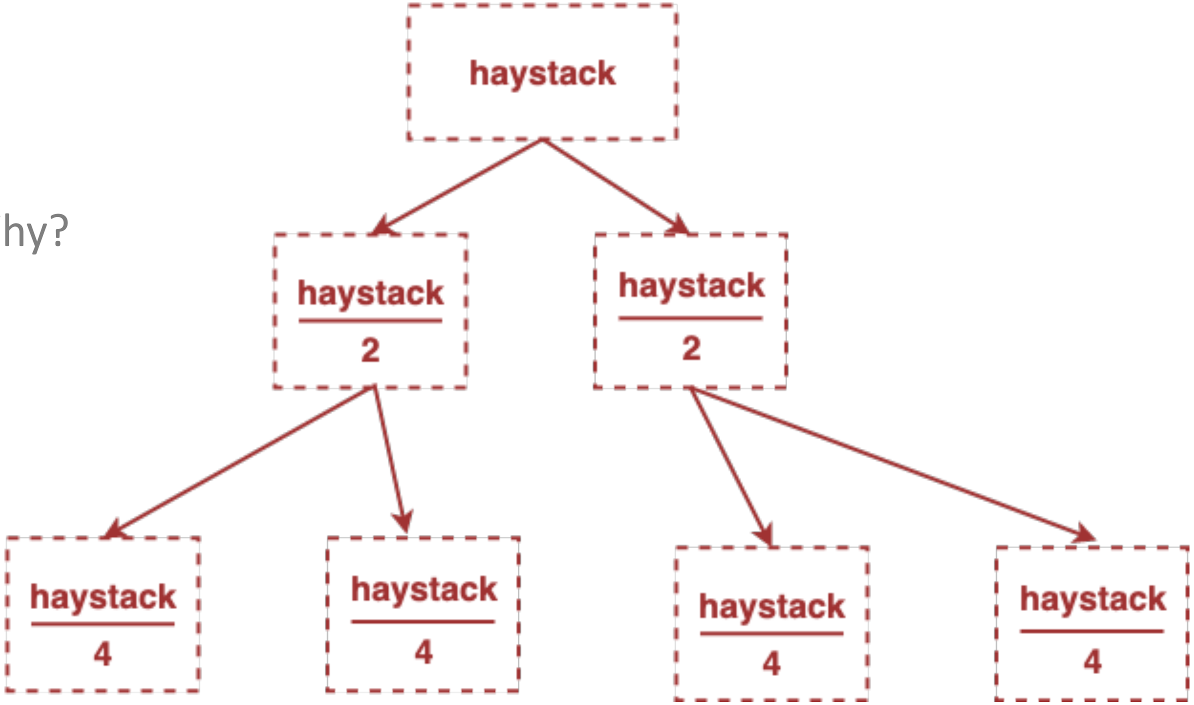
- Signatures can target sequences of bytes of arbitrary lengths
- Sequences of bytes can have “malicious” or “benign” scores
- How to find those with the highest score?

Divide and conquer search algorithm 101



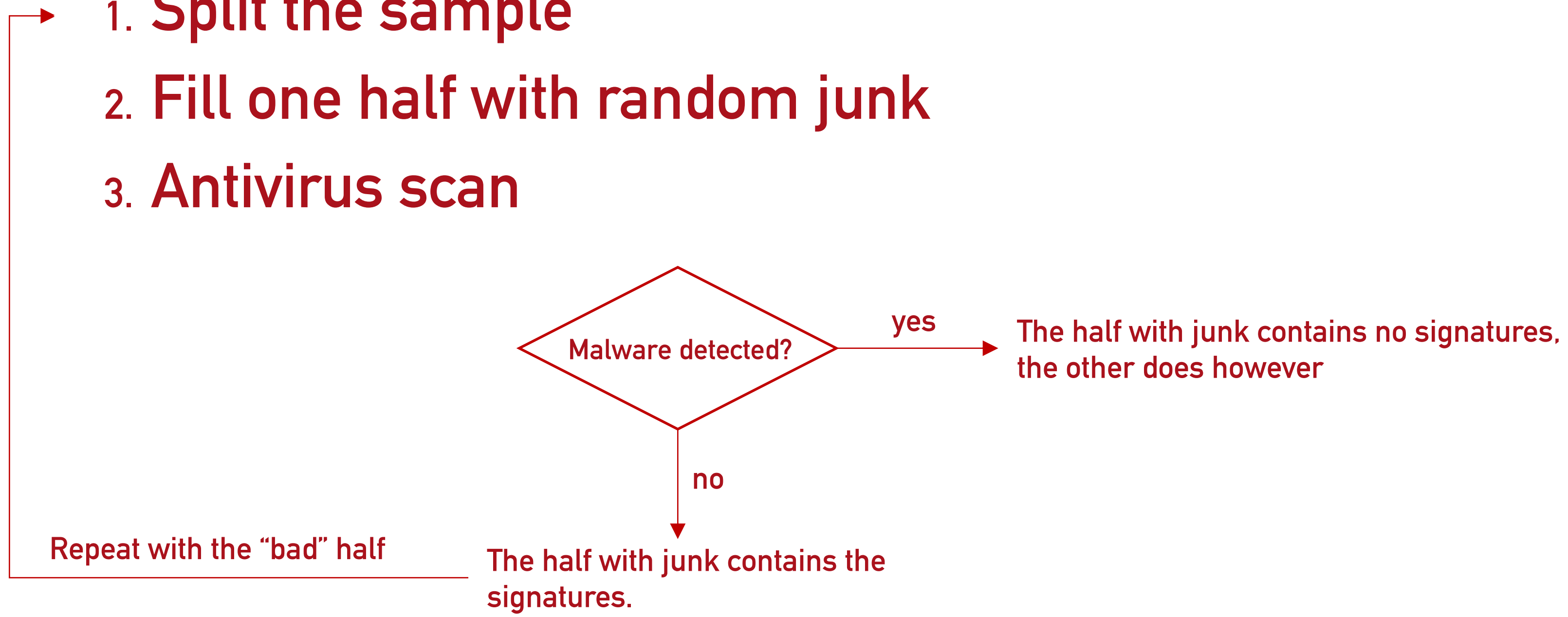
Divide and conquer search algorithm 101

Why?



Divide and conquer

1. Split the sample
2. Fill one half with random junk
3. Antivirus scan


Malware detected?
yes
The half with junk contains no signatures, the other does however
no
The half with junk contains the signatures.
Repeat with the “bad” half

Granularity

- We can do better than splitting and scanning things
- “Know your data”

What	Where	Granularity	Identification
Sequence of bytes to find shellcodes	.text section	Functions	Radare2 / rizin disassembler
Sequence of bytes to find (big) shellcodes	.data section	Global variables	Custom algo
Strings	.rdata section	…Strings	Radare2 / rizin
Embeddeds files with known hashes	.rsrc section	Resources	Doesn’t matter

Example with strings

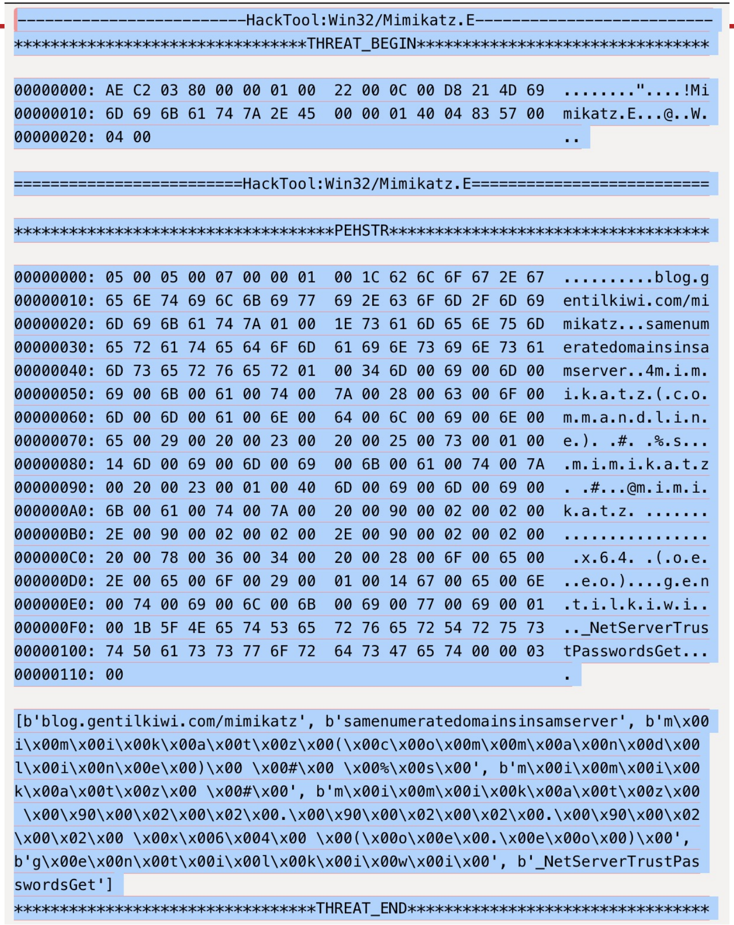
- # Mimikatz contains ~5 thousands strings
- # 5-100 characters per strings -> good granularity
- # Divide and conquer
  - # Divide the 5k strings into 2 clusters
    - # Replace every string in cluster 1 with random data of equal size
    - # Cluster 2 is left intact
    - # Patch the sample with these modifications
    - # Antivirus scan to find which cluster contains “bad” strings.
    - # Repeat, until you have 2 clusters with 1 string each.

Example with strings

- Demo
- Proof

Validation

Reverse-engineering of Defender’s signatures database by @commial and Romain Melchiorre (SCRT)

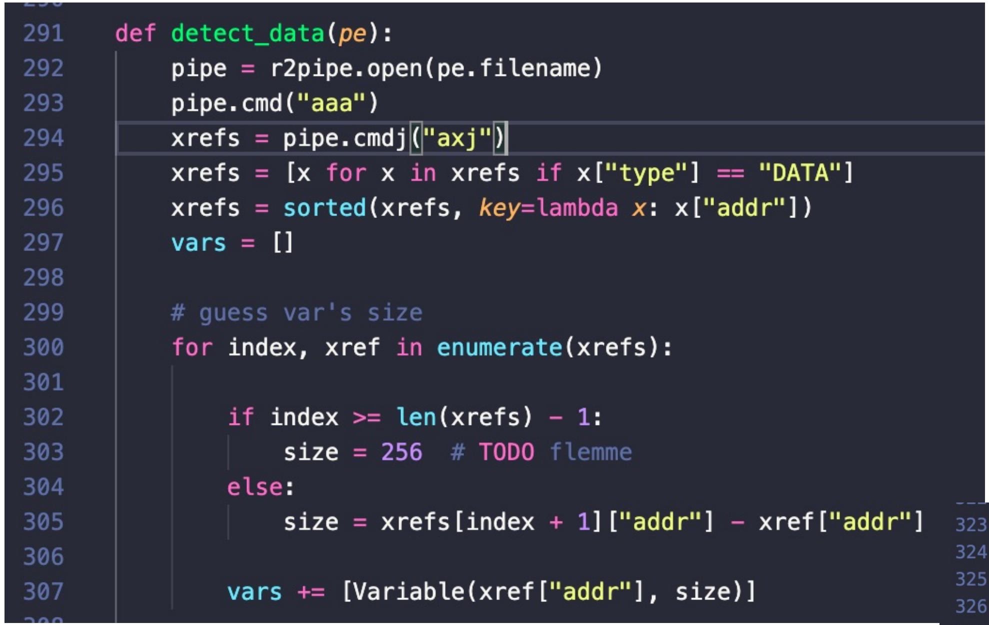


```
-------------------------HackTool:Win32/Mimikatz.E--------------------------
********************************THREAT_BEGIN********************************

00000000: AE C2 03 80 00 00 01 00  22 00 0C 00 D8 21 4D 69  ........"....!Mi
00000010: 6D 69 6B 61 74 7A 2E 45  00 00 01 40 04 83 57 00  mikatz.E...@..W.
00000020: 04 00                                             ..

=========================HackTool:Win32/Mimikatz.E==========================

***********************************PEHSTR***********************************

00000000: 05 00 05 00 07 00 00 01  00 1C 62 6C 6F 67 2E 67  ..........blog.g
00000010: 65 6E 74 69 6C 6B 69 77  69 2E 63 6F 6D 2F 6D 69  entilkiwi.com/mi
00000020: 6D 69 6B 61 74 7A 01 00  1E 73 61 6D 65 6E 75 6D  mikatz...samenum
00000030: 65 72 61 74 65 64 6F 6D  61 69 6E 73 69 6E 73 61  eratedomainsinsa
00000040: 6D 73 65 72 76 65 72 01  00 34 6D 00 69 00 6D 00  mserver..4m.i.m.
00000050: 69 00 6B 00 61 00 74 00  7A 00 28 00 63 00 6F 00  i.k.a.t.z.(.c.o.
00000060: 6D 00 6D 00 61 00 6E 00  64 00 6C 00 69 00 6E 00  m.m.a.n.d.l.i.n.
00000070: 65 00 29 00 20 00 23 00  20 00 25 00 73 00 01 00  e.). .#. .%.s...
00000080: 14 6D 00 69 00 6D 00 69  00 6B 00 61 00 74 00 7A  .m.i.m.i.k.a.t.z
00000090: 00 20 00 23 00 01 00 40  6D 00 69 00 6D 00 69 00  . .#...@m.i.m.i.
000000A0: 6B 00 61 00 74 00 7A 00  20 00 90 00 02 00 02 00  k.a.t.z. .......
000000B0: 2E 00 90 00 02 00 02 00  2E 00 90 00 02 00 02 00  ................
000000C0: 20 00 78 00 36 00 34 00  20 00 28 00 6F 00 65 00   .x.6.4. .(.o.e.
000000D0: 2E 00 65 00 6F 00 29 00  01 00 14 67 00 65 00 6E  ..e.o.)....g.e.n
000000E0: 00 74 00 69 00 6C 00 6B  00 69 00 77 00 69 00 01  .t.i.l.k.i.w.i..
000000F0: 00 1B 5F 4E 65 74 53 65  72 76 65 72 54 72 75 73  .._NetServerTrus
00000100: 74 50 61 73 73 77 6F 72  64 73 47 65 74 00 00 03  tPasswordsGet...
00000110: 00                                                .

[b'blog.gentilkiwi.com/mimikatz', b'samenumeratedomainsinsamserver', b'm\x00
i\x00m\x00i\x00k\x00a\x00t\x00z\x00(\x00c\x00o\x00m\x00m\x00a\x00n\x00d\x00
l\x00i\x00n\x00e\x00)\x00 \x00#\x00 \x00%\x00s\x00', b'm\x00i\x00m\x00i\x00
k\x00a\x00t\x00z\x00 \x00#\x00', b'm\x00i\x00m\x00i\x00k\x00a\x00t\x00z\x00
 \x00\x90\x00\x02\x00\x02\x00.\x00\x90\x00\x02\x00\x02\x00.\x00\x90\x00\x02
\x00\x02\x00 \x00x\x006\x004\x00 \x00(\x00o\x00e\x00.\x00e\x00o\x00)\x00',
b'g\x00e\x00n\x00t\x00i\x00l\x00k\x00i\x00w\x00i\x00', b'_NetServerTrustPas
swordsGet']
*********************************THREAT_END*********************************
```

Granularity: global variables

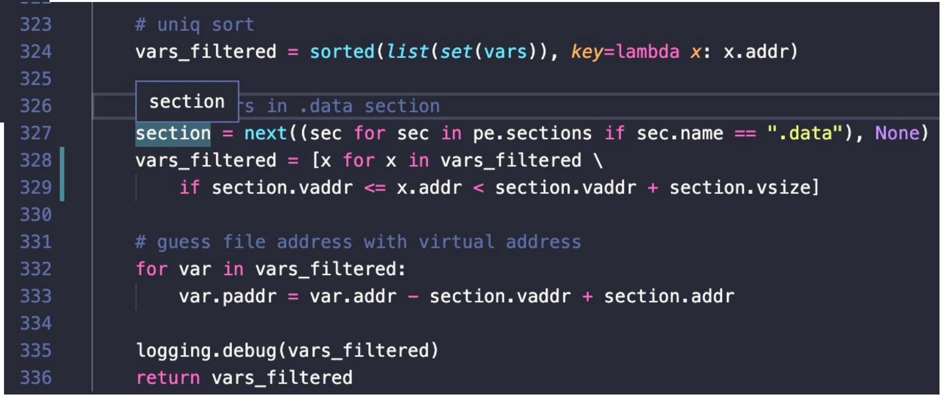
- # What if there is a signature in the .data section?
  - # Hard for the human eye and / or intuition to recognize raw binary
- # Solution: recover global variables
  - # Use radare2 / rizin to find cross-references in .data section
  - # The length of the variable is determined by the next XREF location.
  - # Divide and conquer
  - # Once you know which variable, use its XREF to understand what it is.

Global variables recovery



```python
def detect_data(pe):
    pipe = r2pipe.open(pe.filename)
    pipe.cmd("aaa")
    xrefs = pipe.cmdj("axj")
    xrefs = [x for x in xrefs if x["type"] == "DATA"]
    xrefs = sorted(xrefs, key=lambda x: x["addr"])
    vars = []

    # guess var's size
    for index, xref in enumerate(xrefs):

        if index >= len(xrefs) - 1:
            size = 256  # TODO flemme
        else:
            size = xrefs[index + 1]["addr"] - xref["addr"]

        vars += [Variable(xref["addr"], size)]
```

- pipe.cmdj: run r2 cmd and parse JSON
- axj: enum xrefs as JSON



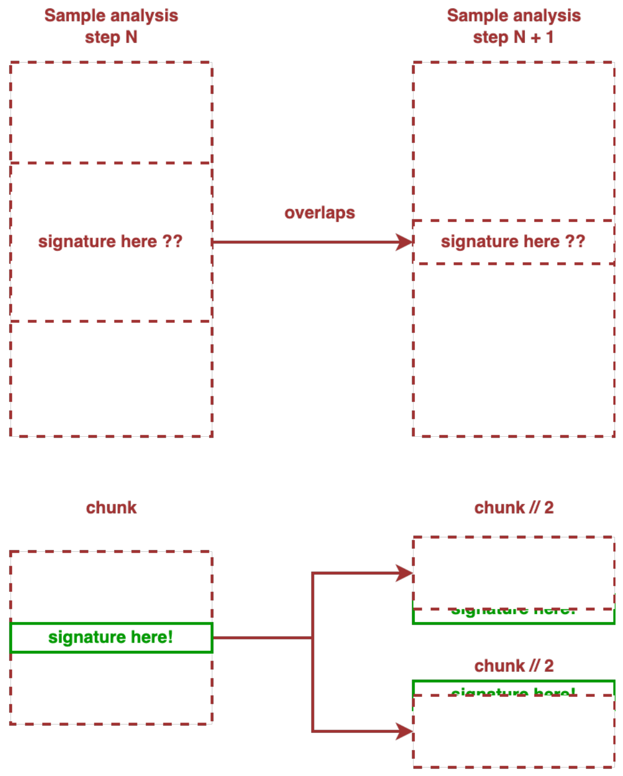
```python
    # uniq sort
    vars_filtered = sorted(list(set(vars)), key=lambda x: x.addr)

    [illegible]s in .data section
    section = next((sec for sec in pe.sections if sec.name == ".data"), None)
    vars_filtered = [x for x in vars_filtered \
        if section.vaddr <= x.addr < section.vaddr + section.vsize]

    # guess file address with virtual address
    for var in vars_filtered:
        var.paddr = var.addr - section.vaddr + section.addr

    logging.debug(vars_filtered)
    return vars_filtered
```

Global variables

- Is that really necessary?
- Enter "SLFPER:Win32/Meterpreter!ApiRetrieval”
- Present in the .data section of metsrv.x64.dll
- Showtime

Filtering results

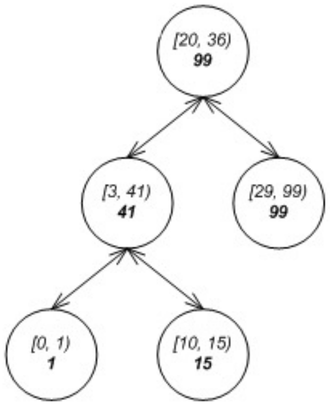
- # When all else fails
  - # -> chunks
- # But chunks may:
  - # overlap
  - # envelop
  - # intersect



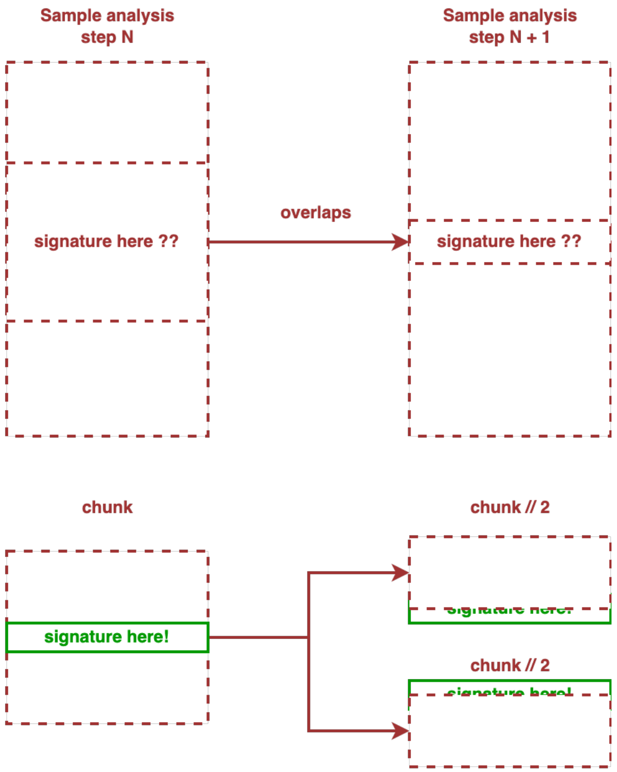
Filtering results

# Solution

# Interval trees



img src: https://en.wikipedia.org/wiki/Interval_tree

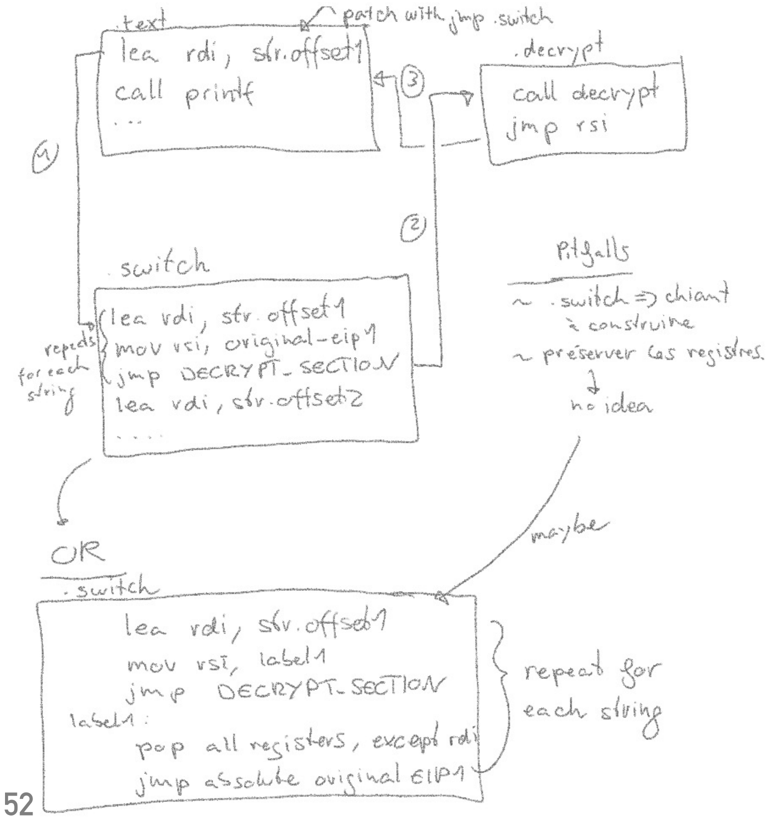


Bonus: Automated binary patching

Automated binary patching

- Can we encrypt strings in a binary without breaking anything?

Fun with LIEF, radare2 and keystone



```
.text
lea rdi, str.offset1
call printf
...
```

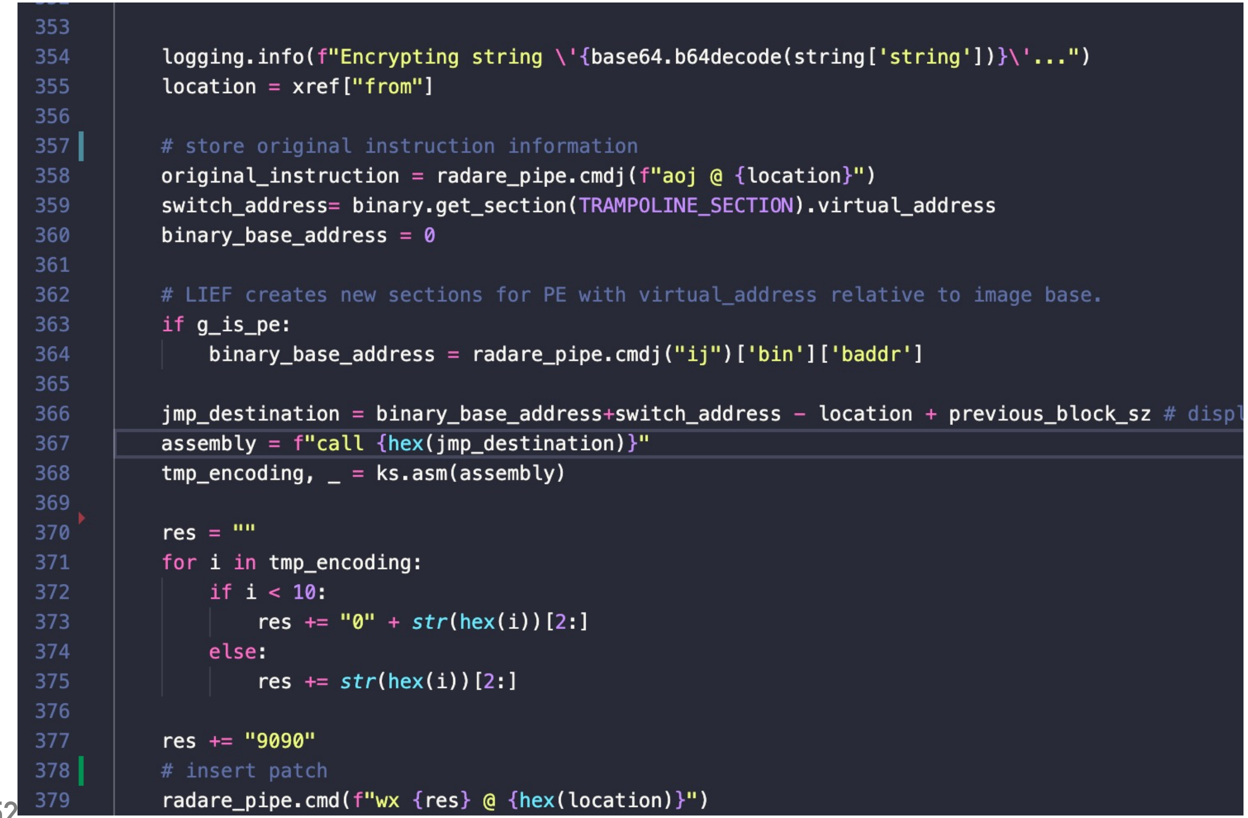
```
.decrypt
call decrypt
jmp rsi
```

```
.switch
lea rdi, str.offset1
mov rsi, original-eip1
jmp DECRYPT_SECTION
lea rdi, str.offset2
....
```

```
.switch
    lea rdi, str.offset1
    mov rsi, label1
    jmp DECRYPT_SECTION
label1:
    pop all registers, except rdi
    jmp absolute original EIP1
```

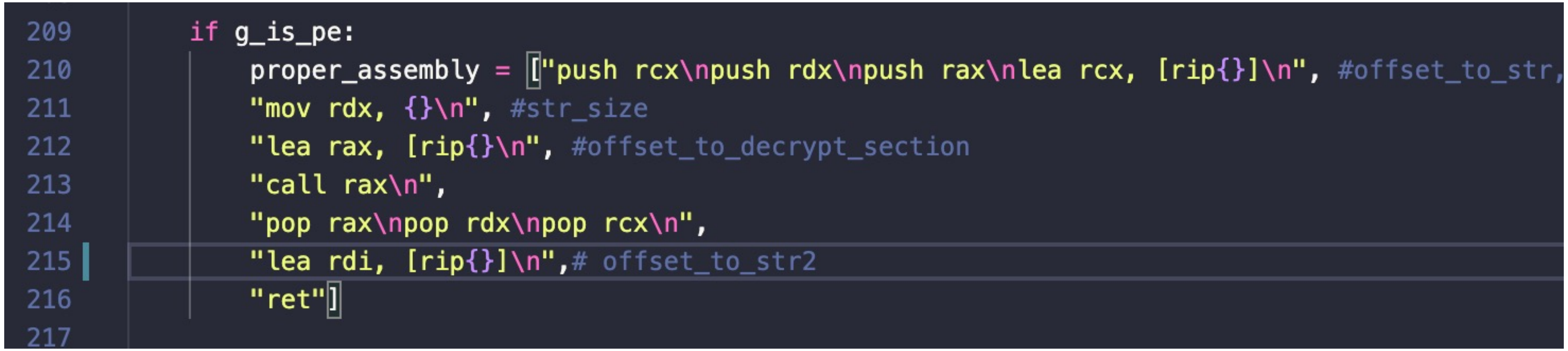
1. Inject a function that takes a string as input and decrypts it
2. Enumerate strings
3. Enumerate xrefs to each string
4. Patch the xref to hijack execution flow
5. Redirect into a switch table that
    1. Saves original instruction pointer into a RSI
    2. Set registers (string address, string size)
    3. Call decryption function
    4. Jump to RSI

Hooking with radare2



```python
    logging.info(f"Encrypting string \'{base64.b64decode(string['string'])}\'...")
    location = xref["from"]

    # store original instruction information
    original_instruction = radare_pipe.cmdj(f"aoj @ {location}")
    switch_address= binary.get_section(TRAMPOLINE_SECTION).virtual_address
    binary_base_address = 0

    # LIEF creates new sections for PE with virtual_address relative to image base.
    if g_is_pe:
        binary_base_address = radare_pipe.cmdj("ij")['bin']['baddr']

    jmp_destination = binary_base_address+switch_address - location + previous_block_sz # displ
    assembly = f"call {hex(jmp_destination)}"
    tmp_encoding, _ = ks.asm(assembly)

    res = ""
    for i in tmp_encoding:
        if i < 10:
            res += "0" + str(hex(i))[2:]
        else:
            res += str(hex(i))[2:]

    res += "9090"
    # insert patch
    radare_pipe.cmd(f"wx {res} @ {hex(location)}")
```

Hook content



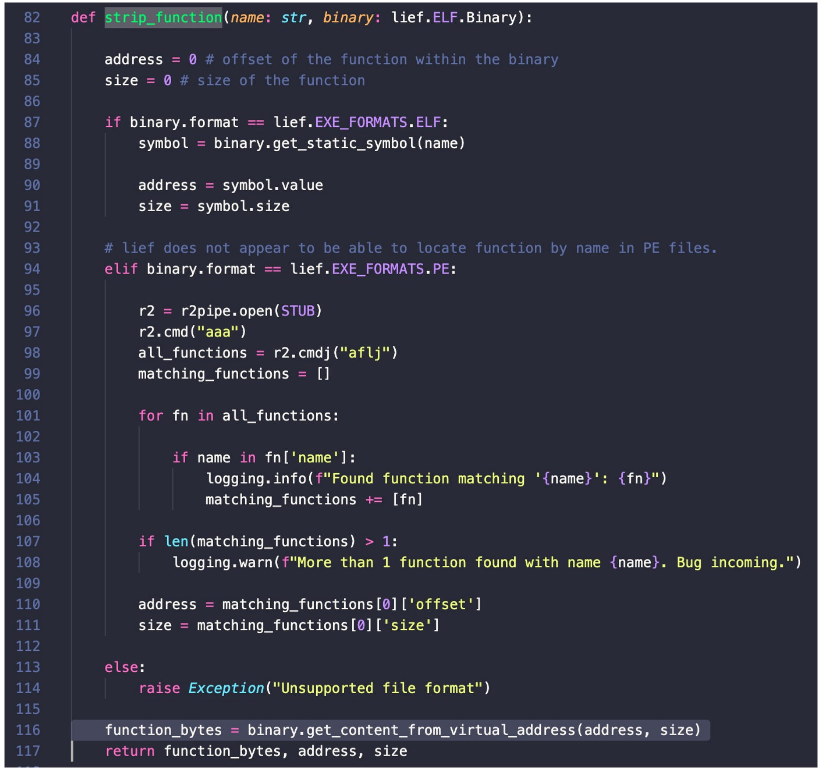
```
    if g_is_pe:
        proper_assembly = ["push rcx\npush rdx\npush rax\nlea rcx, [rip{}]\n", #offset_to_str,
        "mov rdx, {}\n", #str_size
        "lea rax, [rip{}\n", #offset_to_decrypt_section
        "call rax\n",
        "pop rax\npop rdx\npop rcx\n",
        "lea rdi, [rip{}]\n",# offset_to_str2
        "ret"]
```

- Built each time for each string
- Dynamically assembled with Keystone
- Merged at the end of the switch table

Inject a decryption function

1. Don’t want to program in assembly, so I write a C function that encrypts stuff
2. Build binary with –fpie
3. Copy the function’s code with LIEF into the other binary.

Inject a decryption function



```python
def strip_function(name: str, binary: lief.ELF.Binary):

    address = 0 # offset of the function within the binary
    size = 0 # size of the function

    if binary.format == lief.EXE_FORMATS.ELF:
        symbol = binary.get_static_symbol(name)

        address = symbol.value
        size = symbol.size

    # lief does not appear to be able to locate function by name in PE files.
    elif binary.format == lief.EXE_FORMATS.PE:

        r2 = r2pipe.open(STUB)
        r2.cmd("aaa")
        all_functions = r2.cmdj("aflj")
        matching_functions = []

        for fn in all_functions:

            if name in fn['name']:
                logging.info(f"Found function matching '{name}': {fn}")
                matching_functions += [fn]

        if len(matching_functions) > 1:
            logging.warn(f"More than 1 function found with name {name}. Bug incoming.")

        address = matching_functions[0]['offset']
        size = matching_functions[0]['size']

    else:
        raise Exception("Unsupported file format")

    function_bytes = binary.get_content_from_virtual_address(address, size)
    return function_bytes, address, size
```

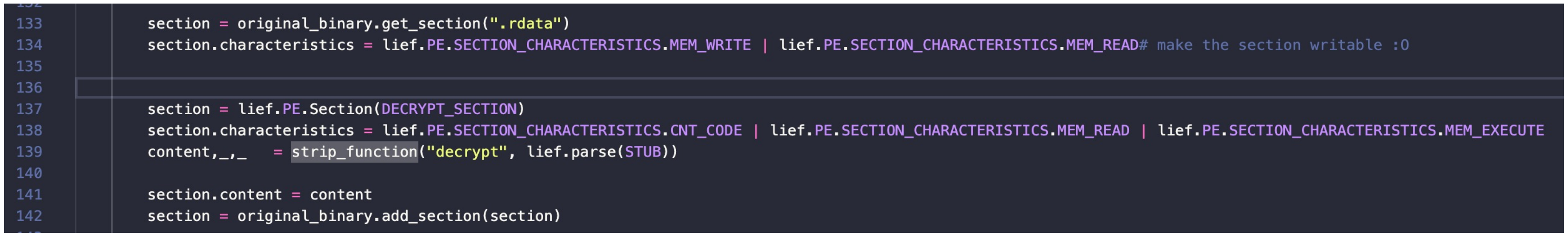
r2.cmdj("aflj"): enum functions as JSON

LIEF: get_content_from_virtual_address

Inject a decryption function

1. Don’t want to program in assembly, so I write a C function that encrypts stuff
2. Build binary with –fpie
3. Copy the function’s code with LIEF into the other binary.

Simple code injection:



```
section = original_binary.get_section(".rdata")
section.characteristics = lief.PE.SECTION_CHARACTERISTICS.MEM_WRITE | lief.PE.SECTION_CHARACTERISTICS.MEM_READ# make the section writable :O


section = lief.PE.Section(DECRYPT_SECTION)
section.characteristics = lief.PE.SECTION_CHARACTERISTICS.CNT_CODE | lief.PE.SECTION_CHARACTERISTICS.MEM_READ | lief.PE.SECTION_CHARACTERISTICS.MEM_EXECUTE
content,_,_   = strip_function("decrypt", lief.parse(STUB))

section.content = content
section = original_binary.add_section(section)
```

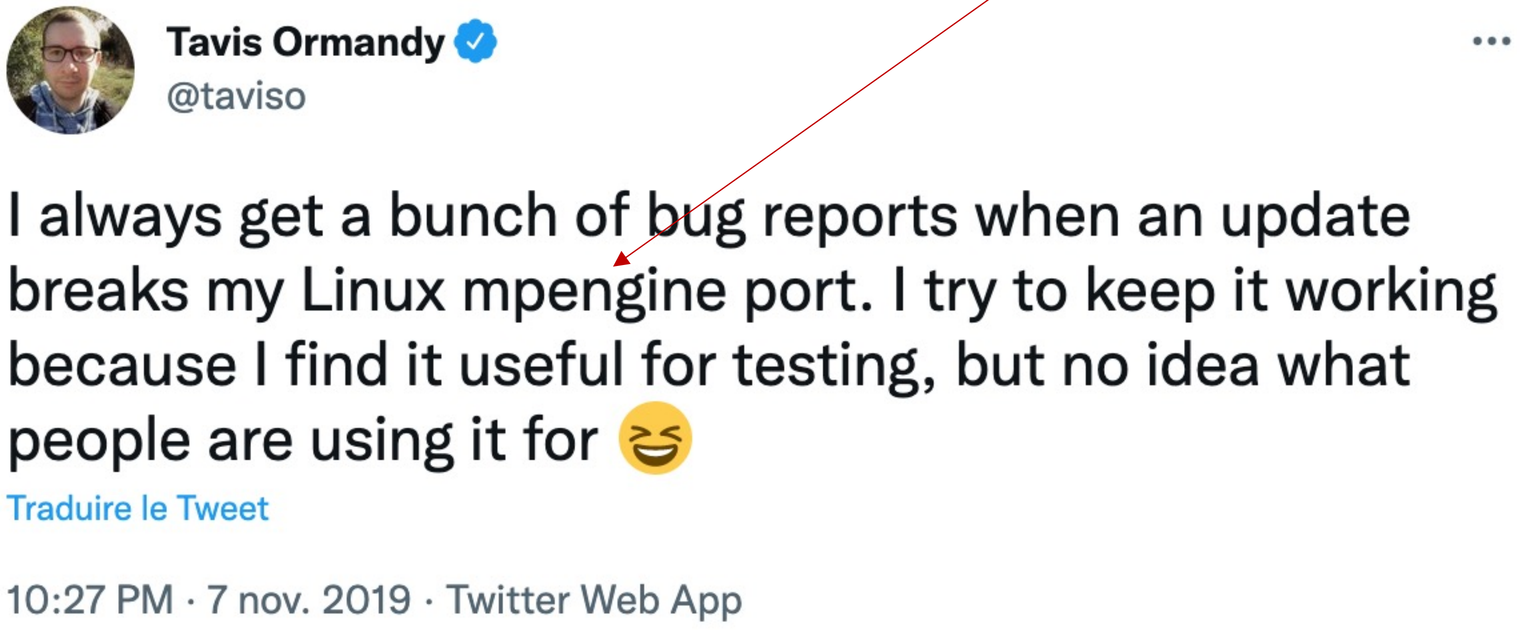
Totally neat and legit :p

Limitations & Future work

- Not for script kiddies :p
- .text section: divide and conquer with functions boundaries
- Optimization: only analyze strings present in source code
- Divide and conquer with differential builds

Conclusion

Windows Defender scan engine



Tavis Ormandy
@taviso

I always get a bunch of bug reports when an update breaks my Linux mpengine port. I try to keep it working because I find it useful for testing, but no idea what people are using it for 😆

Traduire le Tweet

10:27 PM · 7 nov. 2019 · Twitter Web App