I opened Pandora’s box and it was full of obfuscation
~# whoami

Geri Revay 🇮🇱 🇩🇪
Security Researcher at FortiGuard Labs

Ethical Hacking
Malware Research
Threat Intelligence
Twitter: @geri_revay
Agenda

• Introduction
• Obfuscation Techniques in Pandora
• Control-Flow Flattening
• Emulation
• The End
Introduction
Background

CRIME SERVICES ENABLERS

Quality Assurance
Crypters / Packers
Scanners

Hosting
Infections / Drop Zones
Management

Botnet Rentals
Installs / Spam / SEO / DDoS

Money Mules
Accounts Receivable

Consulting

COMPOUNDED CYBERCRIME

Affiliates

Affiliate Programs
Ransomware / Botnets

Victims

Criminal Organizations

Sales, Licensing,
Maintenance

Partnerships

CRIMEWARE PRODUCERS

Exploits
Packers
Special Platforms
Mobile

Source Code

Copy & paste

Junior Developers

Senior Developers

Bank Accounts

Credentials & Data

Digital Real Estate

Victims

Criminal Organizations

Sales, Licensing,
Maintenance

Partnerships

CRIMEWARE PRODUCERS

Exploits
Packers
Special Platforms
Mobile

Source Code

Copy & paste

Junior Developers

Senior Developers

Bank Accounts

Credentials & Data

Digital Real Estate

Victims

Criminal Organizations

Sales, Licensing,
Maintenance

Partnerships

CRIMEWARE PRODUCERS

Exploits
Packers
Special Platforms
Mobile

Source Code

Copy & paste

Junior Developers

Senior Developers

Bank Accounts

Credentials & Data

Digital Real Estate
FortiEDR shows how malware is getting better

Figure 9 - Top malware tactics and techniques in EDR data for 2022-H1
Why Obfuscation?

• No Silver Bullet rather a Ball and Chain
• Cheap for the adversary
• Expensive for the analyst
• Different techniques and different levels of obfuscation
• There are obfuscators for most programming languages
• We will focus on C++
Use Case: Pandora Ransomware

- Analysis: https://www.fortinet.com/blog/threat-research/looking-inside-pandoras-box
- Contains everything a modern ransomware should
- Multi-Threading
- Strong Encryption
- Disable AMSI
- Disable Event Logging
- Unlocking files with Restart Manager
- And all of the world’s Evils…
All of the World’s Evils

Obfuscation Techniques in Pandora
Overview

• Packed with custom UPX
• Strings encoding (14 different decoding functions)
• CALL addresses obfuscated with opaque predicates
• JMP addresses obfuscated with opaque predicates
• Control-Flow Flattening
• Windows API call obfuscation
Opaque Predicates for CALL and JMP addresses

\[ \text{rax} = (*\text{address}\_\text{table}\_\text{base} + 0x260BB2E4) + 0xFFFFFFFFF97CA7C9] \]

- Static data that still calculated in runtime
- Obfuscates connections between basic blocks
Control-Flow Flattening
Control-Flow Flattening

• Obfuscation method

• Cheap for developer, expensive for reverse engineer

• Manipulates the control flow of functions

• Original Basic Block: contain the original logic of the function

• Dispatcher: decides which original basic block comes next

http://tigress.cs.arizona.edu/transformPage/docs/flatten/index.html
Control-Flow Flattening in Real Life
Control-Flow Flattening in Real Life

Gergely Revay @geri_revay - Sep 21
Welcome to Hell! All hail the Great Obfuscator!
How to deal with CFF?
How to deal with CFF?

Pack your stuff and run!
How to deal with CFF?

**Statically**
- Restore control-flow in IDA Pro
  - Emulation
  - Symbolic/Concolic Execution
  - Custom IDApython scripts
- .NET: Restore control-flow in MSIL
  - De4dot and other deobfuscators might be able to do it
  - Custom de4dot plugin

**Dynamically**
- Sandbox detonation
  - Finding IOCs
  - Next stage from memory/file dumps
- Debugging
  - Works but very tedious and slow
  - There might be other Anti-Analysis/Debugging measures in place
Restoring the Control-Flow

- Identify Dispatcher Basic Blocks
- Identify Original Basic Blocks
- Identify State variable
- Map States to OBBs
- Map Next States to OBBs
- Reconstruct code based on recovered paths

- Added fun in Pandora: Dispatcher is also spread around in multiple
Pandora: Dispatcher
Pandora: Some Heuristics

- Manipulate state variable with cmovX or setlX
- Dispatcher BB starts with cmp or xor
- In case of xor a cmp follows
- The cmp instruction has the state value

Original BB or Code BB ends in relative jump
Dispatcher BB ends in jump to register
Decision in OBBs

- If OBB would end in a decision, that is moved to another BB
- Some comparison (here test ecx, 1) sets the next state
- These decisions needs to be tracked to learn potential next states
Emulation

Encouragement and cautionary tale
Emulation: the good and evil

• As many complex analysis technique, emulation can be a great help and an enormous time waster

• In practice, the goal is to find the places where it is useful

• Problems with emulation:
  • It does not really run
  • Dependency on other functions
  • Dependency on APIs and libraries

https://www.previewsworld.com/SiteImage/MainImage/STL120308.jpg
Pandora: where emulation worked well

• Opaque Predicates
  ‘Static’ calculated in run-time
Pandora: Opaque Predicates

```python
import flare_emu
from ida_funcs import *

def call_hook(address, arguments, functionName, userData):
    print(f"[+] CALL at 0x{hex(address)}")
    #check if call target a register
    if eh.analysisHelper.getOpndType(address, 0) != eh.analysisHelper.o_reg:
        return

    operand_name = eh.analysisHelper.getOperand(address, 0)
    operand_value = eh.getRegVal(operand_name)
    print(f"[+] {} = 0x{operand_value}"

if __name__ == '__main__':
    ea = get_screen_ea()
    print('[+] Starting emulation')
    eh = flare_emu.EmuHelper()
    function = get_func(ea)
    eh.emulateRange(function.start_ea, callHook=call_hook)
```
Pandora: where emulation worked well

- String decryption
- 14 different decryption functions, same algorithm different constants
- Iterative process
  - First debugging, later 'visual inspection'
def call_hook(address, arguments, functionName, userData):
    print("[+] CALL at 0x{}").format(eh.hexString(address))
    # check if call target a register
    if eh.analysisHelper.getOpndType(address, 0) != eh.analysisHelper.o_reg:
        return
    # comment to call function: args, function addr
    operand_name = eh.analysisHelper.getOperand(address, 0)
    operand_value = eh.getRegVal(operand_name)

    fname = ""
    res = ""
    # check if points to the jump table
    if eh.analysisHelper.getMnem(operand_value).lower() == "jmp":
        fname = eh.analysisHelper.getName(eh.analysisHelper.getOpndValue(operand_value, 0))
        print("[+] API call found: {}".format(fname))
    else:
        fname = eh.analysisHelper.getName(operand_value)
        if "mw_decrypt_str" in fname:
            res = decrypt(arguments, fname)
            print("[+] Decrypted string: 0x{} {}".format(eh.hexString(address), res))

    # if call target is not a start of a function then turn it to a function
    # 00007FF6B6F947A0
    if idaapi.get_func(operand_value) == None:
        print("[+] Creating function at 0x{:x}".format(operand_value))
        ida_funcs.add_func(operand_value)
Pandora: String decryption

def decrypt(argv, fname):
    print([+] Decrypting ...)
    myEH = flare_emu.EmuHelper()
    myEH.emulateRange(myEH.analysisHelper.getNameAddr(fname), registers = {"arg1":argv[0], "arg2":argv[1], "arg3":argv[2], "arg4":argv[3]})
    return myEH.getEmuString(argv[0])

00007FF68F96770 mov rdx, cs:qword_7FF68F9AC0
00007FF68F9676D add rdx, rbp
00007FF68F96770 call rax

Decrypted str: 'ThisIsMutexa'

100.00% (2032,1229) (3,359) 0000E670 00007FF68F96770: main+80 (Synchronized with Hex)
Pandora: I wasted my time so you don’t have to

• I worked on CFF resolution for pandora

• Problem:
  • Emulation was not able to recover next states from decision OBBs
  • Emulating all function calls is risky
  • Decisions might depend on these calls
  • Pandora has a complex way to calculate the values of next states

• Conclusion
  • In practice (where time is money) it is not worth the time
  • Analysis can be done in a debugger in less time
  • In other malware with less complex obfuscation might worth is
Thanks and Q’n’A

Geri Revay 🇵🇸 🇩🇪 Security Researcher at FortiGuard Labs

Ethical Hacking
Malware Research
Threat Intelligence
Twitter: @geri_revy
References

https://www.fortinet.com/blog/threat-research/looking-inside-pandoras-box

https://www.fortinet.com/blog/threat-research/Using-emulation-against-anti-reverse-engineering-techniques

https://research.openanalysis.net/pandora/ransomware/malware/unpacking/dumpulator/emulation/2022/03/19/pandora_ransomware.html

https://github.com/mandiant/flare-emu