



RE  
RE  
RE  
RE

# Breaking Encrypted USB Drives with Time-Travel Debugging

Xusheng Li | Vector 35

# About Me

1. Binary Ninja developer at Vector 35
2. Lead debugger development (including TTD integration)
3. Reviewer at Paged Out! magazine
4. Admin/reviewer at crackmes.one
5. 4x Flare-On finisher, Shellphish CTF team member

# About Me

1. Binary Ninja developer at Vector 35
2. Lead debugger development (including TTD integration)
3. Reviewer at Paged Out! magazine
4. Admin/reviewer at crackmes.one
5. 4x Flare-On finisher, Shellphish CTF team member
- 6. Security researcher (new!)**

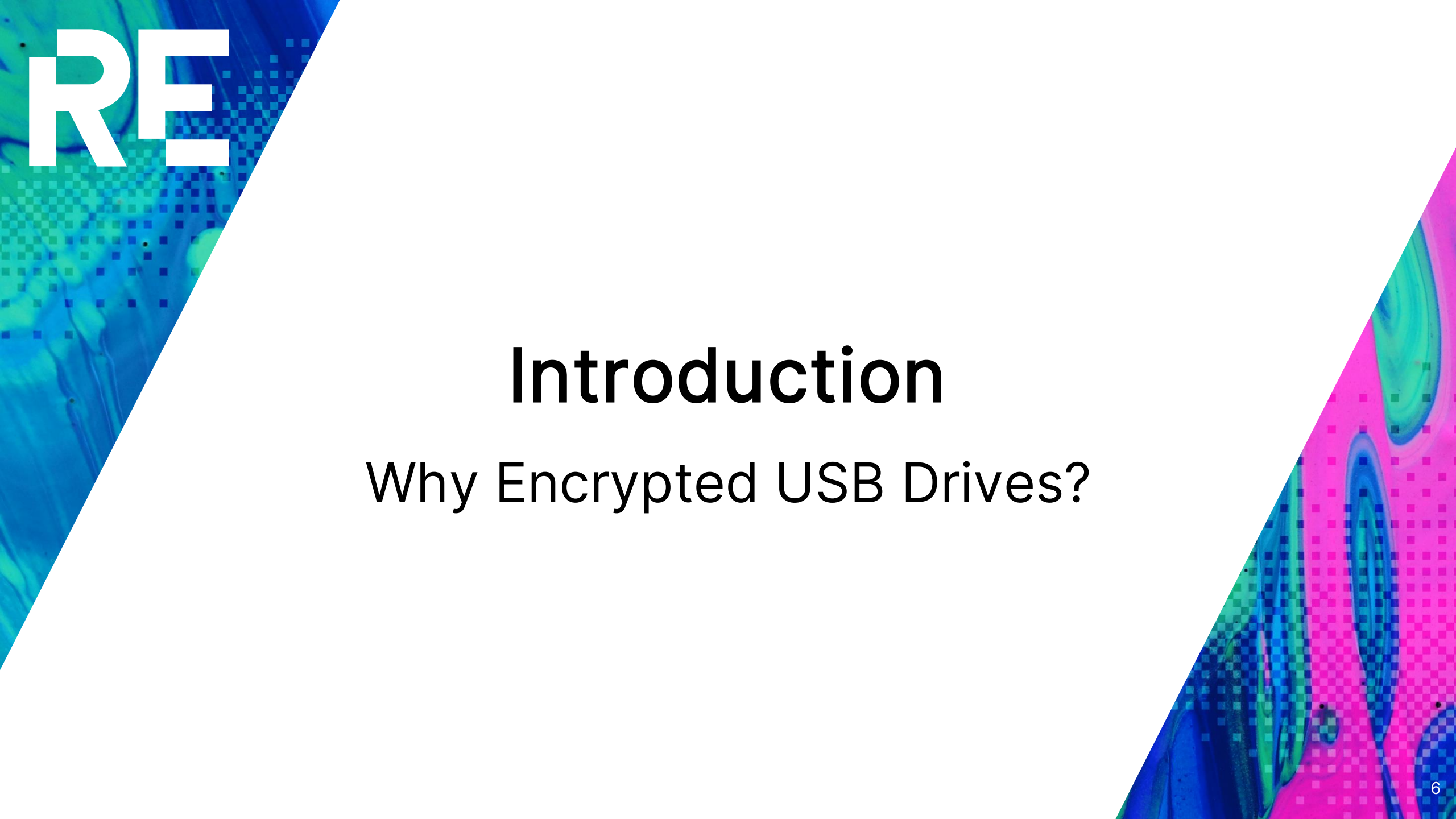
# What to Expect

**Don't trust "encrypted" USB drives**  
Verify vendor claims yourself

**TTD is a powerful, underutilized RE tool**  
Integrate it into your RE workflow

# Agenda

1. Introduction & Threat Model
2. Vulnerability #1: No Encryption At All
3. Vulnerability #2: Broken Software Encryption
4. Introduction to Time-Travel Debugging
5. Vulnerability #3: Password Disclosure
6. Lessons Learned & Wrap-up



# RE

## Introduction

Why Encrypted USB Drives?

# The Suspicious Security Design

I came across an encrypted USB drive

It can be unlocked by fingerprint OR password

# The Suspicious Security Design

I came across an encrypted USB drive

It can be unlocked by fingerprint OR password

In Full-Disk Encryption (FDE), the initial unlocking requires a password

➔ That is why you need to type in the password after a phone reboot

# The Suspicious Security Design

I came across an encrypted USB drive

It can be unlocked by fingerprint OR password

In Full-Disk Encryption (FDE), the initial unlocking requires a password

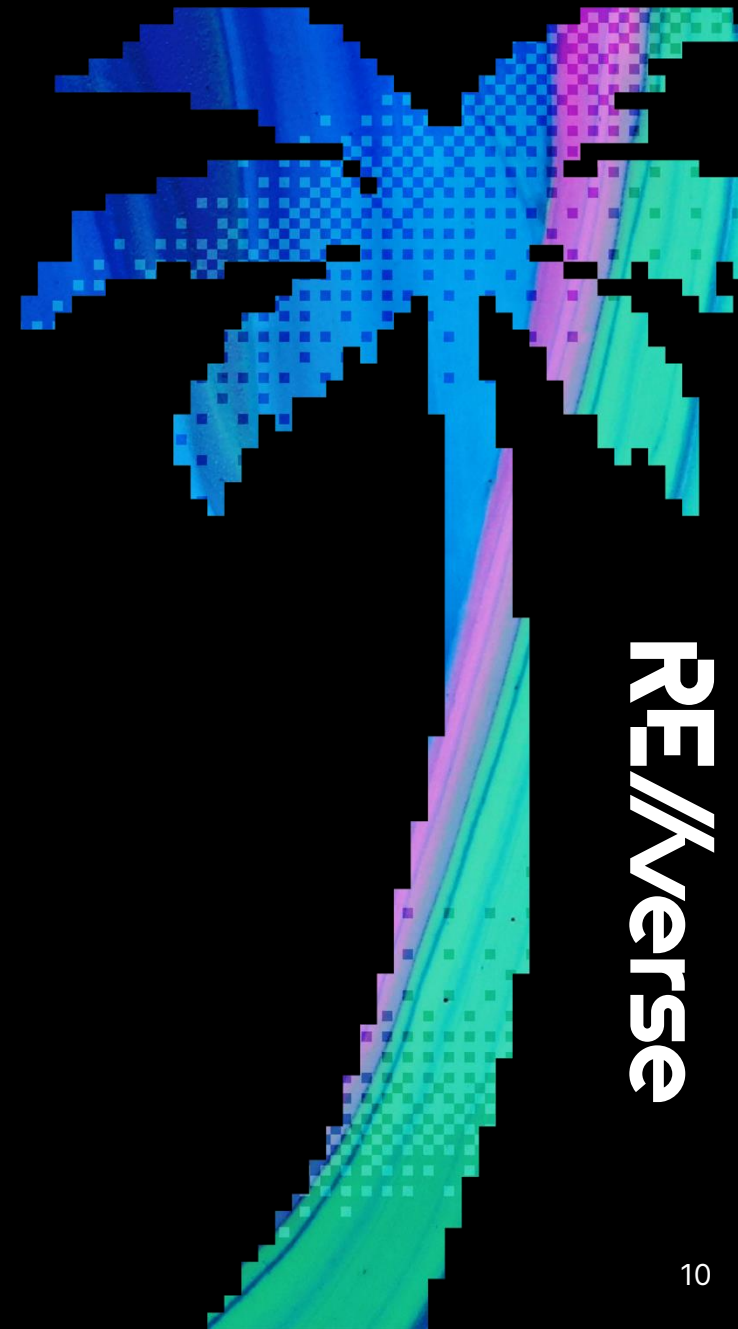
→ That is why you need to type in the password after a phone reboot

Not the case for the device...

# Research Targets

1. Lenovo ThinkPlus FU100 - Fingerprint encrypted USB
2. Netac (with NetacLockFile software)
3. Lenovo ThinkPlus TSD303 - Fingerprint encrypted USB

And many more affected devices





# RE

## Vulnerability #1

What's Inside?

# Case Study: Lenovo ThinkPlus FU100

1. Marketed as "hardware encrypted" USB drive
2. Fingerprint or password authentication
3. Claims AES-256 encryption



Image Credits: Alibaba.com

# Case Study: Lenovo ThinkPlus FU100

1. Marketed as "hardware encrypted" USB drive
2. Fingerprint or password authentication
3. Claims AES-256 encryption

Very Popular!

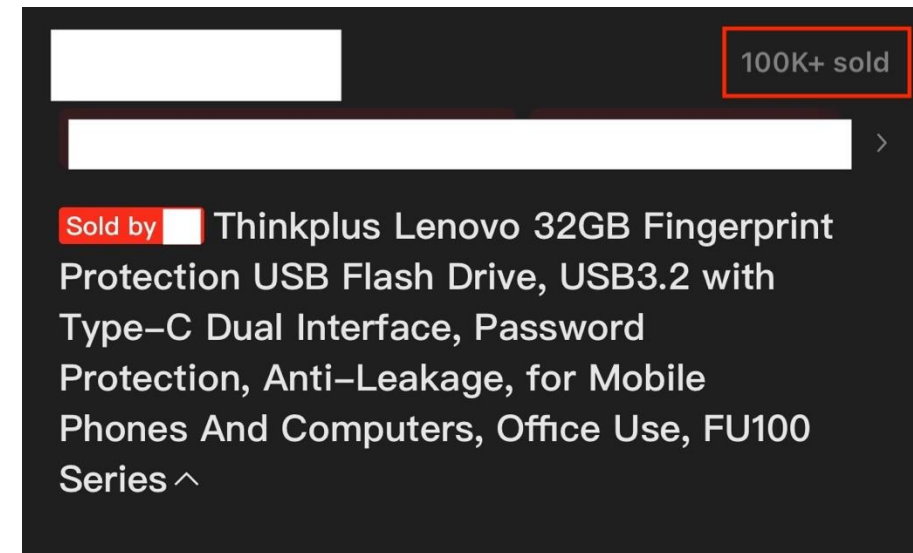
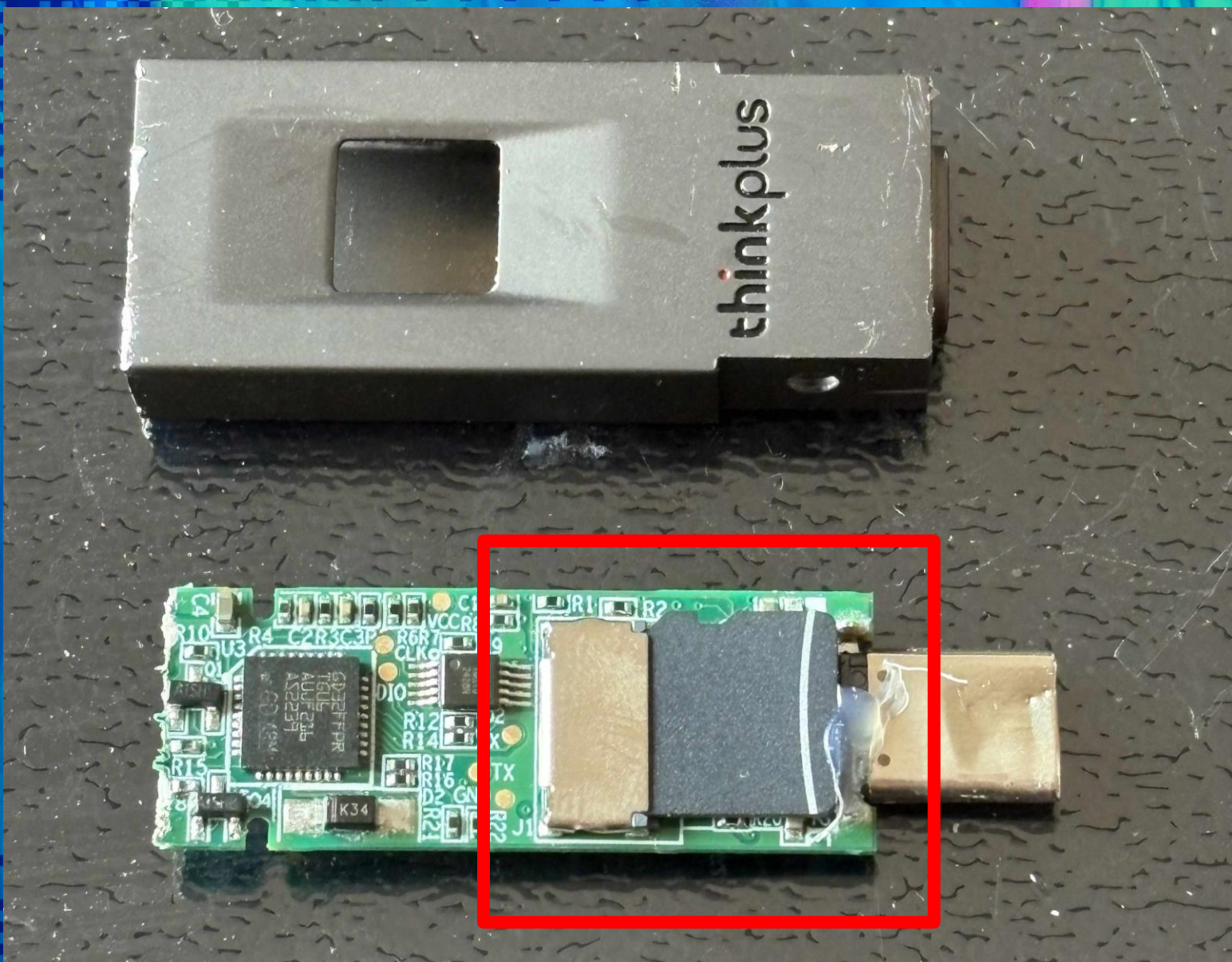
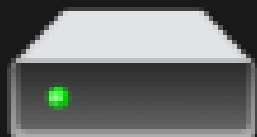
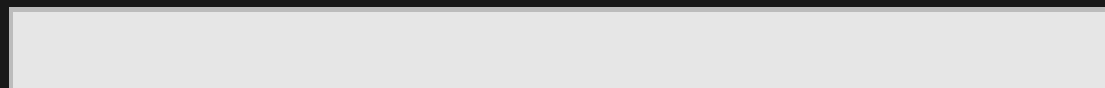


Image Credits: Alibaba.com

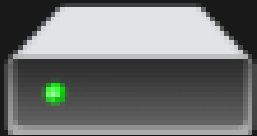




FU100 (G:)



58.2 GB free of 58.2 GB



FU100 (G:)

58.2 GB free of 58.2 GB

| Name                                                                                              | Date modified     | Type        | Size     |
|---------------------------------------------------------------------------------------------------|-------------------|-------------|----------|
|  thinkplus.exe | 5/27/2025 9:06 AM | Application | 3,637 KB |

# The Reality -- No Encryption At All

It's just an SD card in a fancy case

No encryption (of data) whatsoever

Fingerprint gates USB connection

A photograph of three pairs of pliers with black handles and red accents, lying on a light-colored wooden surface. The pliers are arranged diagonally from the bottom left towards the top right. A blue rectangular box is overlaid on the right side of the image, containing the title text.

# Advanced Penetration Testing Toolkit

# Interim Mitigation

Reported on Aug 2<sup>nd</sup>

Temporary fix available on Aug 27<sup>th</sup>

(\*) Proper fix to follow in December



Epoxy cover on the SD card (left)





# RE

## **Vulnerability #2**

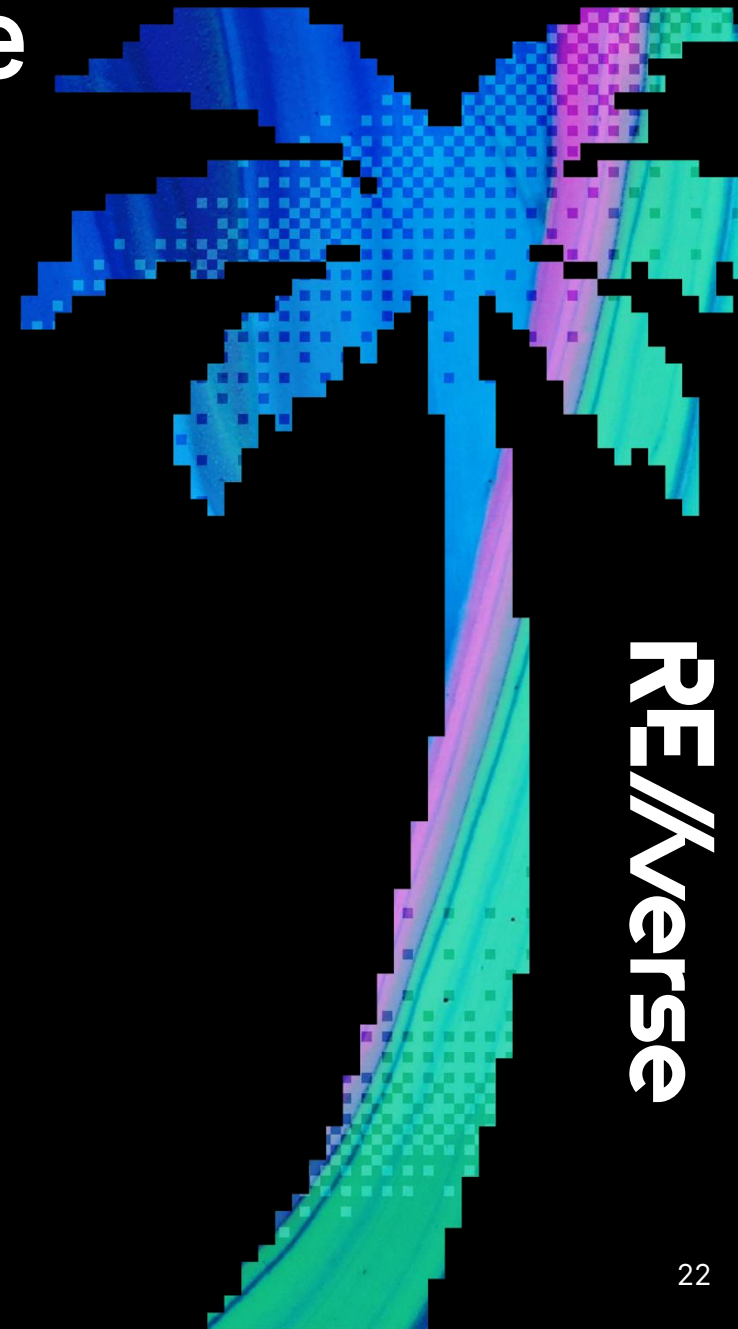
### Broken Software Encryption

# Case Study: Netac NetacLockFile

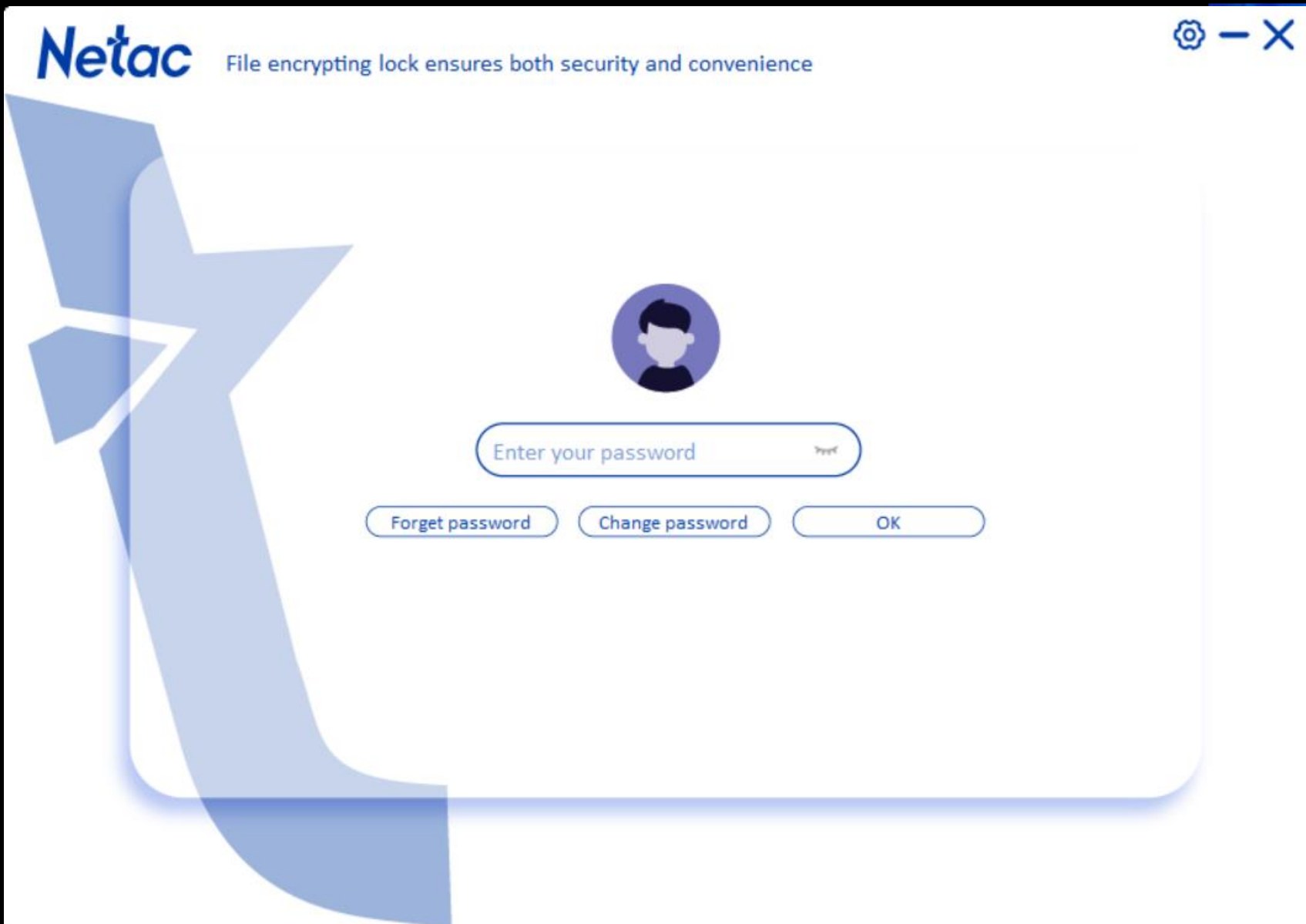
Software-based encryption tool for Netac drives

Promises to encrypt files with user password

Creates "encrypted" container files



RE//verse



RE//verse



# RE//verse

# Not All Bytes are Encrypted...

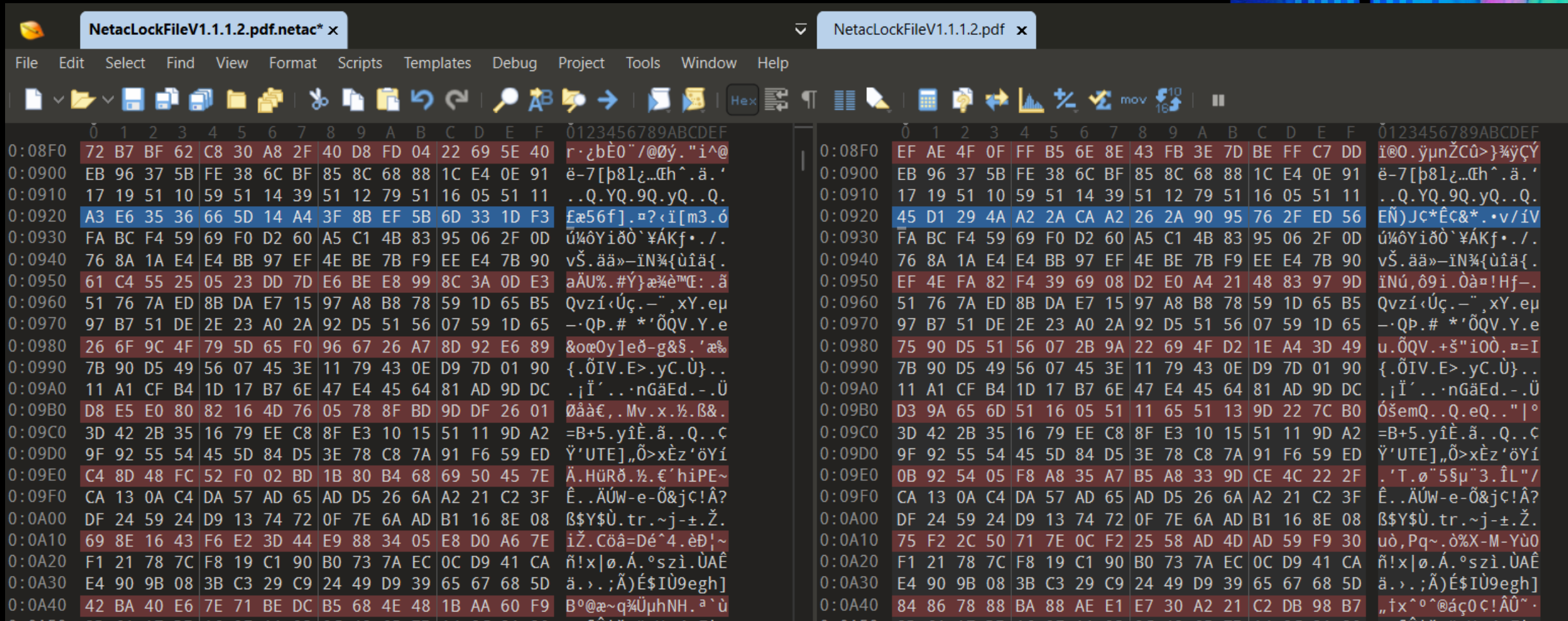
→ NetacLockFil....2.pdf.netac X +

Raw ▾ Hex ▾

00000000 20 20 20 20 20 20 20 20-00 00 00 00 00 00 00 00-01 09 00 00 0d 00 00 00-00 00 00 00 00 00 00 00 .....  
00000020 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 00 00 00 00 .....  
00000040 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00-61 75 74 68 6f 72 00 00 .....author..  
00000060 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00-bf cd f0 68 00 00 00 00 .....h...  
00000080 43 38 32 33 33 37 45 38-31 41 32 37 30 34 44 33-34 44 30 39 39 34 42 33-44 35 38 39 39 46 41 34 C82337E81A2704D34D0994B3D5899FA4  
000000a0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 00 00 00 .....  
000000c0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 00 00 .....  
000000e0 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 00 00 .....  
00000100 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 00 00 .....  
00000120 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 00 00 .....  
00000140 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 00 00 .....  
00000160 00 00 00 00 00 00 00 00-02 00 00 00 00 00 00 00-00 00 00 00 00 00 00-25 50 44 46 2d 31 2e 37 .....%PDF-1.7  
00000180 0d 25 e2 e3 cf d3 0d 0a-31 36 38 20 30 20 6f 62-6a 0d 3c 3c 2f 4c 69 6e-16 1b a6 21 4b d2 87 98 .%.168 0 obj.<</Lin...!K...  
000001a0 a2 d8 84 88 90 00 8e 3d-31 38 2f 4f 20 31 37 30-2f 45 20 31 32 30 36 35-30 2f 4e 20 31 31 2f 54 .....=18/0 170/E 120650/N 11/T  
000001c0 20 39 35 39 31 33 36 2f-81 1b 40 2b 2c 5b 30 e2-fe 51 72 64 62 75 44 30-6e 64 6f 62 6a 0d 20 20 959136/..@+, [0..QrdbuD0ndobj.  
000001e0 20 20 20 20 20 20 20 20-20 20 0d 0a 31 37 39 20-30 20 6f 62 6a 0d 3c 3c-83 9a 36 43 6e 35 13 b6 ..179 0 obj.<<..6Cn5..  
00000200 fe 81 e8 46 84 c1 a8 1d-6f 6c 75 6d 6e 73 20 35-2f 50 72 65 64 69 63 74-6f 72 20 31 32 3e 3e 2f ...F....olumns 5/Predictor 12>>/  
00000220 46 69 6c 74 65 72 2f 46-77 16 15 ba 05 ae 87 5f-1b 53 e2 84 a0 8c 40 49-37 42 37 32 33 34 32 37 Filter/Fw.....S....@I7B723427  
00000240 46 33 36 34 41 35 33 38-30 36 36 34 32 30 38 34-33 34 39 30 46 39 38 3e-3b a2 cf 53 5e 43 6a 60 F364A538066420843490F98>;..S^Cj`  
00000260 2d 20 74 27 d5 f3 6c 98-33 39 46 45 46 43 46 32-42 41 35 36 39 41 33 45-37 3e 5d 2f 49 6e 64 65 - t'..l.39FEFCF2BA569A3E7>]/Inde  
00000280 78 5b 31 36 38 20 32 36-7e f8 bf 64 22 ad f9 57-b2 cb 94 ee a7 c9 72 80-65 6e 67 74 68 20 37 35 x[168 26~..d"..W.....r.length 75  
000002a0 2f 50 72 65 76 20 39 35-39 31 33 37 2f 52 6f 6f-74 20 31 36 39 20 30 20-79 b0 12 fd ea cc 2b a0 /Prev 959137/Root 169 0 y.....+.  
000002c0 3d b3 1c 74 63 a9 b7 d4-58 52 65 66 2f 57 5b 31-20 33 20 31 5d 3e 3e 73-74 72 65 61 6d 0d 0a 68 =..tc...XRef/W[1 3 1]>>stream..h  
000002e0 de 62 62 64 60 10 60 60-1f 8c ce 78 26 d2 25 d1-60 0c fe 1c dd ba fe f3-72 c4 80 48 a6 d3 60 f6 .bbd`.``...x&.%.`.....r..H..`

RE//verse

# Only 1/3 are Encrypted!



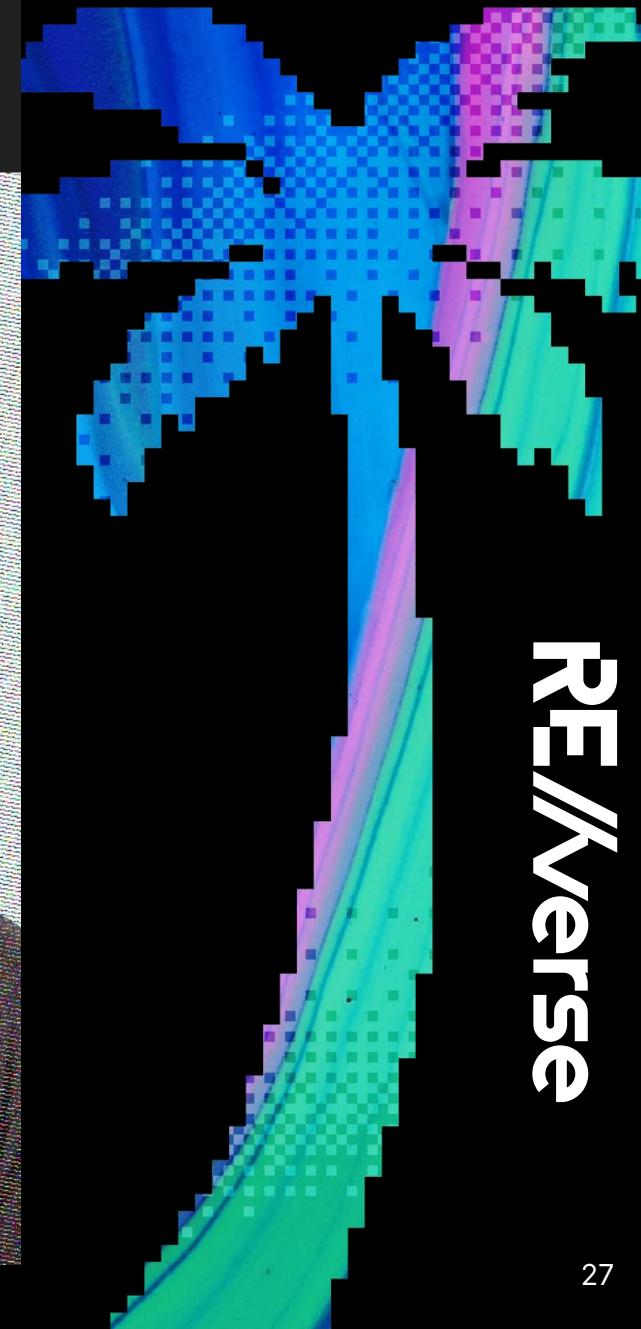
# ORIGINAL

Before "encryption"



# "ENCRYPTED"

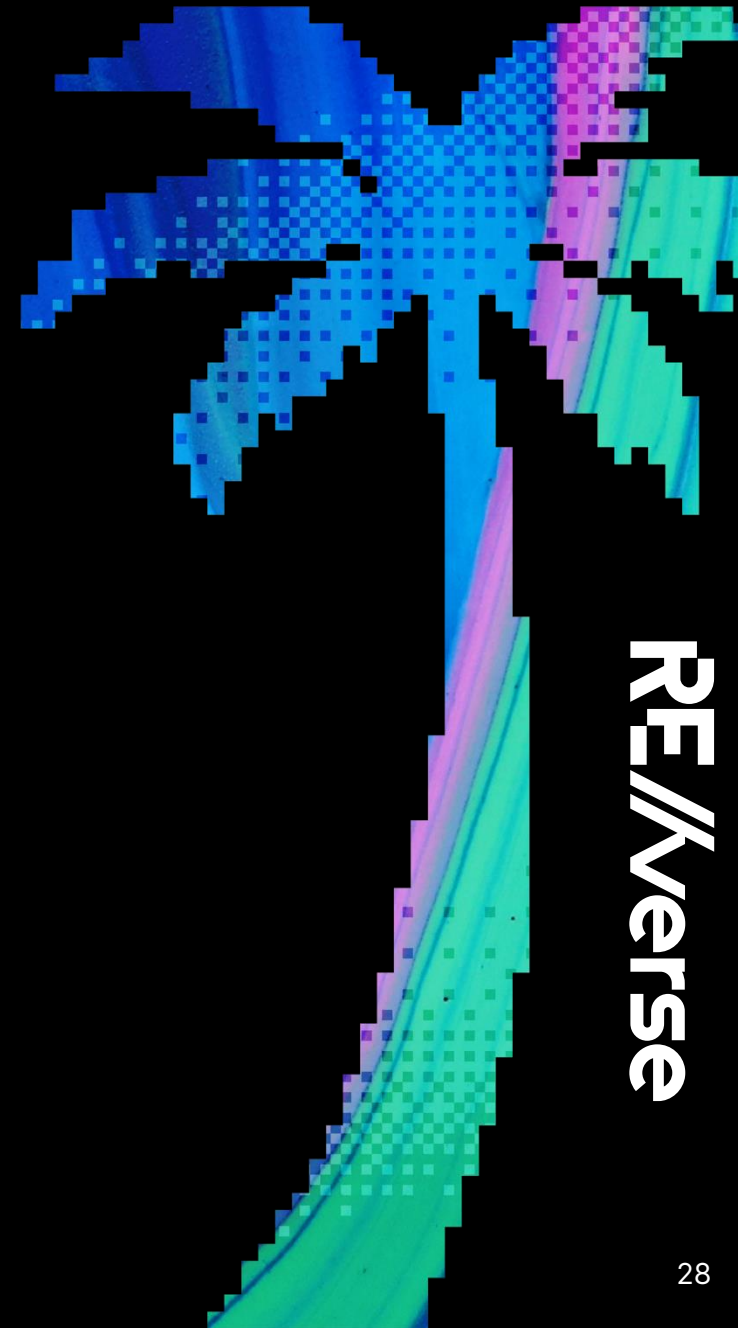
16 of every 48 bytes encrypted



RE//verse

# Why?

No idea



RE//verse

# Why?

No idea

Survey Question

Personal Experiences

Coding Language

Human Activity

Odd Behavior

+ 5

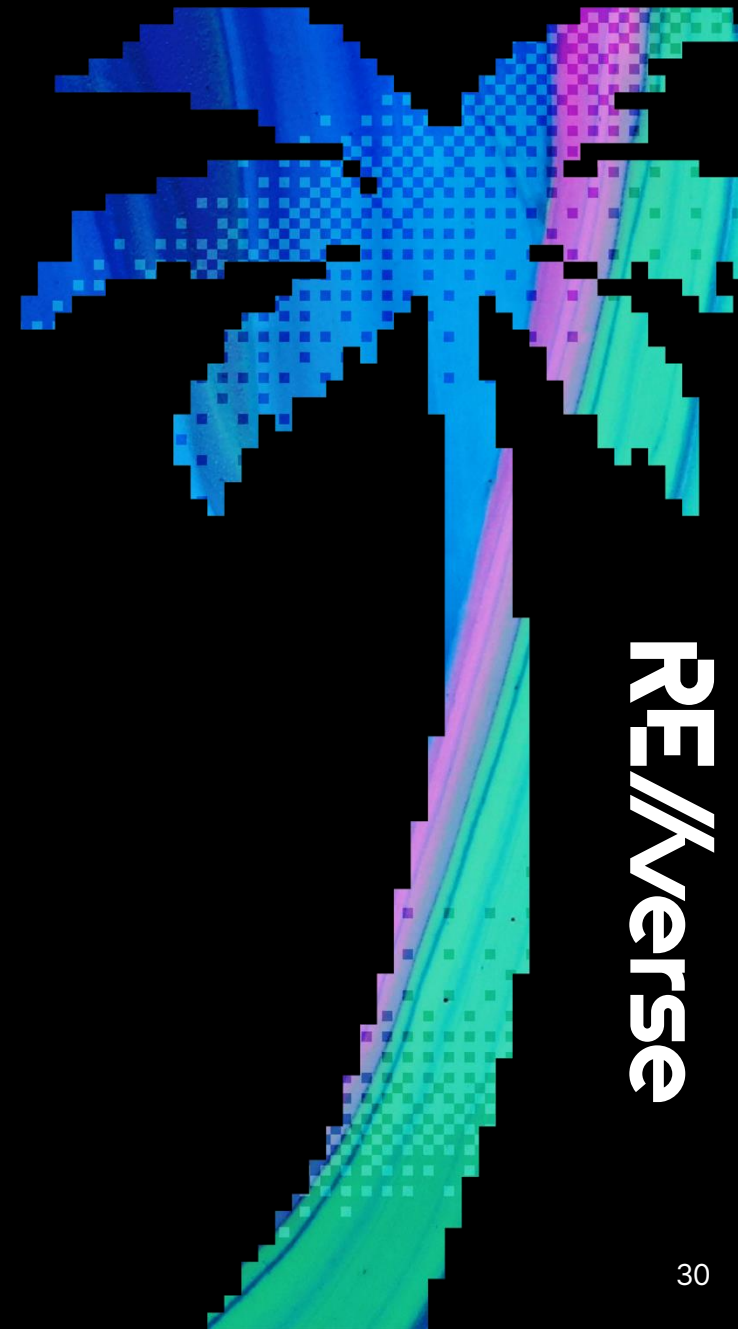
**What's the weirdest thing you've seen someone do while coding?**

RE//verse

# Is the 1/3 PROPERLY Encrypted?

I wish to examine the crypto system to see if these are secure

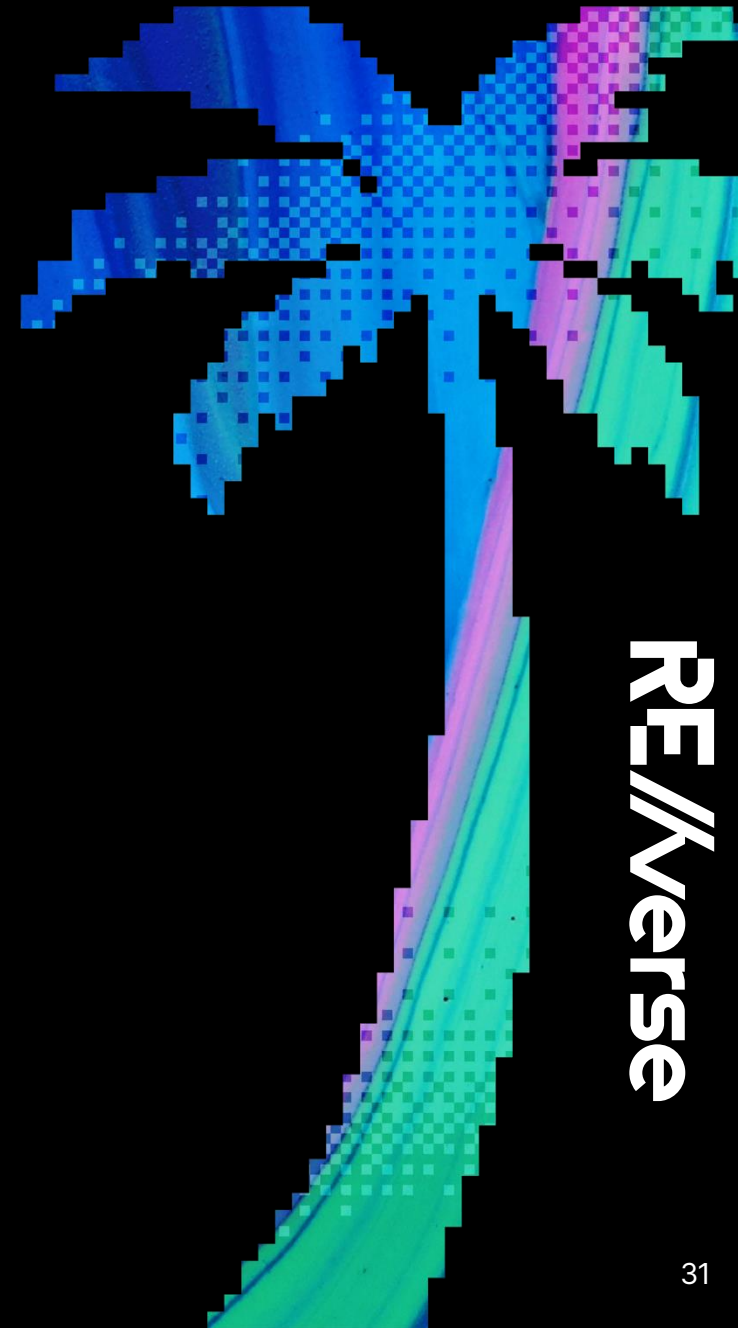
This determines if this is a trivial fix or something bigger



# Traditional Approach

1. Reverse engineer the binary
2. Find the cryptographic function
3. Find the encryption keys

- ^ Manageable, but time-consuming
- ^ I will show a new method today!



RE//verse



# RE

## Time-Travel Debugging

The Game Changer for RE

# What is Time-Travel Debugging?

1. Record program execution into a trace file
2. Replay execution history deterministically
3. Step forward AND backward through execution

TTD is also called record & replay in the literature

# How Do You Debug a Trace?

Replay engine creates the illusion of debugging

Example: Set a breakpoint at 0x123456 and resume

What happens: replay engine scans the trace and finds the next time the instruction at 0x123456 is executed

# TTD Solutions

1. WinDbg TTD: free, from Microsoft, for Windows ← we will be using this
2. rr (<https://rr-project.org/>): open-source, from Mozilla, for Linux
3. esReverse (<https://eshard.com/esreverse>): commercial, full system record & replay
4. Undo (<https://undo.io/>): commercial, for Linux
5. Panda-re (<https://github.com/panda-re/panda>): open-source, academic, full system record & replay

Detailed list: <https://github.com/xusheng6/awesome-ttd?tab=readme-ov-file#ttd-solutions>

# TTD Advantages

1. Full execution replay — never miss a moment
2. Deterministic reproduction of bugs
3. Programmatic access of all trace information
4. Better suited for AI integration

# TTD Limitations

1. Performing queries on large traces can be slow  
TTD is NOT slow on small or medium-sized traces  
It is FASTER than your expectation  
Even for giant traces, it is still often worthwhile compared to what you get
2. TTD does not work for everything  
It works for Windows/Linux/Qemu  
No IoT/mobile/macOS support (yet!)

# TTD Philosophy

With TTD, you use breakpoints/execution less

More often you run queries to find the event you care about:

1. The last time this address is written
2. All calls to a specific function
3. Next time the register is used
4. Every time this instruction is executed

# The Killer Feature: Backtracking

We can track the formation of a buffer

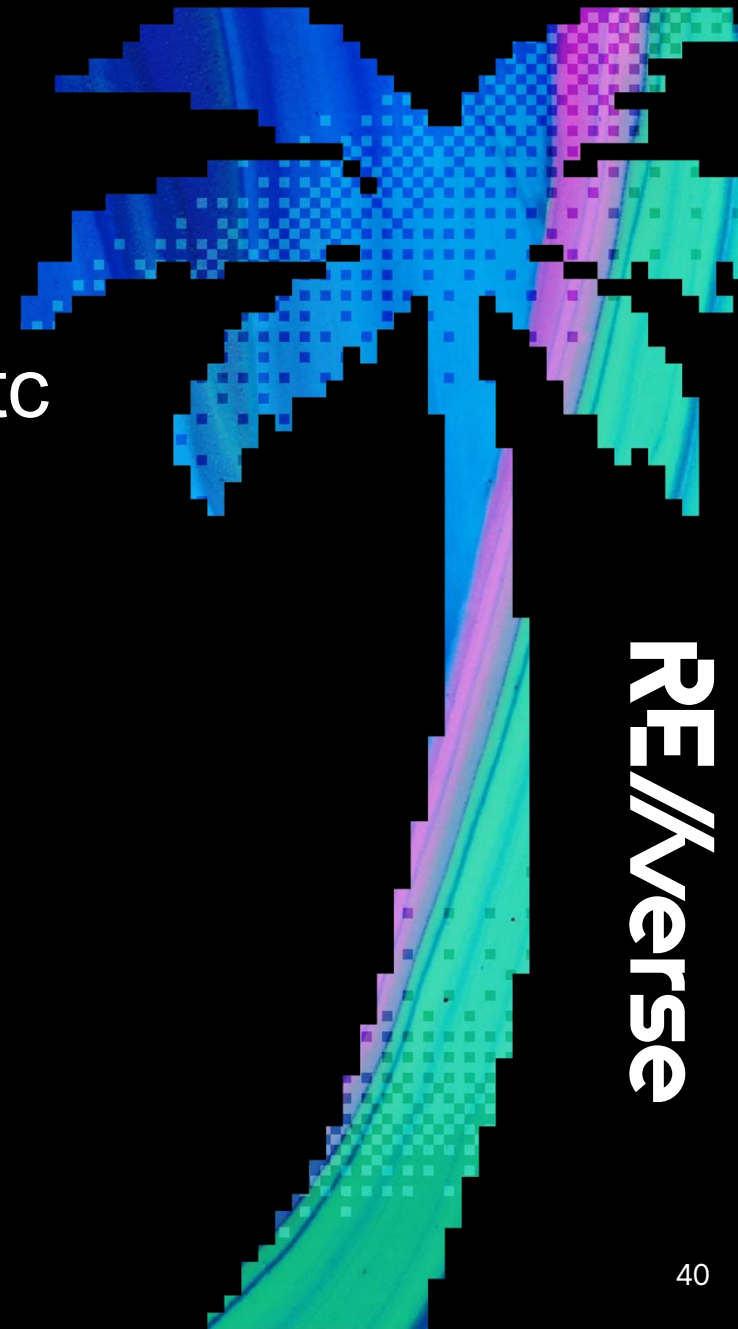
Find an encryption key? Find its source

Observe decrypted data? Find the decryption routine

Impossible with traditional debugging!

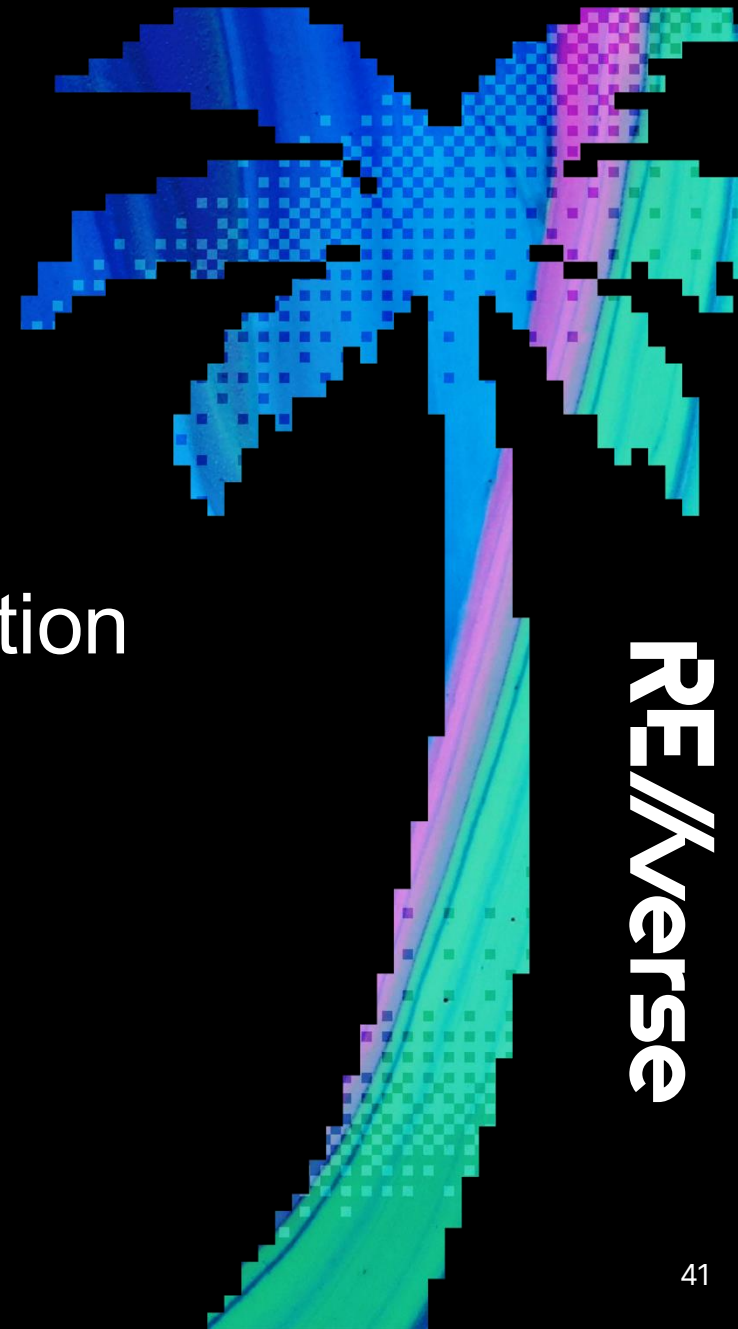
# TTD in Binary Ninja

1. Native TTD integration in the debugger
2. Works with WinDbg TTD, rr, esReverse, etc
3. Full reverse execution support
4. Memory RWX query
5. Function call query
6. Dedicated UI, API, and unique analysis

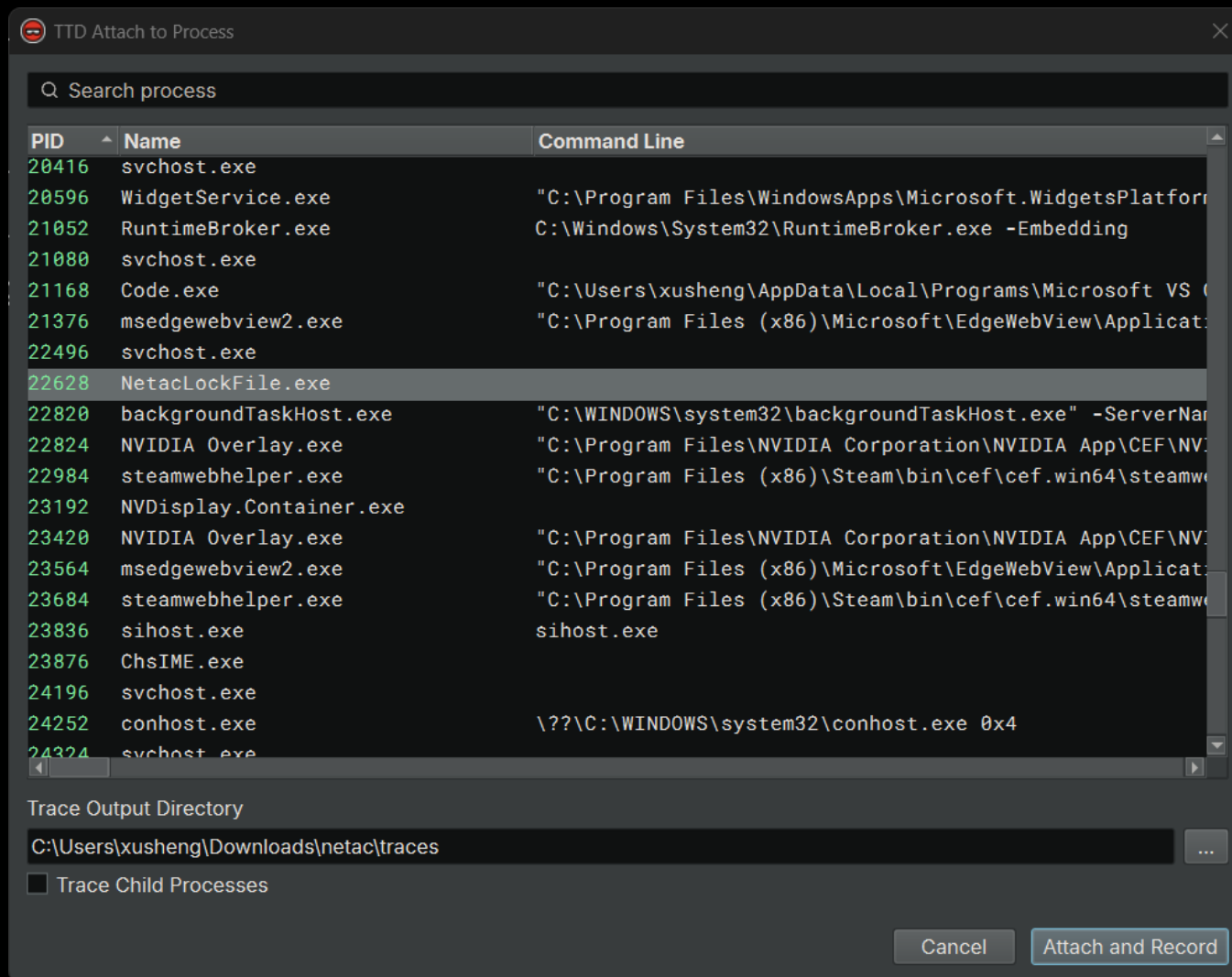


# Analyzing the Crypto System

1. Get a TTD recording of the file encryption
2. Find calls to `kernel32!WriteFile`
3. Find buffers being written
4. Backtrace the buffer to find the crypto function
5. Find the encryption key



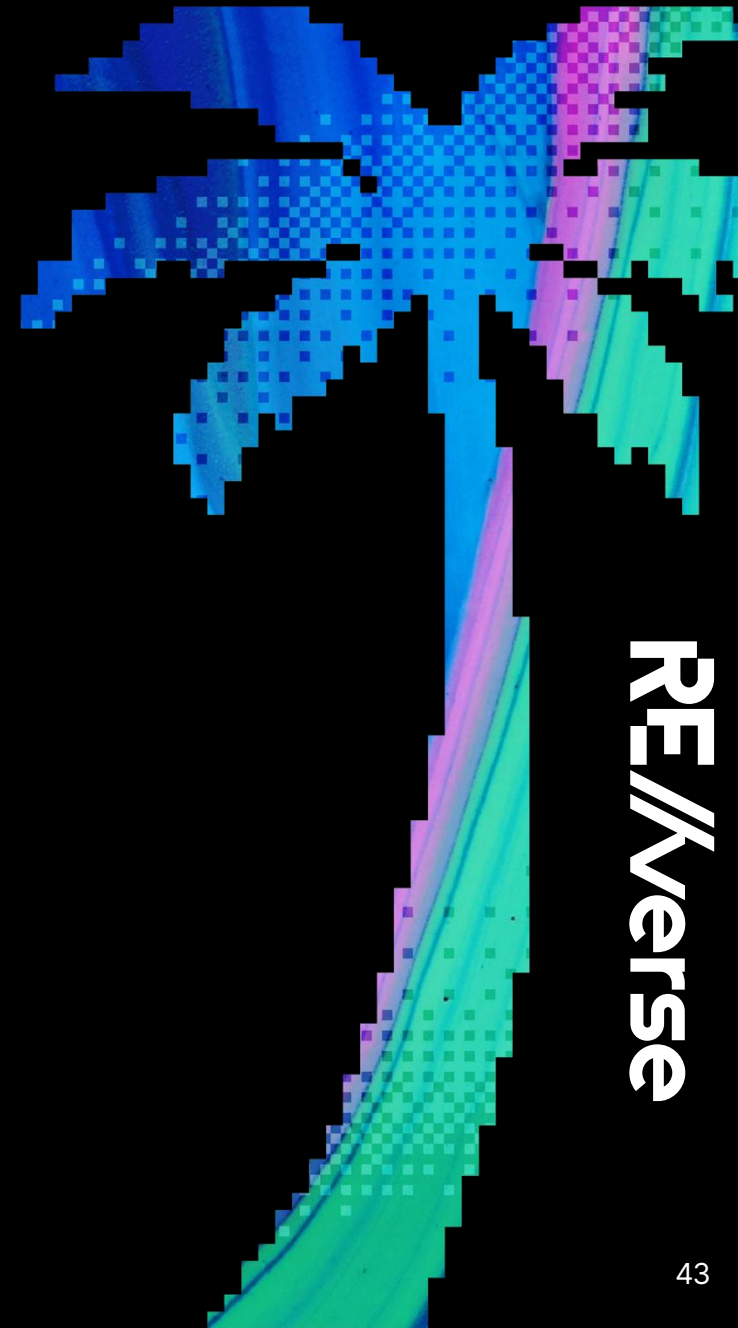
# Attach To a Running Process and Start Recording



RE//verse

# TTD Trace Is Not Huge

| Name                                                                                                  | Date modified     | Type                  | Size       |
|-------------------------------------------------------------------------------------------------------|-------------------|-----------------------|------------|
| ▼ Today                                                                                               |                   |                       |            |
|  NetacLockFile01.out | 1/30/2026 4:40 PM | Wireshark capture ... | 3 KB       |
|  NetacLockFile01.run | 1/30/2026 4:40 PM | RUN File              | 471,040 KB |



RE//verse

NetacLockFile.exe.bndb — Binary Ninja 5.3.9011-dev

File Edit View Analysis Debugger Plugins Window Help

← → NetacLockFile.exe.bndb +

# Debugger

DBGENG\_TTD

Registers Breakpoints

Q Search registers

| Name | Value      | Hint   |
|------|------------|--------|
| gs   | 0x2b       | '+'    |
| fs   | 0x53       | 'S'    |
| es   | 0x2b       | '+'    |
| ds   | 0x2b       | '+'    |
| edi  | 0x0        |        |
| esi  | 0xffffffff |        |
| ebx  | 0x772c7240 | '@r,w' |
| edx  | 0x772442f0 |        |
| ecx  | 0x159cfbed |        |
| eax  | 0x7002c    |        |
| ebp  | 0x1b3f754  |        |
| eip  | 0x77187000 |        |
| cs   | 0x23       | '#'    |
| efl  | 0x246      |        |
| esp  | 0x1b3f678  |        |
| ss   | 0x2b       | '+'    |
| dr0  | 0x0        |        |
| dr1  | 0x0        |        |

Cross References

Filter (0)

PE Graph Disassembly

int32\_t sub\_77187000()

→ 77187000 sub\_77187000:

→ 77187000 jmp far 0x77187009, 0x33

Console Python Debugger Target

2026-01-30 16:40:25 ~eCustomTableWidget 31

2026-01-30 16:40:25 ~eCustomTableWidget 31

2026-01-30 16:40:25 ~dCustomTableView 37 beg destructor function

2026-01-30 16:40:25 ~dCustomTableView 37 beg destructor function

2026-01-30 16:40:26 ~dCustomTableView 113 end destructor function

2026-01-30 16:40:26 ~dCustomTableView 113 end destructor function

2026-01-30 16:40:26 ~pwdLoginDlg 859

2026-01-30 16:40:26 ~pwdLoginDlg 859

TTD: End of trace reached.

(5e34.3978): Break instruction exception - code 80000003 (first/second chance not available)

Time Travel Position: 83F96:0

eax=0007002c ebx=772c7240 ecx=159cfbed edx=772442f0 esi=ffffffff edi=00000000

eip=77187000 esp=01b3f678 ebp=01b3f754 iopl=0 nv up ei pl zr na pe nc

cs=0023 ss=002b ds=002b es=002b fs=0053 gs=002b efl=00000246

77187000 ea097018773300 jmp 0033:77187009

Unable to load image F:\NetacLockFile.exe, Win32 error 0n2

>>>

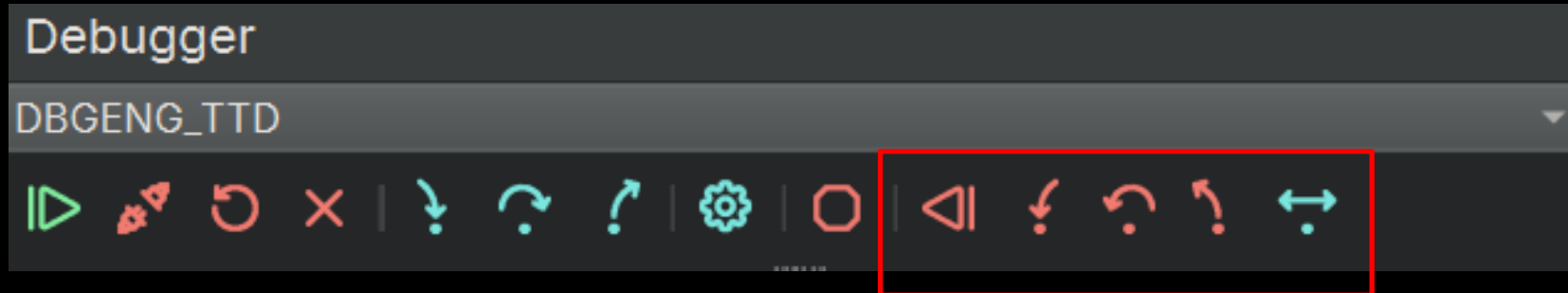
Stack Trace

| State    | Thread               | Frame # | Module        | Function                         |
|----------|----------------------|---------|---------------|----------------------------------|
| Unfrozen | (*) 0x1 @ 0x77187000 | 0       | <unknown>     | 0x77187000                       |
|          |                      | 1       | ntdll         | NtTerminateProcess + 0xc         |
|          |                      | 2       | ntdll         | RtlExitUserProcess + 0x184       |
|          |                      | 3       | KERNEL32      | ExitProcessImplementation + 0x12 |
|          |                      | 4       | NetacLockFile | NetacLockFile + 0x80d205         |
|          |                      | 5       | NetacLockFile | NetacLockFile + 0x80d1c3         |
|          |                      | 6       | NetacLockFile | NetacLockFile + 0x80d2d5         |
|          |                      | 7       | NetacLockFile | NetacLockFile + 0x80058c         |
|          |                      | 8       | KERNEL32      | BaseThreadInitThunk + 0x19       |
|          |                      | 9       | ntdll         | _RtlUserThreadStart + 0x2b       |
|          |                      | 10      | ntdll         | _RtlUserThreadStart + 0x1b       |
| Unfrozen | 0x0 @ 0x77187007     |         |               |                                  |
| Unfrozen | 0x2 @ 0x77187007     |         |               |                                  |
| Unfrozen | 0x4 @ 0x77187007     |         |               |                                  |
| Unfrozen | 0x5 @ 0x77187007     |         |               |                                  |

Analysis Phase 1...Analyzing (55501/55512)

2 windows-x86 0x77187000-0x77187007 (0x7 bytes)

# Reverse execution buttons



Buttons only available during TTD

RE//verse

# Getting All Calls to kernel32!WriteFile

TTD Calls

Query 1 ✕

▼ Query Parameters

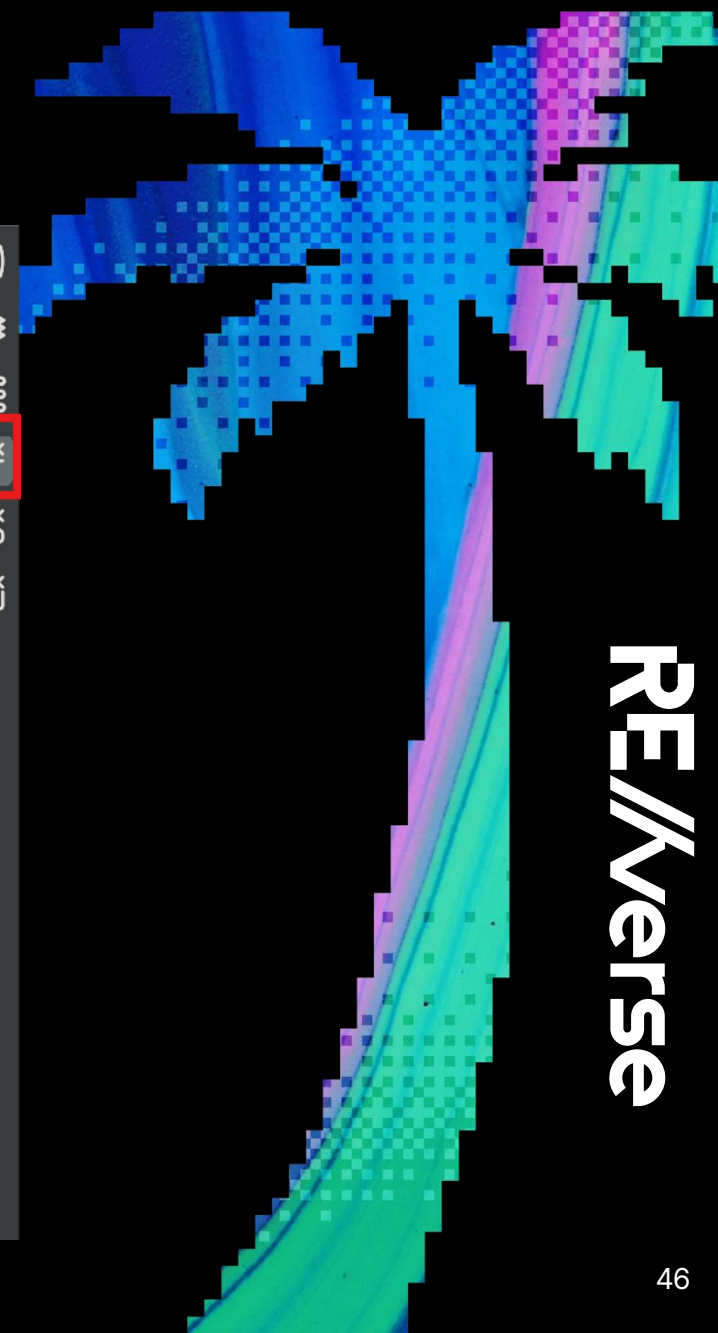
Symbols: **kernel32!WriteFile**

Return Address Range: Start address (hex, optional) to End address (hex, optional)

Query TTD Calls Clear Results

| Index | Time Start | Time End | Function           | Function Address | Return Address | Return Value | Parameters        |
|-------|------------|----------|--------------------|------------------|----------------|--------------|-------------------|
| 0     | 29f2c:1a1  | 29f2e:14 | kernel32!WriteFile | 0x74f4eed0       | 0x6817b502     | 0x1          | {389266080176...  |
| 1     | 44294:7a   | 4429e:14 | kernel32!WriteFile | 0x74f4eed0       | 0x12da4fe      | 0x1          | {3296171964120... |
| 2     | 4431f:7a   | 44328:14 | kernel32!WriteFile | 0x74f4eed0       | 0x12da4fe      | 0x1          | {3296171964120... |
| 3     | 443a8:4de  | 443b1:14 | kernel32!WriteFile | 0x74f4eed0       | 0x12da4fe      | 0x1          | {3296171964120... |
| 4     | 44421:8b2  | 44426:14 | kernel32!WriteFile | 0x74f4eed0       | 0x12da4fe      | 0x1          | {3296171964120... |
| 5     | 44496:7a   | 4449e:14 | kernel32!WriteFile | 0x74f4eed0       | 0x12da4fe      | 0x1          | {3296171964120... |
| 6     | 44572:dfe  | 44577:14 | kernel32!WriteFile | 0x74f4eed0       | 0x12da4fe      | 0x1          | {3296171964120... |
| 7     | 4469e:2610 | 446a5:14 | kernel32!WriteFile | 0x74f4eed0       | 0x12da4fe      | 0x1          | {3296171964120... |
| 8     | 447a6:7a   | 447ae:14 | kernel32!WriteFile | 0x74f4eed0       | 0x12da4fe      | 0x1          | {3296171964120... |
| 9     | 4490e:5e4  | 44915:14 | kernel32!WriteFile | 0x74f4eed0       | 0x12da4fe      | 0x1          | {3296171964120... |
| 10    | 44a8e:7a   | 44a98:14 | kernel32!WriteFile | 0x74f4eed0       | 0x12da4fe      | 0x1          | {3296171964120... |
| 11    | 44dc1:110  | 44dd2:14 | kernel32!WriteFile | 0x74f4eed0       | 0x12da4fe      | 0x1          | {3296171964120... |

Found 236 call events



RE//verse

| Index ^ | Time Start | Time End | Function           | Function Address | Return Address | Return Value | Parameters                                   |
|---------|------------|----------|--------------------|------------------|----------------|--------------|----------------------------------------------|
| 0       | 29f2c:1a1  | 29f2e:14 | kernel32!WriteFile | 0x74f4eed0       | 0x6817b502     | 0x1          | {389266080176015272, 389266166075359240, ... |
| 1       | 44294:7a   | 4429e:14 | kernel32!WriteFile | 0x74f4eed0       | 0x12da4fe      | 0x1          | {329617196412045328, 389243127870787584, ... |
| 2       | 4431f:7a   | 44328:14 | kernel32!WriteFile | 0x74f4eed0       | 0x12da4fe      | 0x1          | {329617196412045328, 389243127870787584, ... |
| 3       | 443a8:4de  | 443b1:14 | kernel32!WriteFile | 0x74f4eed0       | 0x12da4fe      | 0x1          | {329617196412045328, 389243127870787584, ... |
| 4       | 44421:8b2  | 44426:14 | kernel32!WriteFile | 0x74f4eed0       | 0x12da4fe      | 0x1          | {329617196412045328, 389243127870787584, ... |
| 5       | 44496:7a   | 4449e:14 | kernel32!WriteFile | 0x74f4eed0       | 0x12da4fe      | 0x1          | {329617196412045328, 389243127870787584, ... |
| 6       | 44572:dfe  | 44577:14 | kernel32!WriteFile | 0x74f4eed0       | 0x12da4fe      | 0x1          | {329617196412045328, 389243127870787584, ... |
| 7       | 4469e:2610 | 446a5:14 | kernel32!WriteFile | 0x74f4eed0       | 0x12da4fe      | 0x1          | {329617196412045328, 389243127870787584, ... |
| 8       | 447a6:7a   | 447ae:14 | kernel32!WriteFile | 0x74f4eed0       | 0x12da4fe      | 0x1          | {329617196412045328, 389243127870787584, ... |
| 9       | 4490e:5e4  | 44915:14 | kernel32!WriteFile | 0x74f4eed0       | 0x12da4fe      | 0x1          | {329617196412045328, 389243127870787584, ... |

Found 236 call events

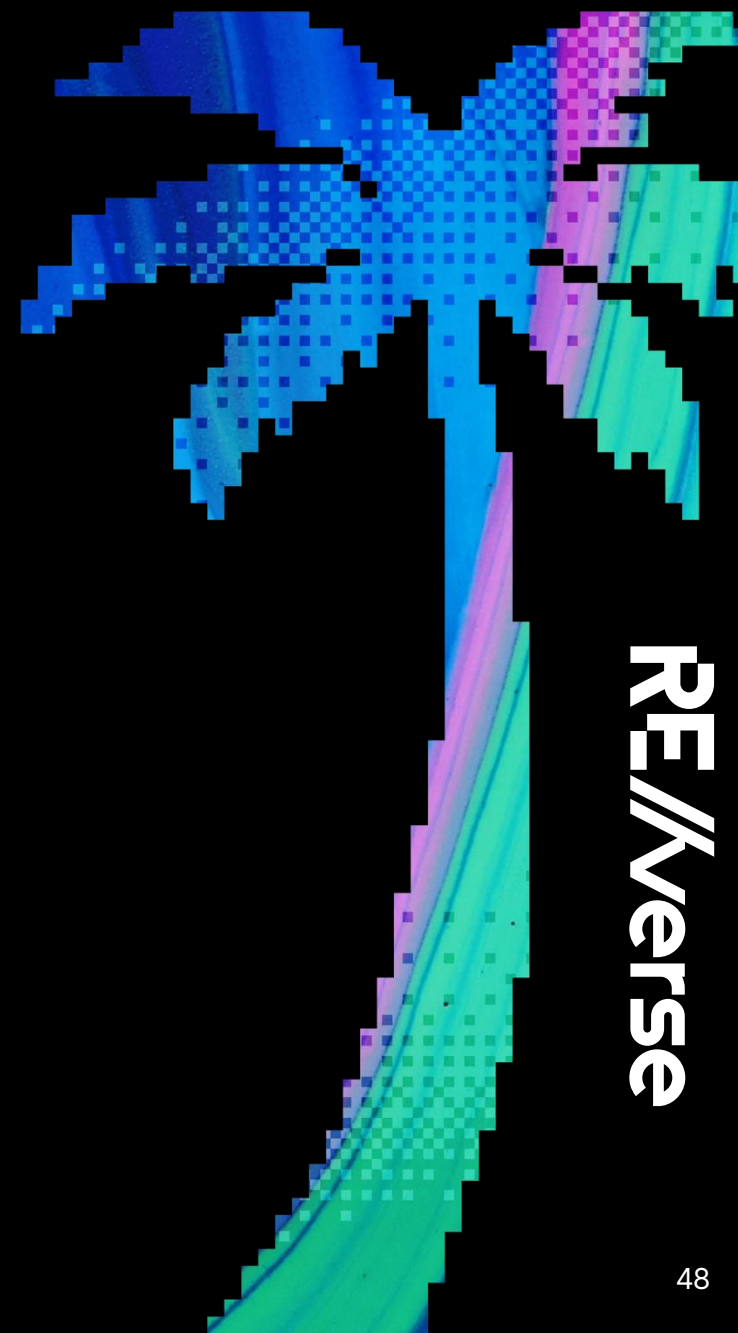
## Stack Trace

| State    | Thread               | Frame # | Module        | Function                                                                                       |
|----------|----------------------|---------|---------------|------------------------------------------------------------------------------------------------|
| Unfrozen | (*) 0x6 @ 0x74f4eed0 |         |               |                                                                                                |
|          |                      | 0       | KERNEL32      | WriteFile                                                                                      |
|          |                      | 1       | NetacLockFile | __write + 0xa1                                                                                 |
|          |                      | 2       | NetacLockFile | write_buffer_nolock<char> + 0x4c                                                               |
|          |                      | 3       | NetacLockFile | common_flush_and_write_nolock<char> + 0xf7                                                     |
|          |                      | 4       | NetacLockFile | __fwrite_nolock + 0x139                                                                        |
|          |                      | 5       | NetacLockFile | <lambda_54dcfcba6f8e0c549fa430f4d53fb7dd>::operator() + 0x2f                                   |
|          |                      | 6       | NetacLockFile | __crt_seh_guarded_call<class <lambda_cd3bfd923e8d1349099e544c97b1bd8e>,class <lambda_54dcfcba6 |
|          |                      | 7       | NetacLockFile | _fwrite + 0x63                                                                                 |
|          |                      | 8       | NetacLockFile | sub_ac3b70 + 0x46d                                                                             |
|          |                      | 9       | NetacLockFile | sub_ac4e40 + 0x1c6                                                                             |
|          |                      | 10      | NetacLockFile | sub_b2d860 + 0x24a1                                                                            |

# Find the Buffer Written To the File

```
BOOL WriteFile(  
    [in]          HANDLE      hFile,  
    [in]          LPCVOID     lpBuffer,  
    [in]          DWORD       nNumberOfBytesToWrite,  
    [out, optional] LPDWORD    lpNumberOfBytesWritten,  
    [in, out, optional] LPOVERLAPPED lpOverlapped  
);
```

```
0566de68  int32_t return_address = 0x12da4fe  
0566de6c  int32_t hFile = 0x810  
0566de70  int32_t lpBuffer = 0x4930918  
0566de74  int32_t nNumberOfBytesToWrite = 0x1000  
0566de78  int32_t lpNumberOfBytesWritten = 0x566de90  
0566de7c  int32_t lpOverlapped = 0x0
```



RE//verse

# This is Our Target -- Buffer Written to the File

```
049308e0 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
04930900 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-af d0 1a cd 00 14 00 88-20 20 20 20 20 20 20 20 20
04930920 00 00 00 00 00 00 00 00 00-01 09 00 00 0d 00 00 00-00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
04930940 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 00 00 00 00
04930960 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-61 75 74 68 6f 72 00 00-00 00 00 00 00 00 00 00
04930980 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-76 6e 7c 69 00 00 00 00-43 38 32 33 33 37 45 38
049309a0 31 41 32 37 30 34 44 33-34 44 30 39 39 34 42 33-44 35 38 39 39 46 41 34-00 00 00 00 00 00 00 00
049309c0 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 00 00 00
049309e0 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 00 00 00
04930a00 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 00 00 00
04930a20 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 00 00 00
04930a40 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 00 00 00
04930a60 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 00 00 00 00
04930a80 02 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-25 50 44 46 2d 31 2e 37-0d 25 e2 e3 cf d3 0d 0a
04930aa0 31 36 38 20 30 20 6f 62-6a 0d 3c 3c 2f 4c 69 6e-16 1b a6 21 4b d2 87 98-a2 d8 84 88 90 00 8e 3d
04930ac0 31 38 2f 4f 20 31 37 30-2f 45 20 31 32 30 36 35-30 2f 4e 20 31 31 2f 54-20 39 35 39 31 33 36 2f
04930ae0 81 1b 40 2b 2c 5b 30 e2-fe 51 72 64 62 75 44 30-6e 64 6f 62 6a 0d 20 20-20 20 20 20 20 20 20 20
04930b00 20 20 0d 0a 31 37 39 20-30 20 6f 62 6a 0d 3c 3c-83 9a 36 43 6e 35 13 b6-fe 81 e8 46 84 c1 a8 1d
04930b20 6f 6c 75 6d 6e 73 20 35-2f 50 72 65 64 69 63 74-6f 72 20 31 32 3e 3e 2f-46 69 6c 74 65 72 2f 46
04930b40 77 16 15 ba 05 ae 87 5f-1b 53 e2 84 a0 8c 40 49-37 42 37 32 33 34 32 37-46 33 36 34 41 35 33 38
04930b60 30 36 36 34 32 30 38 34-33 34 39 30 46 39 38 3e-3b a2 cf 53 5e 43 6a 60-2d 20 74 27 d5 f3 6c 98
04930b80 33 39 46 45 46 43 46 32-42 41 35 36 39 41 33 45-37 3e 5d 2f 49 6e 64 65-78 5b 31 36 38 20 32 36
04930ba0 7e f8 bf 64 22 ad f9 57-b2 cb 94 ee a7 c9 72 80-65 6e 67 74 68 20 37 35-2f 50 72 65 76 20 39 35
04930bc0 39 31 33 37 2f 52 6f 6f-74 20 31 36 39 20 30 20-79 b0 12 fd ea cc 2b a0-3d b3 1c 74 63 a9 b7 d4
04930be0 58 52 65 66 2f 57 5b 31-20 33 20 31 5d 3e 3e 73-74 72 65 61 6d 0d 0a 68-de 62 62 64 60 10 60 60
04930c00 1f 8c ce 78 26 d2 25 d1-60 0c fe 1c dd ba fe f3-72 c4 80 48 a6 d3 60 f6-5b 20 c9 a8 e8 06 56 23
```

```
.....
.....
.....author.....
.....vn|i....C82337E8
1A2704D34D0994B3D5899FA4.....
.....
.....
.....
.....%PDF-1.7.%.....
168 0 obj.<</Lin...!K.....=
18/O 170/E 120650/N 11/T 959136/
..@+, [0..QrdbuD0ndobj.
..179 0 obj.<<..6Cn5.....F....
columns 5/Predictor 12>>/Filter/F
w....._S.....@I7B723427F364A538
066420843490F98>;..S^Cj`- t'..l.
39FEFCF2BA569A3E7>]/Index[168 26
~..d"..W.....r.length 75/Prev 95
9137/Root 169 0 y.....+..=..tc...
XRef/W[1 3 1]>>stream..h.bbd`.``
...x&.%.`.....r..H..`. [ ....V#
```

# Matches What We Saw From the File

→ NetacLockFil....2.pdf.netac × +

Raw ▾ Hex ▾

|          |                                                                                                    |                                  |
|----------|----------------------------------------------------------------------------------------------------|----------------------------------|
| 00000000 | 20 20 20 20 20 20 20 20 20-00 00 00 00 00 00 00 00-01 09 00 00 0d 00 00 00-00 00 00 00 00 00 00 00 | .....                            |
| 00000020 | 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 | .....                            |
| 00000040 | 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-61 75 74 68 6f 72 00 00 | .....author..                    |
| 00000060 | 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-bf cd f0 68 00 00 00 00 | .....h...                        |
| 00000080 | 43 38 32 33 33 37 45 38-31 41 32 37 30 34 44 33-34 44 30 39 39 34 42 33-44 35 38 39 39 46 41 34    | C82337E81A2704D34D0994B3D5899FA4 |
| 000000a0 | 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 | .....                            |
| 000000c0 | 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 | .....                            |
| 000000e0 | 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 | .....                            |
| 00000100 | 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 | .....                            |
| 00000120 | 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 | .....                            |
| 00000140 | 00 00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00 | .....                            |
| 00000160 | 00 00 00 00 00 00 00 00 00-02 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00-25 50 44 46 2d 31 2e 37 | .....%PDF-1.7                    |
| 00000180 | 0d 25 e2 e3 cf d3 0d 0a-31 36 38 20 30 20 6f 62-6a 0d 3c 3c 2f 4c 69 6e-16 1b a6 21 4b d2 87 98    | %.168 0 obj.<</Lin...!K...       |
| 000001a0 | a2 d8 84 88 90 00 8e 3d-31 38 2f 4f 20 31 37 30-2f 45 20 31 32 30 36 35-30 2f 4e 20 31 31 2f 54    | .....=18/0 170/E 120650/N 11/T   |
| 000001c0 | 20 39 35 39 31 33 36 2f-81 1b 40 2b 2c 5b 30 e2-fe 51 72 64 62 75 44 30-6e 64 6f 62 6a 0d 20 20    | 959136/..@+, [0..QrdbuD0ndobj.   |
| 000001e0 | 20 20 20 20 20 20 20 20 20-20 20 0d 0a 31 37 39 20-30 20 6f 62 6a 0d 3c 3c-83 9a 36 43 6e 35 13 b6 | ..179 0 obj.<<..6Cn5..           |
| 00000200 | fe 81 e8 46 84 c1 a8 1d-6f 6c 75 6d 6e 73 20 35-2f 50 72 65 64 69 63 74-6f 72 20 31 32 3e 3e 2f    | ...F....olumns 5/Predictor 12>>/ |
| 00000220 | 46 69 6c 74 65 72 2f 46-77 16 15 ba 05 ae 87 5f-1b 53 e2 84 a0 8c 40 49-37 42 37 32 33 34 32 37    | Filter/Fw.....S....@I7B723427    |
| 00000240 | 46 33 36 34 41 35 33 38-30 36 36 34 32 30 38 34-33 34 39 30 46 39 38 3e-3b a2 cf 53 5e 43 6a 60    | F364A538066420843490F98>;..S^Cj` |
| 00000260 | 2d 20 74 27 d5 f3 6c 98-33 39 46 45 46 43 46 32-42 41 35 36 39 41 33 45-37 3e 5d 2f 49 6e 64 65    | - t'..l.39FEFCF2BA569A3E7>]/Inde |
| 00000280 | 78 5b 31 36 38 20 32 36-7e f8 bf 64 22 ad f9 57-b2 cb 94 ee a7 c9 72 80-65 6e 67 74 68 20 37 35    | x[168 26~..d"..W.....r.length 75 |
| 000002a0 | 2f 50 72 65 76 20 39 35-39 31 33 37 2f 52 6f 6f-74 20 31 36 39 20 30 20-79 b0 12 fd ea cc 2b a0    | /Prev 959137/Root 169 0 y.....+  |
| 000002c0 | 3d b3 1c 74 63 a9 b7 d4-58 52 65 66 2f 57 5b 31-20 33 20 31 5d 3e 3e 73-74 72 65 61 6d 0d 0a 68    | =..tc...XRef/W[1 3 1]>>stream..h |
| 000002e0 | de 62 62 64 60 10 60 60-1f 8c ce 78 26 d2 25 d1-60 0c fe 1c dd ba fe f3-72 c4 80 48 a6 d3 60 f6    | .bbd`.``...x&.%.`.....r..H..`    |

Part of the file remains readable after encryption

RE/verse

```
0    KERNEL32      WriteFile
1    NetacLockFile __write + 0xa1
2    NetacLockFile write_buffer_nolock<char> +
3    NetacLockFile common_flush_and_write_nolo
4    NetacLockFile __fwrite_nolock + 0x139
5    NetacLockFile <lambda_54dcfcba6f8e0c549fa
6    NetacLockFile __crt_seh_guarded_call<clas
7    NetacLockFile _fwrite + 0x63
8    NetacLockFile sub_ac3b70 + 0x46d
9    NetacLockFile sub_ac4e40 + 0x1c6
```

We are deeply  
buried in the call  
stack

We can step out 8  
times until we return  
from the fwrite call

```
? 00ac3fd0 push    dword [ebx+0x10]
00ac3fd3 push    0x10
00ac3fd5 push    0x1
00ac3fd7 push    eax
00ac3fd8 call    _fwrite
→ 00ac3fdd add     esp, 0x10
00ac3fe0 cmp     eax, 0x10
00ac3fe3 jne     0xac4324
```

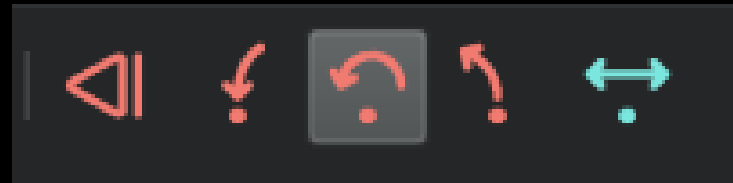
```

? 00ac3fd0 push dword [ebx+0x10]
00ac3fd3 push 0x10
00ac3fd5 push 0x1
00ac3fd7 push eax
00ac3fd8 call _fwrite
→ 00ac3fdd add esp, 0x10
00ac3fe0 cmp eax, 0x10
00ac3fe3 jne 0xac4324

```

In traditional debugging, this means we have missed the opportunity to examine the buffer

But with TTD, we simply need to step over backwards once to get to the state before the call



```

? 00ac3fd0 push dword [ebx+0x10]
00ac3fd3 push 0x10
00ac3fd5 push 0x1
00ac3fd7 push eax
→ 00ac3fd8 call _fwrite
00ac3fdd add esp, 0x10
00ac3fe0 cmp eax, 0x10
00ac3fe3 jne 0xac4324

```

```
size_t fwrite(
    const void* buffer,
    size_t size,
    size_t count,
    FILE* stream
);
```

```

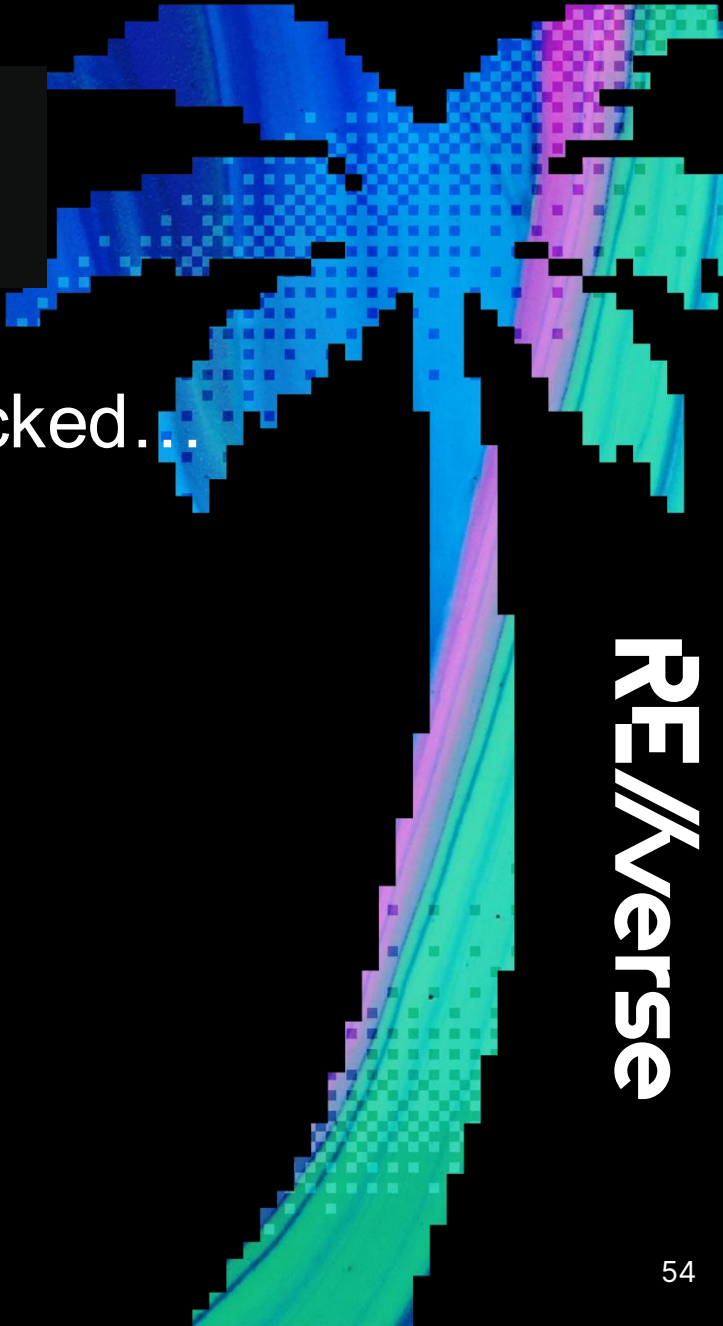
? 00ac3fd0 push    dword [ebx+0x10]
  00ac3fd3 push    0x10
  00ac3fd5 push    0x1
  00ac3fd7 push    eax
→ 00ac3fd8 call    _fwrite
  00ac3fdd add     esp, 0x10
  00ac3fe0 cmp     eax, 0x10
  00ac3fe3 jne     0xac4324

```

eax 0x566f214

|          |             |                |                      |                  |
|----------|-------------|----------------|----------------------|------------------|
| 0566f210 | 70 1b 21 77 | 16 75 11 51-3a | 2b 92 3e 23 79 f6 01 | p.!w.u.Q:+.>#y.. |
| 0566f220 | 9d c4 ba 52 | fb a4 5a f3-74 | f6 66 05 06 50 ac 00 | ...R..Z.t.f..P.. |
| 0566f230 | 1c 00 00 00 | 22 00 00 00-c0 | bc b2 00 90 70 b1 08 | ....".....p..    |
| 0566f240 | 78 df b0 08 | 31 71 b1 08-90 | 70 b1 08 08 3f 27 76 | x...1q...p...?'v |

RE//verse



|          |             |                                     |                  |
|----------|-------------|-------------------------------------|------------------|
| 0566f210 | 70 1b 21 77 | 16 75 11 51-3a 2b 92 3e 23 79 f6 01 | p.!w.u.Q:+.>#y.. |
| 0566f220 | 9d c4 ba 52 | fb a4 5a f3-74 f6 66 05 06 50 ac 00 | ...R..Z.t.f..P.. |
| 0566f230 | 1c 00 00 00 | 22 00 00 00-c0 bc b2 00 90 70 b1 08 | ....".....p..    |
| 0566f240 | 78 df b0 08 | 31 71 b1 08-90 70 b1 08 08 3f 27 76 | x...1q...p...?'v |

This buffer looks different from the one we have tracked...

```
0566f210 70 1b 21 77 16 75 11 51-3a 2b 92 3e 23 79 f6 01 p.!w.u.Q:+.>#y..  
0566f220 9d c4 ba 52 fb a4 5a f3-74 f6 66 05 06 50 ac 00 ...R..Z.t.f..P..  
0566f230 1c 00 00 00 22 00 00 00-c0 bc b2 00 90 70 b1 08 ....".....p..  
0566f240 78 df b0 08 31 71 b1 08-90 70 b1 08 08 3f 27 76 x...1q...p...?'v
```

This buffer looks different from the one we have tracked...

Because `fwrite` caches the smaller 0x10-byte buffers – we see the last 0x10 bytes among the 0x1000 bytes that triggered an actual `WriteFile` call

```
0566f210 70 1b 21 77 16 75 11 51-3a 2b 92 3e 23 79 f6 01 p.!w.u.Q:+.>#y..  
0566f220 9d c4 ba 52 fb a4 5a f3-74 f6 66 05 06 50 ac 00 ...R..Z.t.f..P..  
0566f230 1c 00 00 00 22 00 00 00-c0 bc b2 00 90 70 b1 08 ....".....p..  
0566f240 78 df b0 08 31 71 b1 08-90 70 b1 08 08 3f 27 76 x...1q...p...?'v
```

This buffer looks different from the one we have tracked...

Because fwrite caches the smaller 0x10-byte buffers – we see the last 0x10 bytes among the 0x1000 bytes that triggered an actual WriteFile call

Spoiler: this buffer is NOT encrypted – backtracking it will go to fread...

# Let us check all fwrite calls from here

The screenshot displays a debugger window with assembly code on the left and a context menu open over it. The assembly code includes instructions like `push dword [ebx+0x10]`, `call _fwrite`, and `add esp, 0x1`. The context menu is divided into two main sections. The top section contains options like 'Enter Comment...', 'Change Type...', 'Undefined Current Function', 'Add Cross Reference to...', 'Tags and Bookmarks', 'Debugger', 'Copy', 'Copy Address', 'Copy As', 'Paste', 'Patch', 'Edit Function Properties...', 'Make Function at This Address', 'Override Call Type...', 'Rename Current Function...', 'Set Stack Adjustment...', 'Reanalyze Current Function', 'Disable Current Function Analysis', 'Function Settings', 'Halt Disassembly', 'Highlight Block', 'Highlight Instruction', 'HashDB', and 'Plugins'. The bottom section contains 'Add Hardware Breakpoint...', 'Toggle Breakpoint', 'Restart', 'Resume', 'Run Back To Here', 'Run To Here', 'Step Into', 'Step Over', 'Step Return', 'Create Stack View', 'Override IP', 'Rebase to Remote Base...', 'Navigate to TTD Timestamp...', 'TTD Calls', 'TTD Memory Access', 'Execute', 'Read', 'Read/Write', 'Read/Write/Execute', and 'Write'. The 'Debugger' and 'TTD Memory Access' options are highlighted in blue.

```
? 00ac3fd0 push dword [ebx+0x10]
00ac3fd3 push 0x10
00ac3fd5 push 0x1
00ac3fd7 push eax
→ 00ac3fd8 call _fwrite
00ac3fdd add esp, 0x1
00ac3fe0 cmp eax, 0x1
00ac3fe3 jne 0xac4324
```

00ac3fef  
00ac3ff1  
00ac3ff3  
00ac3ff4  
00ac3ffa  
00ac3ffb  
00ac4000  
00ac4007

```
00ac4017 push dword [ebp-0x1]
00ac401d push dword [ebp-0x1]
```

00ac4023  
00ac4024  
00ac4025 call \_\_alldiv

- Enter Comment... ;
- Change Type... Y
- Undefined Current Function U
- Add Cross Reference to...
- Tags and Bookmarks
- Debugger
- Copy Ctrl+C
- Copy Address Ctrl+Shift+A
- Copy As
- Paste Ctrl+V
- Patch
- Edit Function Properties... Shift+E
- Make Function at This Address
- Override Call Type...
- Rename Current Function...
- Set Stack Adjustment...
- Reanalyze Current Function
- Disable Current Function Analysis
- Function Settings
- Halt Disassembly
- Highlight Block
- Highlight Instruction
- HashDB
- Plugins
- Add Hardware Breakpoint... F3
- Toggle Breakpoint F2
- Restart
- Resume F9
- Run Back To Here Shift+F4
- Run To Here F4
- Step Into F7
- Step Over F8
- Step Return Ctrl+F9
- Create Stack View
- Override IP
- Rebase to Remote Base...
- Navigate to TTD Timestamp... Shift+G
- TTD Calls
- TTD Memory Access
- Execute
- Read
- Read/Write
- Read/Write/Execute
- Write

RE//verse

| Index | Position   | Access Type | Address  | Size | Value      | Thread ID |
|-------|------------|-------------|----------|------|------------|-----------|
| 0xe4  | 44292:36e  | E           | 0xac3fd8 | 5    | 0x807fd3e8 | 12376     |
| 0xe5  | 44292:742  | E           | 0xac3fd8 | 5    | 0x807fd3e8 | 12376     |
| 0xe6  | 44292:1062 | E           | 0xac3fd8 | 5    | 0x807fd3e8 | 12376     |
| 0xe7  | 44292:1436 | E           | 0xac3fd8 | 5    | 0x807fd3e8 | 12376     |
| 0xe8  | 44292:180a | E           | 0xac3fd8 | 5    | 0x807fd3e8 | 12376     |
| 0xe9  | 4429e:8ff  | E           | 0xac3fd8 | 5    | 0x807fd3e8 | 12376     |
| 0xea  | 442a0:36e  | E           | 0xac3fd8 | 5    | 0x807fd3e8 | 12376     |
| 0xeb  | 442a0:742  | E           | 0xac3fd8 | 5    | 0x807fd3e8 | 12376     |

Navigate to TTD Timestamp

Navigate to Position

Timestamp (Sequence:Step): 44292:180A

Enter timestamp in hex format: SEQUENCE:STEP

NavigateCancel

0xe8 calls of fwrite == 1 call of WriteFile, caching is in effect

We need to find a different fwrite call that writes an encrypted buffer (it is 1 in 3)

| Index | Position  |
|-------|-----------|
| 0x0   | 44228:24c |

```
eax 0x566f214 "%PDF-1.7\r%"
```

| Index | Position  |
|-------|-----------|
| 0x1   | 4422a:164 |

```
eax 0x566f214 "168 0 obj\r<</Lin"
```

| Index | Position  |
|-------|-----------|
| 0x2   | 4422a:a84 |

```
0566f1b0 26 0a 22 77 20 00 00 00-16 1b a6 21 4b d2 87 98 &."w ...!K...
0566f1c0 a2 d8 84 88 90 00 8e 3d-08 09 0a 0b 0d 0e 0f 10 .....=.....
```

The last one is what we want

TTD Calls

TTD Memory Access

Execute

Read

Read/Write

Read/Write/Execute

Write

TTD Memory

Query 1

Query 2

Query 3

Query 4

▼ Query Parameters

Address Range:

Start: 0x566f1b8

End: 0x566f1b9

Time Range (Optional):

Start Time: e.g. 0:0

End Time: e.g. 23f:a7

Access Types:

☐ Read

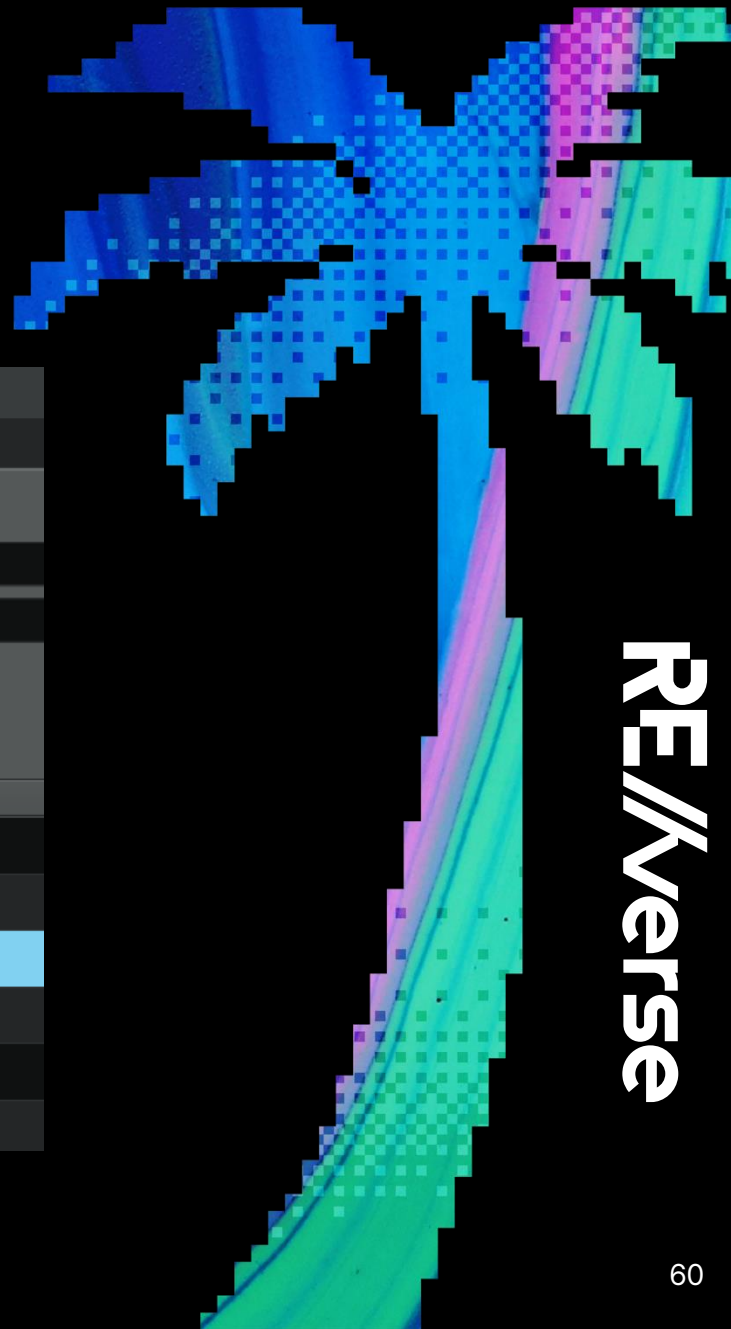
☒ Write

☐ Execute

Query Memory Events

Clear Results

| Index | Position   | Access Type | Address   | Size | Value | Thread ID | IP       |
|-------|------------|-------------|-----------|------|-------|-----------|----------|
| 0xc2  | 4422c:8a1  | W           | 0x566f1b8 | 16   | 0x0   | 12376     | 0xac5995 |
| 0xc3  | 4422c:1969 | W           | 0x566f1b8 | 16   | 0x0   | 12376     | 0xac5995 |
| 0xc4  | 4422e:f4f  | W           | 0x566f1b8 | 16   | 0x0   | 12376     | 0xac5995 |
| 0xc5  | 44232:971  | W           | 0x566f1b8 | 16   | 0x0   | 12376     | 0xac5995 |
| 0xc6  | 44234:c75  | W           | 0x566f1b8 | 16   | 0x0   | 12376     | 0xac5995 |
| 0xc7  | 44236:6a1  | W           | 0x566f1b8 | 16   | 0x0   | 12376     | 0xac5995 |



# This Looks Like a Crypto Function

```
00ac5950 crypto:
00ac5950 push    ebp {__saved_ebp}
00ac5951 mov     ebp, esp {__saved_ebp}
00ac5953 and     esp, 0xffffffff
00ac5956 sub     esp, 0x7c
00ac5959 mov     eax, dword [__security_cookie]
00ac595e xor     eax, esp {var_8c}
00ac5960 mov     dword [esp+0x78 {var_14}], eax
00ac5964 mov     eax, dword [ebp+0x10 {arg3}]
00ac5967 lea     edx, [esp+0x1c {var_70}]
00ac596b mov     ecx, dword [ebp+0x8 {arg1}]
00ac596e push    esi {__saved_e si}
00ac596f mov     esi, dword [ebp+0xc {arg2}]
00ac5972 movups xmm0, xmmword [eax]
00ac5975 lea     eax, [esp+0x10 {var_80}]
00ac5979 push    eax {var_80} {var_94}
00ac597a movups xmmword [esp+0x14 {var_80}], xmm0
00ac597f call   sub_ac5590
00ac5984 movups xmm0, xmmword [esp+0x24 {var_70}]
00ac5989 add     esp, 0x4
00ac598c lea     eax, [esp+0x38 {var_58}]
00ac5990 mov     ecx, 0x40
→ 00ac5995 movups xmmword [esi], xmm0
00ac5998 nop     dword [eax+eax]
```

```
00ac55d3 shr     eax, 0x10
00ac55d6 movzx   eax, al
00ac55d9 mov     ecx, dword [ecx*4+0x16c7230]
00ac55e0 xor     ecx, dword [eax*4+0x16c6060]
00ac55e7 mov     eax, ebx
00ac55e9 shr     eax, 0x18
00ac55ec xor     ecx, dword [eax*4+0x16c7e50]
00ac55f3 mov     eax, dword [ebp+0x8 {arg4}]
00ac55f6 movzx   eax, al
00ac55f9 xor     ecx, dword [eax*4+0x16c5c60]
00ac5600 mov     eax, ebx
00ac5602 shr     eax, 0x10
00ac5605 mov     dword [edi], ecx
00ac5607 mov     dword [ebp-0x1c {var_20}], ecx
00ac560a movzx   ecx, al
00ac560d mov     eax, dword [ebp-0x10 {var_14}]
00ac5610 shr     eax, 0x8
00ac5613 movzx   eax, al
00ac5616 mov     esi, dword [ecx*4+0x16c6060]
00ac561d xor     esi, dword [eax*4+0x16c7230]
00ac5624 mov     eax, dword [ebp+0x8 {arg4}]
```

# Crypto Constants

|          |               |    |    |    |    |    |    |       |    |    |    |    |    |    |       |    |    |    |    |    |    |       |    |    |    |    |    |    |    |  |
|----------|---------------|----|----|----|----|----|----|-------|----|----|----|----|----|----|-------|----|----|----|----|----|----|-------|----|----|----|----|----|----|----|--|
| 016c7230 | data_16c7230: |    |    |    |    |    |    |       |    |    |    |    |    |    |       |    |    |    |    |    |    |       |    |    |    |    |    |    |    |  |
| 016c7230 |               |    |    |    |    |    |    |       |    |    |    |    |    |    |       |    |    |    |    |    |    |       |    |    |    |    |    |    |    |  |
| 016c7240 | 0d            | ff | f2 | f2 | bd | d6 | 6b | 6b-b1 | de | 6f | 6f | 54 | 91 | c5 | c5-50 | 60 | 30 | 30 | 03 | 02 | 01 | 01-a9 | ce | 67 | 67 | 7d | 56 | 2b | 2b |  |
| 016c7260 | 19            | e7 | fe | fe | 62 | b5 | d7 | d7-e6 | 4d | ab | ab | 9a | ec | 76 | 76-45 | 8f | ca | ca | 9d | 1f | 82 | 82-40 | 89 | c9 | c9 | 87 | fa | 7d | 7d |  |
| 016c7280 | 15            | ef | fa | fa | eb | b2 | 59 | 59-c9 | 8e | 47 | 47 | 0b | fb | f0 | f0-ec | 41 | ad | ad | 67 | b3 | d4 | d4-fd | 5f | a2 | a2 | ea | 45 | af | af |  |
| 016c72a0 | bf            | 23 | 9c | 9c | f7 | 53 | a4 | a4-96 | e4 | 72 | 72 | 5b | 9b | c0 | c0-c2 | 75 | b7 | b7 | 1c | e1 | fd | fd-ae | 3d | 93 | 93 | 6a | 4c | 26 | 26 |  |
| 016c72c0 | 5a            | 6c | 36 | 36 | 41 | 7e | 3f | 3f-02 | f5 | f7 | f7 | 4f | 83 | cc | cc-5c | 68 | 34 | 34 | f4 | 51 | a5 | a5-34 | d1 | e5 | e5 | 08 | f9 | f1 | f1 |  |
| 016c72e0 | 93            | e2 | 71 | 71 | 73 | ab | d8 | d8-53 | 62 | 31 | 31 | 3f | 2a | 15 | 15-0c | 08 | 04 | 04 | 52 | 95 | c7 | c7-65 | 46 | 23 | 23 | 5e | 9d | c3 | c3 |  |
| 016c7300 | 28            | 30 | 18 | 18 | a1 | 37 | 96 | 96-0f | 0a | 05 | 05 | b5 | 2f | 9a | 9a-09 | 0e | 07 | 07 | 36 | 24 | 12 | 12-9b | 1b | 80 | 80 | 3d | df | e2 | e2 |  |
| 016c7320 | 26            | cd | eb | eb | 69 | 4e | 27 | 27-cd | 7f | b2 | b2 | 9f | ea | 75 | 75-1b | 12 | 09 | 09 | 9e | 1d | 83 | 83-74 | 58 | 2c | 2c | 2e | 34 | 1a | 1a |  |
| 016c7340 | 2d            | 36 | 1b | 1b | b2 | dc | 6e | 6e-ee | b4 | 5a | 5a | fb | 5b | a0 | a0-f6 | a4 | 52 | 52 | 4d | 76 | 3b | 3b-61 | b7 | d6 | d6 | ce | 7d | b3 | b3 |  |
| 016c7360 | 7b            | 52 | 29 | 29 | 3e | dd | e3 | e3-71 | 5e | 2f | 2f | 97 | 13 | 84 | 84-f5 | a6 | 53 | 53 | 68 | b9 | d1 | d1-00 | 00 | 00 | 00 | 2c | c1 | ed | ed |  |
| 016c7380 | 60            | 40 | 20 | 20 | 1f | e3 | fc | fc-c8 | 79 | b1 | b1 | ed | b6 | 5b | 5b-be | d4 | 6a | 6a | 46 | 8d | cb | cb-d9 | 67 | be | be | 4b | 72 | 39 | 39 |  |
| 016c73a0 | de            | 94 | 4a | 4a | d4 | 98 | 4c | 4c-e8 | b0 | 58 | 58 | 4a | 85 | cf | cf-6b | bb | d0 | d0 | 2a | c5 | ef | ef-e5 | 4f | aa | aa | 16 | ed | fb | fb |  |
| 016c73c0 | c5            | 86 | 43 | 43 | d7 | 9a | 4d | 4d-55 | 66 | 33 | 33 | 94 | 11 | 85 | 85-cf | 8a | 45 | 45 | 10 | e9 | f9 | f9-06 | 04 | 02 | 02 | 81 | fe | 7f | 7f |  |



RE/verse

0x6363c6a5

AI Mode **All** Shopping Images Videos Short videos News More ▾ Tools ▾



Google Open Source

<https://chromium.googlesource.com> › chromium › aes.c ⋮

[crypto/cipher/aes.c](#) - chromium/deps/libsrtp

\* An implemnetation of the AES block cipher. \*. \* David A. McGrew. \* Cisco Systems, Inc.

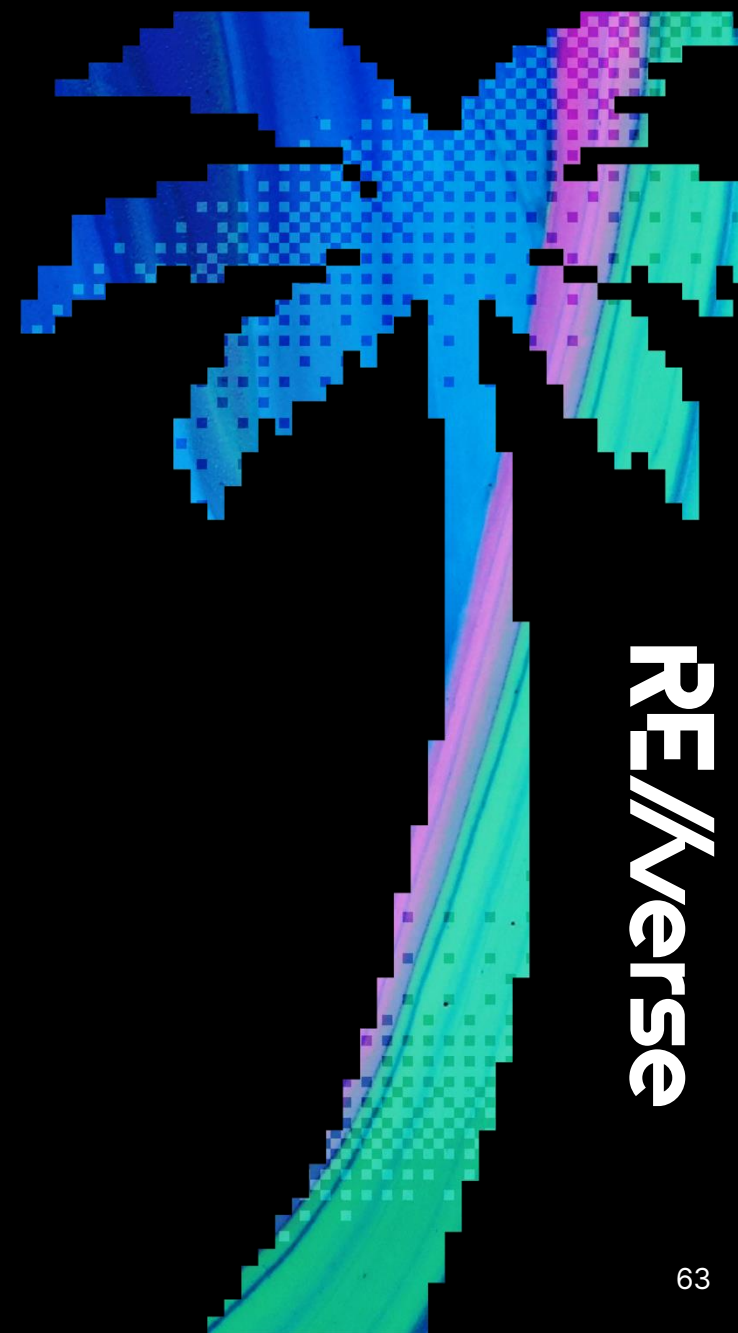


GitHub

<https://github.com> › cisco › libsrtp › blob › cipher › aes.c ⋮

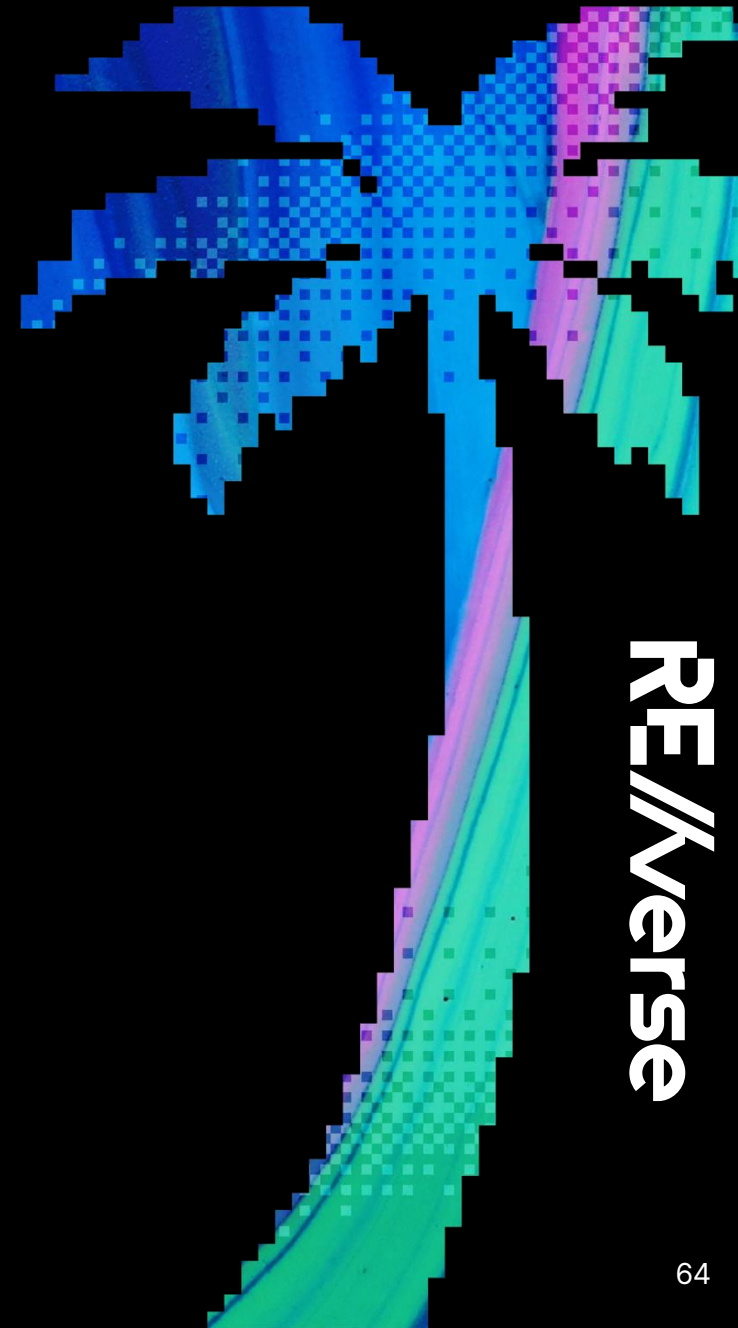
[libsrtp/crypto/cipher/aes.c](#) at main

... 0x6363c6a5, 0x7c7cf884, 0x7777ee99, 0x7b7bf68d, 0xf2f2ff0d, 0x6b6bd6bd ... 0x6363c6a5,  
0x7c7cf884, 0x7777ee99, 0x7b7bf68d,. 0xf2f2ff0d, 0x6b6bd6bd ...



RE//verse

We now know it is AES – not surprising  
What is the key used?



RE//verse

# Asking the Sidekick plugin to help us recover the AES key

Chat

AES Key Recovery from Expanded Key Schedule

This function is an AES implementation. Analyze how parameters are passed in, and how should I recover the key being used

Read visible content

Query: function("crypto\_block")

Query: callers("crypto\_block")

Query: function("crypto")

This is a standard AES encryption block function. Here's the parameter analysis:

**Parameters:**

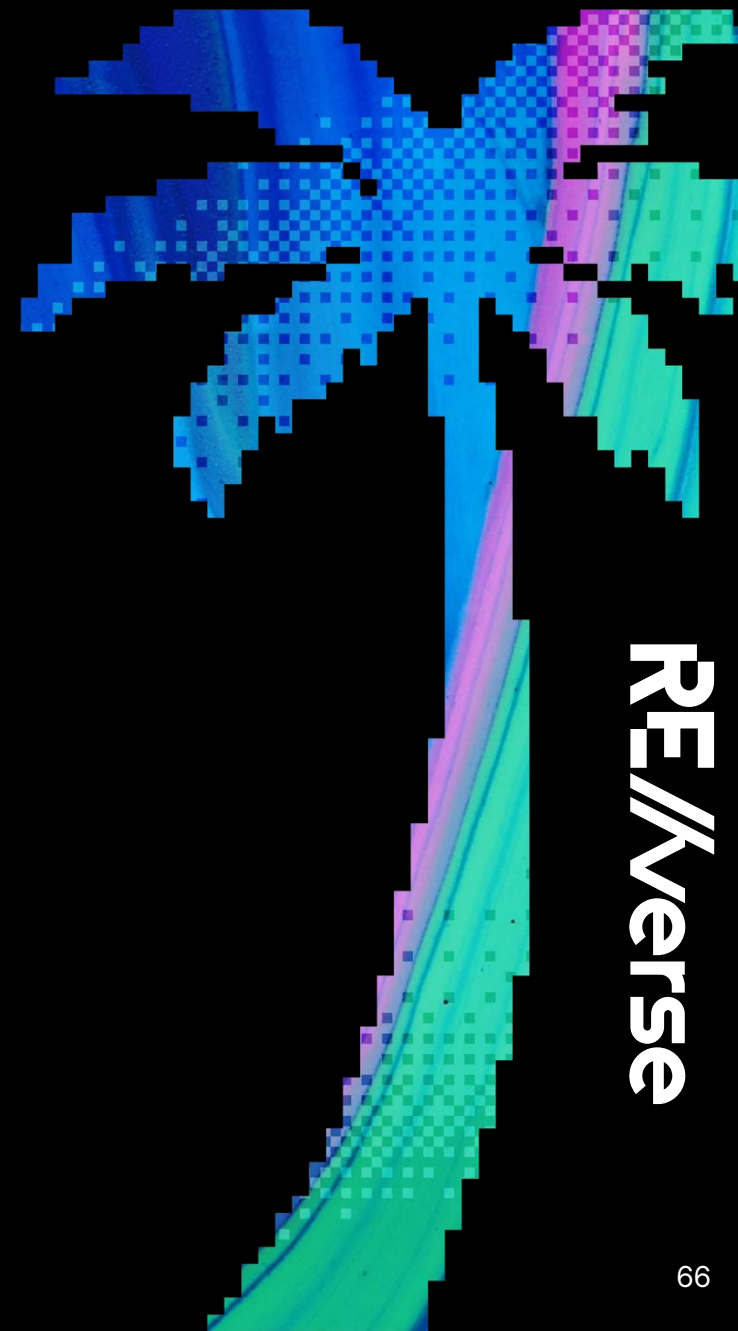
- `arg1 (ecx)`: Pointer to expanded key schedule (round keys)
- `arg2 (edx)`: Pointer to output buffer (16 bytes)
- `arg3 (stack)`: Pointer to input plaintext block (16 bytes)
- `arg4 (stack)`: Number of rounds (10/12/14 for AES-128/192/256)

**To recover the key:**

1. **Set a breakpoint** at `0xac5590` (function entry)
2. **Examine `arg1 (ecx register)`** - this points to the expanded key schedule
3. **The original AES key is in the first 16/24/32 bytes** of the key schedule at `[ecx]`:
  - AES-128: bytes 0-15
  - AES-192: bytes 0-23
  - AES-256: bytes 0-31
4. **Check `arg4`** to determine key size: value of 10 = AES-128, 12 = AES-192, 14 = AES-256

Alternatively, trace back to where `crypto_block` is called from `crypto (0xac5950)` - the key schedule pointer originates from `arg1` passed to that function. Set a breakpoint there and dump the memory at `arg1`.

|          |                                                 |                    |
|----------|-------------------------------------------------|--------------------|
| 08c692a0 | 10 5f ac 00 5c 00 62 00-0e 00 00 00 00 00 00    | ._..\.b.....       |
| 08c692b0 | 08 09 0a 0b 0d 0e 0f 10-12 13 14 15 17 18 19 1a | .....              |
| 08c692c0 | 1c 1d 1e 1f 21 22 23 24-26 27 28 29 2b 2c ff ff | ....!"#\$%&'()+,.. |
| 08c692d0 | 78 1f 1c fa 75 11 13 ea-67 02 07 ff 70 1a 1e e5 | x...u...g...p...   |
| 08c692e0 | 4d bf 6c c6 6c 9d 4f e2-4a ba 67 cb 61 96 98 34 | M.l.l.O.J.g.a..4   |



RE//verse



|          |                                                 |                    |
|----------|-------------------------------------------------|--------------------|
| 08c692a0 | 10 5f ac 00 5c 00 62 00-0e 00 00 00 00 00 00 00 | ._._\..b.....      |
| 08c692b0 | 08 09 0a 0b 0d 0e 0f 10-12 13 14 15 17 18 19 1a | .....              |
| 08c692c0 | 1c 1d 1e 1f 21 22 23 24-26 27 28 29 2b 2c ff ff | ....!"#\$%&'()+,.. |
| 08c692d0 | 78 1f 1c fa 75 11 13 ea-67 02 07 ff 70 1a 1e e5 | x...u...g...p...   |
| 08c692e0 | 4d bf 6c c6 6c 9d 4f e2-4a ba 67 cb 61 96 98 34 | M.l.l.O.J.g.a..4   |

```
0x16c56e0 .rdata {0x1314000-0x1708336} Read-only data
```

```
016c56e0 int128_t data_16c56e0 =
016c56e0 08 09 0a 0b 0d 0e 0f 10-12 13 14 15 17 18 19 1a .....
016c56f0 int128_t data_16c56f0 =
016c56f0 1c 1d 1e 1f 21 22 23 24-26 27 28 29 2b 2c ff ff .....! "$&' ()+,...
```

Found 0 memory access events

These are constants from the PE file (.rdata section)!

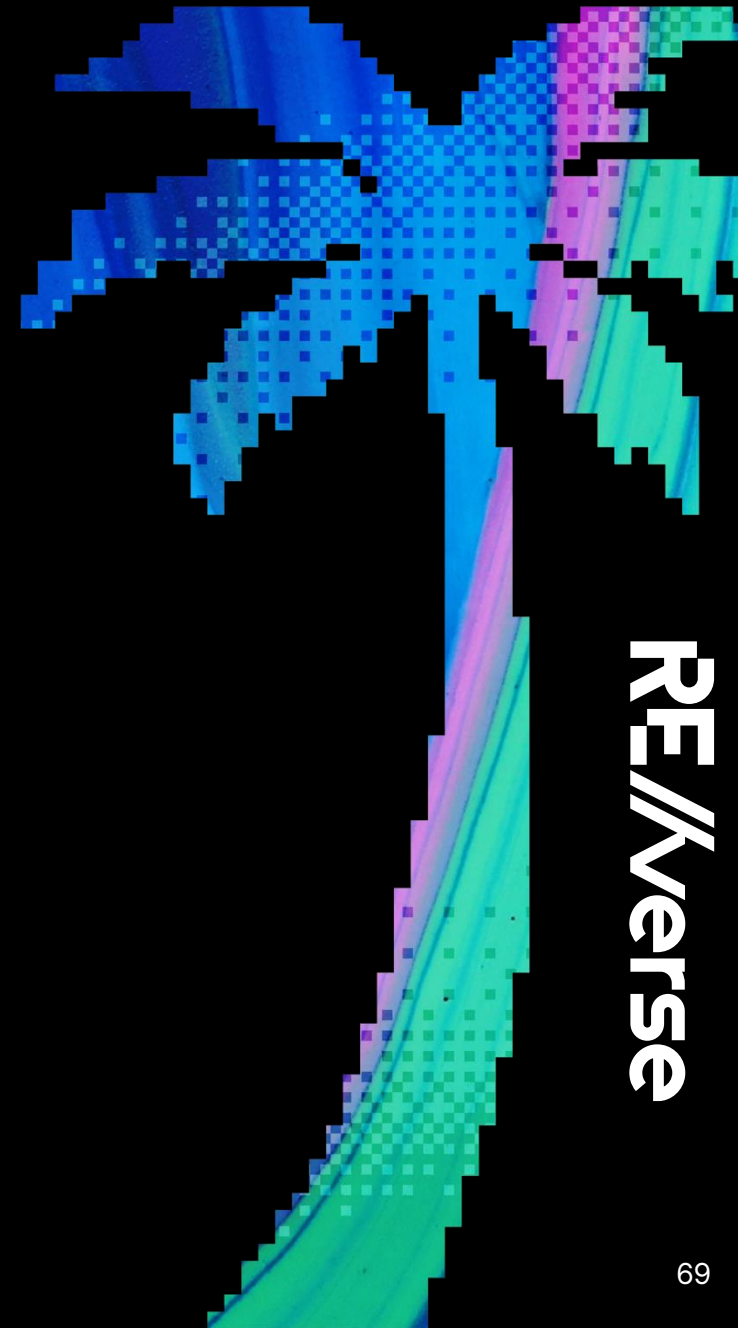
**RE/verse**

# Summary

Only 1/3 of the file is encrypted

The encryption uses a fixed AES key embedded in the binary

→ Totally broken



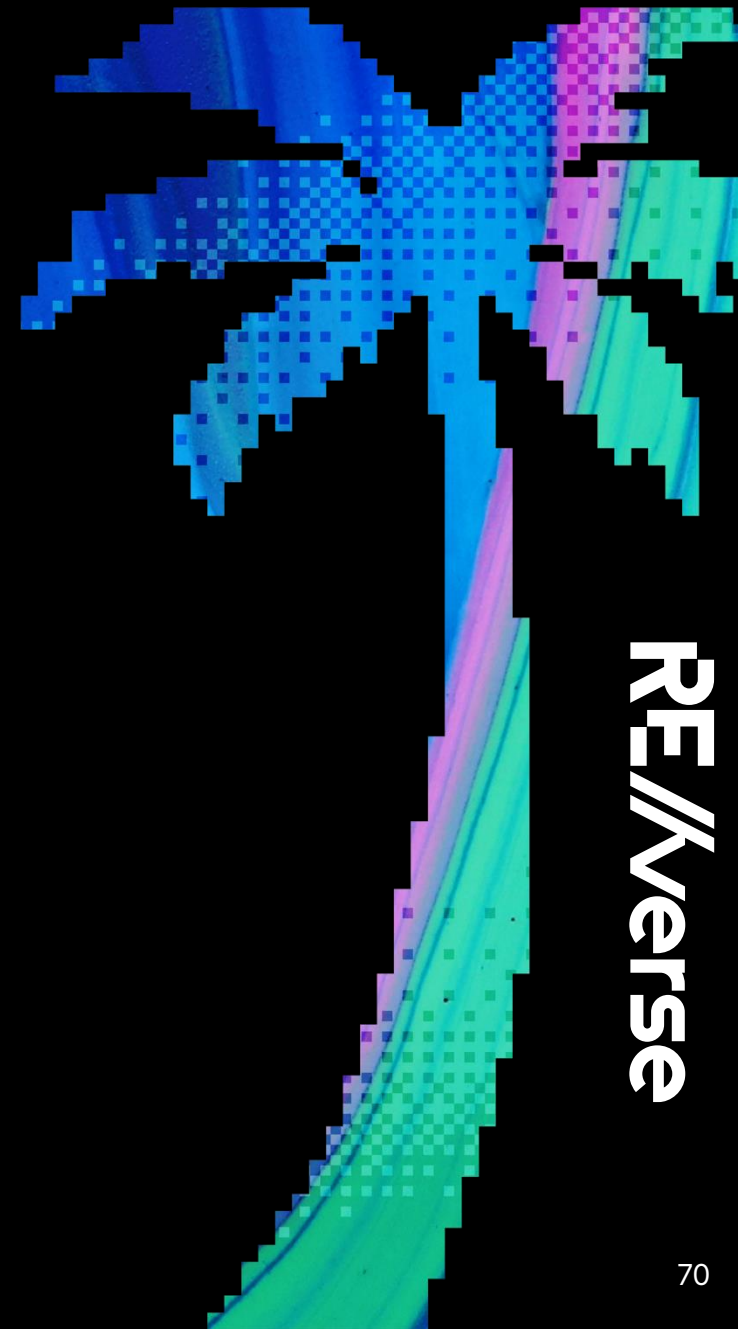
# Disclosure Timeline

Reported to Netac on Oct/19/2025

No response from the vendor

No patch available

Public disclosure in this talk (90+ days)



RE//verse



# RE

## Vulnerability #3

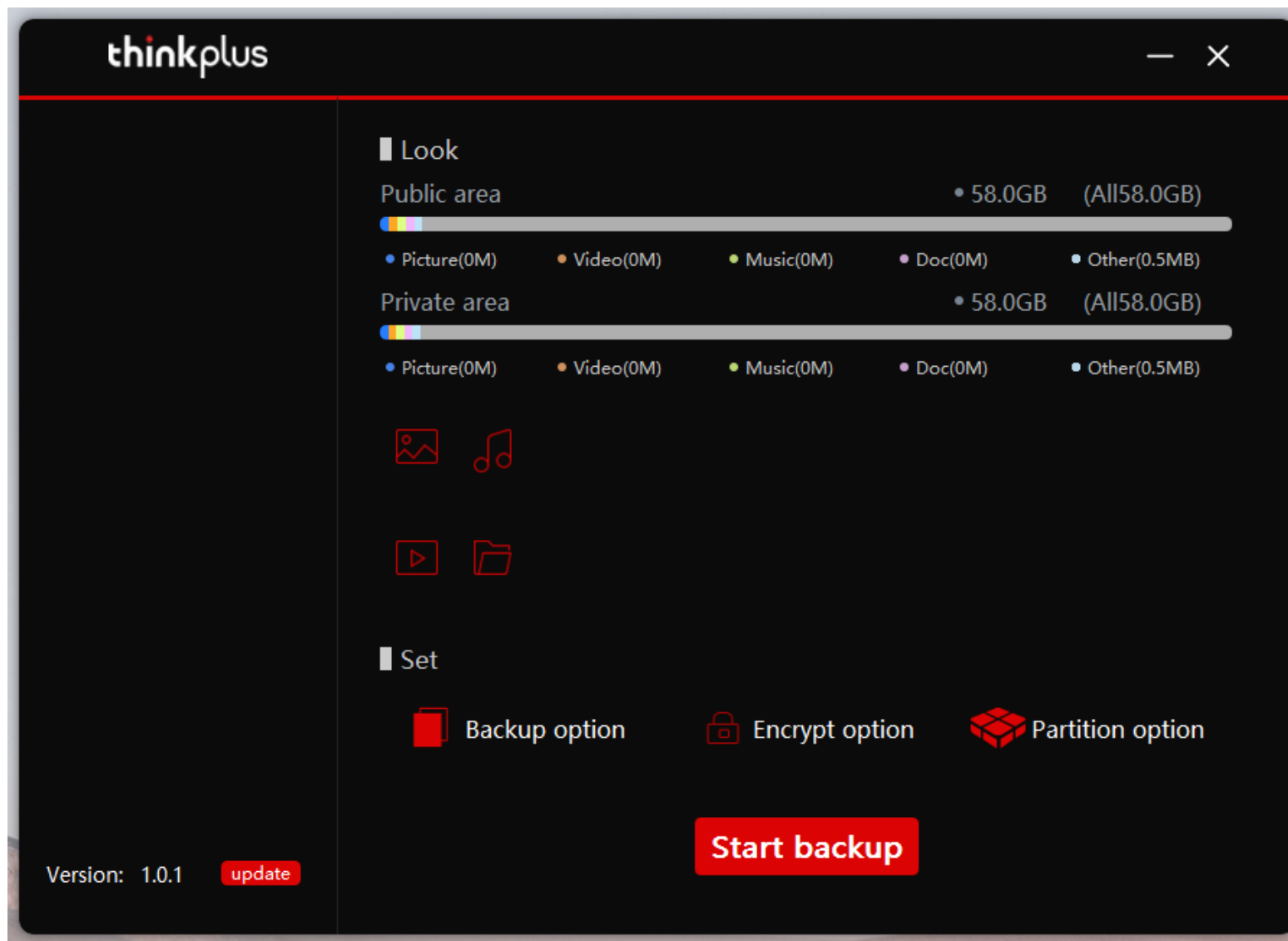
Password Leak

# Case Study: Lenovo ThinkPlus TSD303

1. Encrypted solid-state hard drive
2. Can be unlocked with user password or fingerprint
3. Comes with dedicated software
4. Uses regular M.2 SSD as storage
5. The data is indeed encrypted ✓



Image Credits: Alibaba.com



| No.   | Time     | Source  | Destination | Protocol | Length | Info        |
|-------|----------|---------|-------------|----------|--------|-------------|
| 163   | 1.659713 | 1.20.15 | host        | USB      | 8219   | URB_BULK in |
| 164   | 1.659769 | 1.20.15 | host        | USB      | 43     | URB_BULK in |
| → 165 | 1.661142 | host    | 1.20.15     | USB      | 27     | URB_BULK in |
| 166   | 1.661158 | host    | 1.20.15     | USB      | 27     | URB_BULK in |
| ← 167 | 1.662093 | 1.20.15 | host        | USB      | 539    | URB_BULK in |
| 168   | 1.662149 | 1.20.15 | host        | USB      | 43     | URB_BULK in |
| 169   | 1.662281 | host    | 1.20.15     | USB      | 27     | URB_BULK in |
| 170   | 1.662295 | host    | 1.20.15     | USB      | 27     | URB_BULK in |
| 171   | 1.662453 | 1.20.15 | host        | USB      | 8219   | URB_BULK in |

|                             |      |                                                 |                   |
|-----------------------------|------|-------------------------------------------------|-------------------|
| ▶ Frame 167: Packet, 539 b  | 0000 | 1b 00 10 30 de d1 8a c3 ff ff 00 00 00 00 09 00 | ...               |
| ▶ USB URB                   | 0010 | 01 01 00 14 00 ff 03 00 02 00 00 01 2f 46 00 69 | ...               |
| Leftover Capture Data [...] | 0020 | 00 6e 00 67 00 65 00 72 00 70 00 72 00 69 00 6e | ...n-g-e-r-p-r-i- |
|                             | 0030 | 00 74 00 31 00 00 00 00 00 00 00 00 00 00 00    | ...t.1....        |
|                             | 0040 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 0050 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 0060 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 0070 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 0080 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 0090 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 00a0 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 00b0 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 00c0 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 00d0 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 00e0 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 00f0 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 0100 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 0110 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 0120 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 0130 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 0140 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 0150 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 0160 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 0170 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 0180 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 0190 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 01a0 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 01b0 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 01c0 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 01d0 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 01e0 | 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    | .....             |
|                             | 01f0 | 00 00 00 00 00 00 00 00 00 00 00 66 6f 62 61    | .....fooba        |
|                             | 0200 | 72 00 00 00 00 00 00 00 00 00 00 01 00 00 00    | ...r.....         |
|                             | 0210 | 00 00 00 00 00 00 00 00 00 00 00 e9             | .....             |

Clear-text  
password is  
sent from  
the device to  
the  
computer

# The Attempted Fix

3 months after the report, vendor came back with a fix for me to test

The cleartext password is no longer sent

A binary blob is sent from the device to the computer

|      |                         |                         |                 |
|------|-------------------------|-------------------------|-----------------|
| 01e0 | 00 00 00 00 00 00 00 00 | 00 00 00 00 00 00 00 00 | .....           |
| 01f0 | 00 00 00 00 00 00 00 00 | 00 00 00 f4 64 2f d2 3e | .....d/.>       |
| 0200 | 8e ad 66 fe 59 bf 8d 0b | 62 c4 8f 01 00 00 00 00 | ..f.Y... b..... |
| 0210 | 00 00 00 00 00 00 00 00 | 00 00 d0                | .....           |



# The Attempted Fix

*I thought* it was a hash of the password

The UI performs one round of the password check

If we use a wrong password but force “checkPassword()” to return true, we only bypass the UI, the drive is NOT unlocked

# The Attempted Fix

*I thought* it was a hash of the password

→ The drive performs its own password validation  
Still a questionable practice, but not immediately broken







```
int32_t ok = vsGetPassWord(var_60, passwd)
char* passwd_2

if (ok == 0)
    passwd = &var_28

    if (QString::operator==(this: &user_input, passwd) == 0)
        passwd = 0xffffffff
        var_60 = nullptr
        passwd_2 = sub_72ec30(&var_2c, "Password error,please try again",
            var_60, passwd)
        var_8_3.b = 6
        goto label_71e0a1
```

```

007fb2d5 lea     esi, [edi+0x69]
007fb2d8 push    esi {var_5c}
007fb2d9 call    dword [vsGetPassWord]
→ 007fb2df push    0x3e8 {var_60_10}
007fb2e4 mov     dword [ebp+0x20 {var_24_1}], eax
007fb2e7 call    dword [Q]
007fb2ed add     esp, 0xc
007fb2f0 cmp     dword [e
007fb2f4 jge     0x7fb354

```

We have returned  
from  
vsGetPassword()

```

07a1dbf0 9a 07 00 00 3c 92 03 00 ....\...
07a1dbf8 c0 a1 85 00 54 35 c9 6f ....T5.o
07a1dc00 18 00 00 00 c8 b2 59 09 .....Y.
07a1dc08 10 dd a1 07 18 c0 a1 07 .....
07a1dc10 66 6f 6f 62 61 72 00 00 foobar..
07a1dc18 00 00 00 00 00 00 00 00 .....
07a1dc20 00 00 00 00 7f b5 d0 5a .....Z

```

The correct password  
can be seen on the  
stack

```

07a1dc68 f2 19 be 6f 50 16 35 03 ...oP.5.
07a1dc70 22 c4 42 a4 50 16 35 03 ".B.P.5.
07a1dc78 48 ee 33 03 50 16 35 03 H.3.P.5.
07a1dc80 00 00 00 00 70 7b d3 07 ....p{..
07a1dc88 0a 00 00 00 01 00 00 00 .....
07a1dc90 01 54 db 07 7c dc a1 07 .T...|...
07a1dc98 44 de a1 07 44 de a1 07 D...D...

```

Note: this is NOT the  
password I typed in

#### Stack Trace

| State    | Thread                | Frame # | Module         | Function           | PC         | SP        | FP        |
|----------|-----------------------|---------|----------------|--------------------|------------|-----------|-----------|
| Unfrozen | (*) 0x4eac @ 0x7fb2df |         |                |                    |            |           |           |
|          |                       | 0       | thinkplus.exe  | sub_7fb140 + 0x19f | 0x7fb2df   | 0x7a1dbd8 | 0x7a1dc34 |
|          |                       | 1       | thinkplus.exe  | sub_805910 + 0x1ed | 0x805afd   | 0x7a1dc3c | 0x7a1dca8 |
|          |                       | 2       | Qt5Widgets.dll | 0x703d2c2b         | 0x703d2c2b | 0x7a1dcb0 | 0x7a1dcc0 |
|          |                       | 3       | Qt5Widgets.dll | 0x703d200a         | 0x703d200a | 0x7a1dcc8 | 0x7a1dcc4 |
|          |                       | 4       | <unknown>      | 0x3351650          | 0x3351650  | 0x7a1dcc8 | 0x7a1dcc8 |
|          |                       | 5       | <unknown>      | 0x7db54d0          | 0x7db54d0  | 0x7a1dcd0 | 0x7a1dcc8 |



# TTD Analysis

TTD backtracks the “foobar” to its source  
It comes from SM4 decryption

The source is the buffer that was transferred on USB  
The key is derived from a timestamp thing

(\*) SM4 is a symmetric cipher, like AES





# Still Broken...

The fix is incomplete

The user password is still transferred to the program

The encryption does NOT protect it since the key can be derived in the same way

# Real Fix

A second fix was sent to me in December

This time it is properly fixed

User password is sent to the device for verification

# Disclosure Timeline

Aug 2025: Initial report

Sept – Nov 2025: Various failed/interim fixes

Dec 2025: Final fixes verified

Jan 2026: 3 CVEs assigned:

CVE-2025-13453

CVE-2025-13454

CVE-2025-13455 (\*not discussed in this talk)

Advisory: <https://iknow.lenovo.com.cn/detail/436983>

March 2026: This re//verse talk

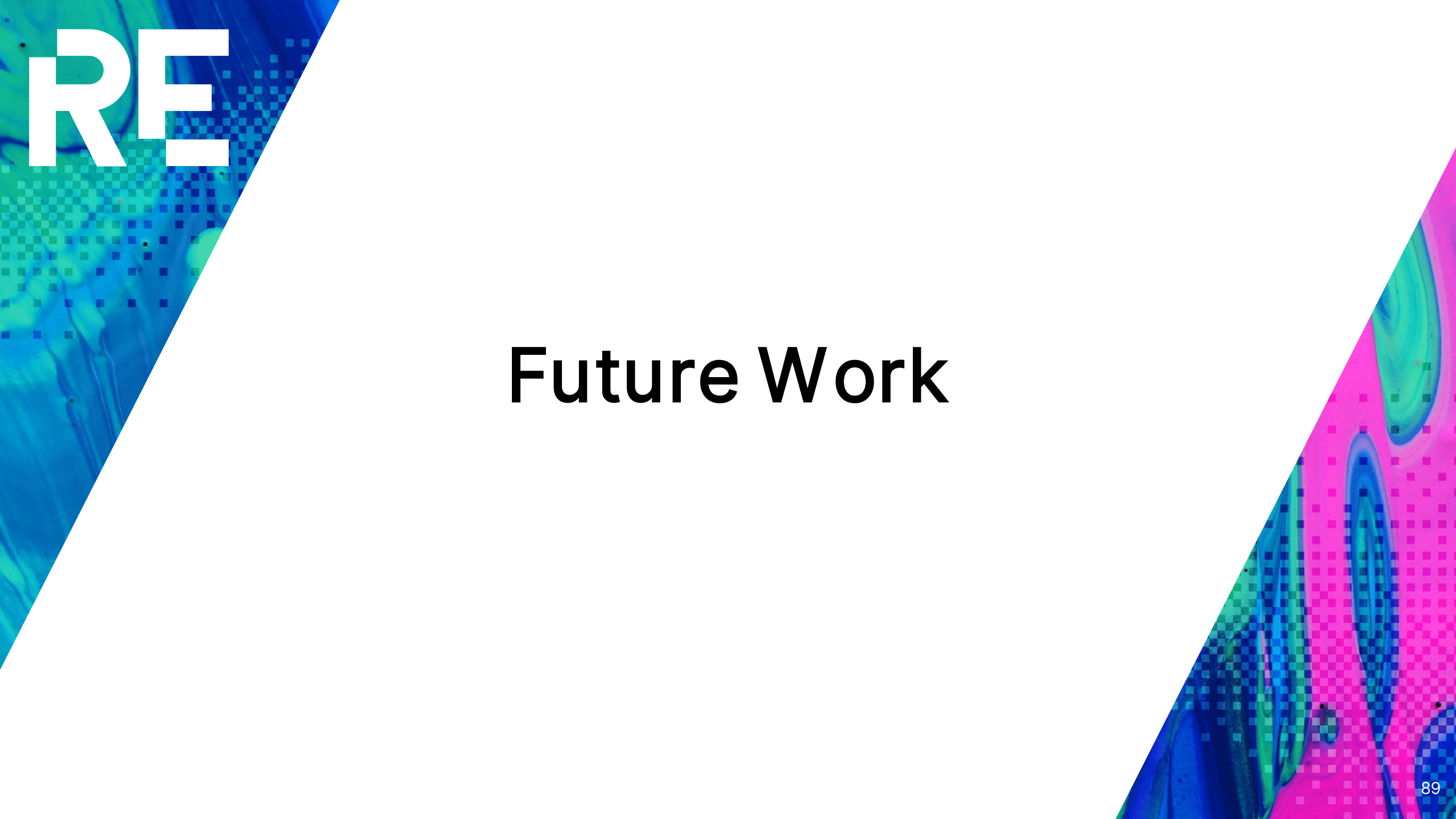
\* Users would need to get their existing devices serviced to adopt the fix



# Affected Vendors

Various other vendors that use the same SDK are similarly affected

| Vendor | Device           | Analysis                          |
|--------|------------------|-----------------------------------|
|        | TSD303 1TB       | Cleartext password disclosure     |
|        | 128GB            | Cleartext password disclosure     |
|        | 32GB Fingerprint | Unencrypted storage on micro-SD   |
|        | GD10 1TB         | Unencrypted storage on micro-SD   |
|        | 128GB USB        | Cleartext password disclosure     |
|        | FM6 1TB          | Cleartext password disclosure     |
|        | 64GB             | AES-encrypted password disclosure |
|        | FS01 128GB       | Cleartext password disclosure     |
|        | D3 32GB          | AES-encrypted password disclosure |
|        | 256GB PSSD       | Cleartext password disclosure     |



# RE

## Future Work

# It All Started with Fingerprints

A password-locked encrypted USB device can be secure

But how does the fingerprint unlocking fit into this?

# The Security of Fingerprint Unlock

Case 1: the fingerprint gates the release of a key

→ Still insecure, needs some hardware knowledge

Case 2: the fingerprint device works with a TPM/security enclave

→ It would be secure by design

→ But I doubt that is the case given the cost



# The Security of Fingerprint Unlock

Case 1: the fingerprint gates the release of a key

→ Still insecure, needs some hardware knowledge

Case 2: the fingerprint device works with a TPM/security enclave

→ It would be secure by design

→ But I doubt that is the case given the cost

3, 2, 1.. New research



# More TTD Analysis

There are many more powerful things that can be done with TTD

We have ALL memory writes from the TTD trace

Can we reconstruct ALL strings that have existed in RAM?



# More TTD Analysis

This would enable us to discover the password “foobar” existed in RAM as cleartext automatically

AI can probably help develop the algorithm, but we still need to know what to ask for

The background features abstract, colorful patterns. On the left, a blue and green pattern with a grid overlay is visible. On the right, a pink and blue pattern with a grid overlay is visible. The central area is white.

# RE

## Lessons Learned & Wrap-up

# Key Takeaways

"Hardware security" can be pure marketing  
Software components can be the weakest link  
Client-side validation is never secure  
Don't trust vendor security claims - verify!

# TTD Best Practices

Record targeted operations, not entire sessions

Check API calls to understand program behaviors

Use backtracking for crypto key/network traffic analysis

Combine with static analysis for best results

Integrate into your standard RE workflow

# Resources (Encrypted USB)

**Black Hat 2017: Attacking Encrypted USB Keys the Hard(ware) Way**

<https://elie.net/talk/attacking-encrypted-usb-keys-the-hardware-way>

**SySS 2010: FIPS 140-2 Certified USB Drives Bypass**

[https://www.schneier.com/blog/archives/2010/01/fips\\_140-2\\_leve.html](https://www.schneier.com/blog/archives/2010/01/fips_140-2_leve.html)

**SySS 2022: Hacking Secure USB Flash Drives (Matthias Deeg)**

<https://blog.syss.com/posts/hacking-usb-flash-drives-part-1/>

**NoHat 2025: Your Security Update is Not Secure Enough (Matthias Deeg)**

<https://www.nohat.it/talks>

# Resources (TTD)

Awesome-TTD: <https://github.com/xusheng6/awesome-ttd>

BN & TTD Documentation:

<https://docs.binary.ninja/guide/debugger/dbgeng-ttd.html>  
(stable)

<https://dev-docs.binary.ninja/guide/debugger/dbgeng-ttd.html> (dev)

# RE

**Thank you!**

**Questions?**

[xusheng@vector35.com](mailto:xusheng@vector35.com)

