

**OFFZONE
2026**

Vulnerabilities in Full-Disk Encryption Systems

Maxim Suhanov

CICADA8



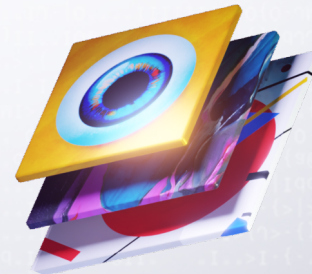
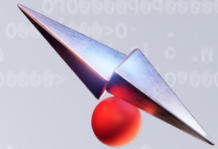
Maxim Suhanov

CICADA8

Vulnerability research since 2007

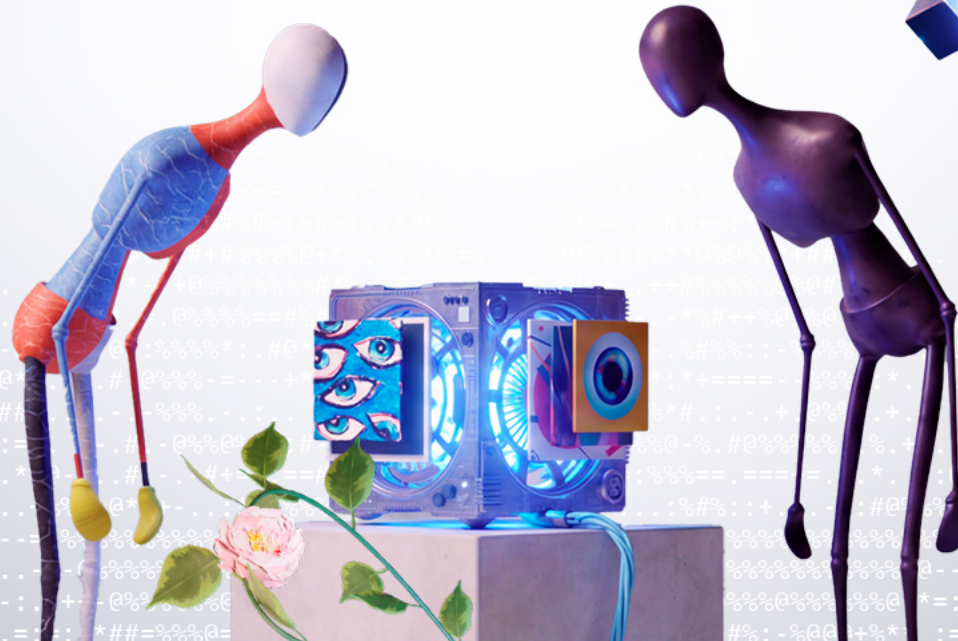
MSRC's leaderboard:

2023 Q4, 2024 Q2, 2024 Q3



@errno_fail

Threats & Threat Models



Common Threats

Stolen Computer (or lost, or seized)

one-time access to the device
owner is usually aware of that
*If powered on: as secure as (a)
the lock screen is, (b) drive-by
DMA protections are*

Evil Maid (covert access)

one-time or repeated access
owner is unaware

Rogue Employee

privesc via physical access
unpriv user == attacker
unpriv user != data owner
*Password is enough to boot,
but not to decrypt*

Access to Cloud Storage

one-time or repeated access
owner is unaware

Malware Access to Unmounted Volume

repeated remote access
owner is unaware

Dual-boot (2nd OS backdoored)

Watermarking

prior upload access to
encrypted storage
one-time access to the storage
*Prove that your encrypted
data is there (without the key)*

Common Threats (more naive)

Reverse Engineering (of proprietary box)

repeated access
unpriv user == attacker

Can boot, can't decrypt

Pressure

legal (key disclosure laws,
border crossing)
rubber-hose cryptanalysis



Threat Model: VeraCrypt

VeraCrypt does **not**:

evil maids are out-of-scope

- Secure any data on a computer if an attacker has physical access to the computer before or while VeraCrypt is running on it.

- Preserve/verify the integrity or authenticity of encrypted or decrypted data.

but XTS has some guarantees
(i.e., no bit flips like in CBC)

- Prevent traffic analysis when encrypted data is transmitted over a network.

they don't talk about *decrypted* data, so
issues during network boot (or cloud mount)
are out-of-scope

<https://veracrypt.io/en/Security%20Model.html>

Threat Model: BitLocker (from 2006)

However, to have a secure boot process we also need to authenticate the data from the disk. We don't want an attacker to modify the OS code in order to weaken the OS security. The normal cryptographic solution is to use an authentication code, but as we will see this is not practical. Therefore, we are left with using the encryption as a poor-man's authentication.

Imagine an attacker trying to break into the laptop using only software. He boots into some other OS. Because the PCRs are different he can't unseal the BitLocker key, so he can't read the encrypted volume. However, the attacker can modify the ciphertext on the hard disk in the hope of introducing a weakness in the OS. He then boots the machine normally, and exploits the created weakness during the boot or login process to gain access to the machine.

stolen laptop with seamless boot enabled
(utilizing TPM)

<https://download.microsoft.com/download/0/2/3/0238acaf-d3bf-4a6d-b3d6-0a0be4bbb36e/bitlockercipher200608.pdf>

Threat Model: BitLocker (from 2006, cont.)

The attacker succeeds if he can modify a ciphertext such that the corresponding plaintext change has some non-random property.

→
XTS-style protections defined before XTS
became a standard

In practice, we expect that many laptops will be used in the TPM-only mode and that scenario is the main driver for the disk cipher design.

→
Microsoft follows that for 20 years!
2006 – 2026 – ...



Further Discussion

XTS mode revisited: high hopes for key scopes?

Milan Brož^{1,2}
gmazyland@gmail.com

Vladimír Sedláček²
vlada.sedlacek@mail.muni.cz

¹Cryptsetup project

²Faculty of Informatics, Masaryk University, Czechia

October 28, 2025

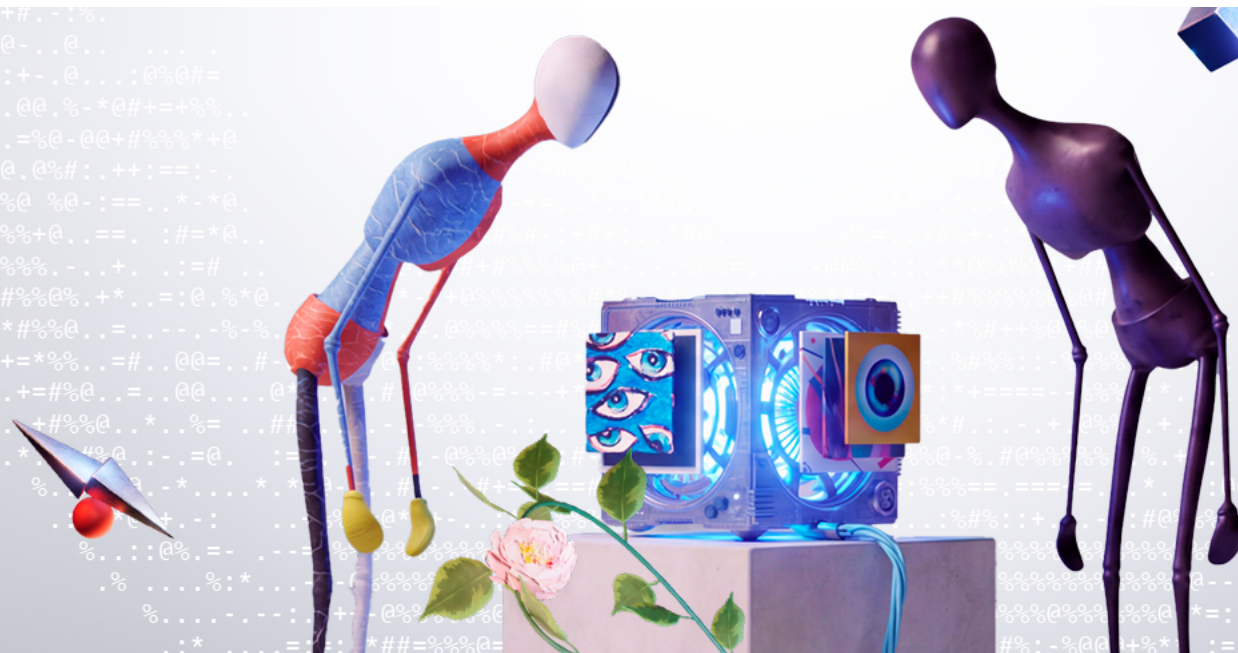
<https://github.com/mbroz/xts-paper/blob/main/xts-paper-v2.pdf>



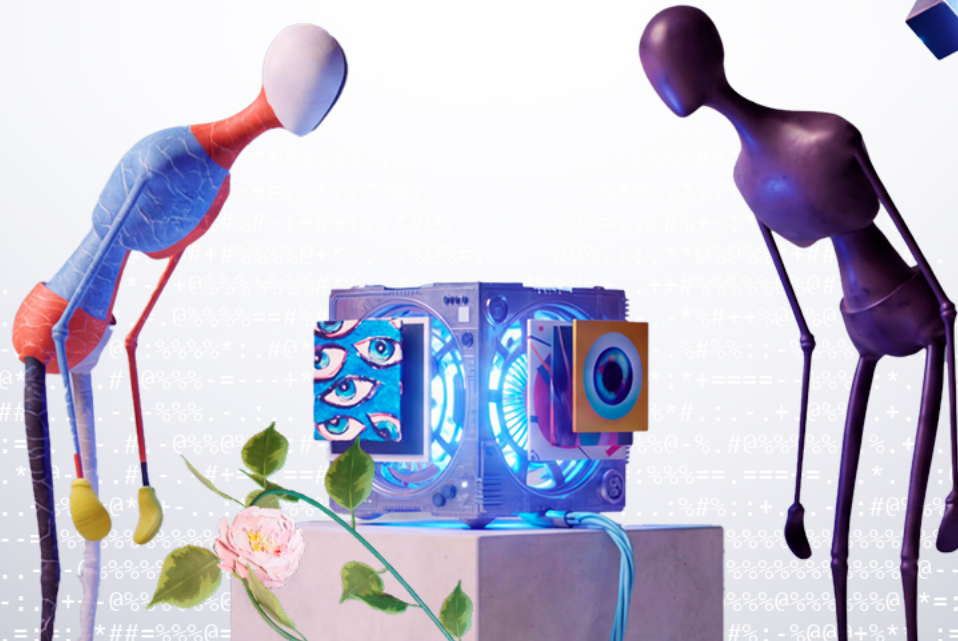
Further Discussion (cont.)

sequent deauthentication).” This model expects the user to recognize when the protected device “left the owner’s control”, which is not always feasible.

The real-world threat model also considers situations where the attacker can manipulate the ciphertext after repeatable access to the encrypted device, even performing some form of “traffic analysis”. This is important, as encrypted images are often stored in cloud environments. Section D.4.3 in [12] suggests (in



“Corner” Cases (not really corner)

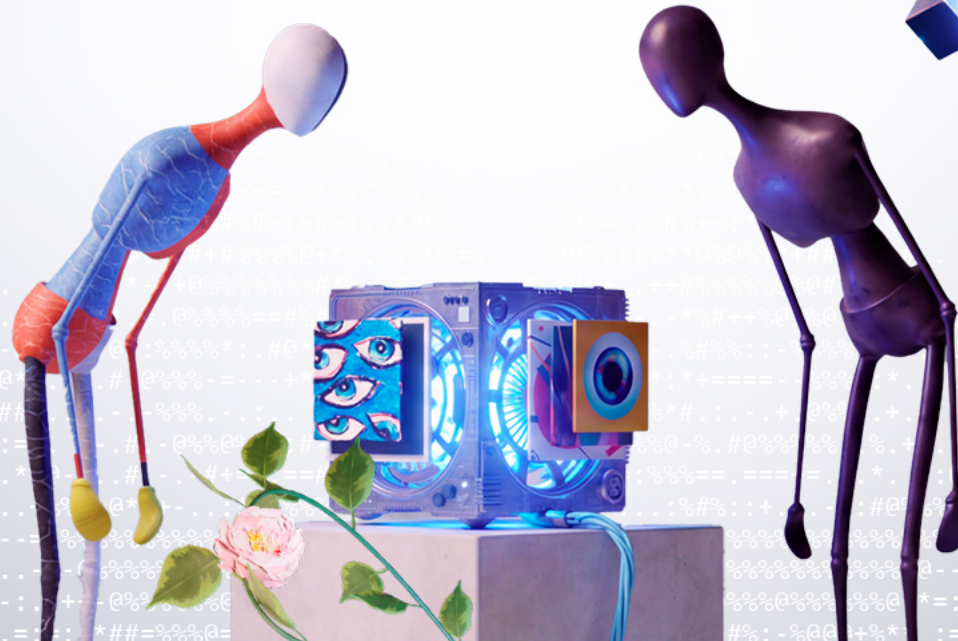


“Corner” Cases (cont.)

- Tampering via HDD/SSD firmware backdoors
excluding DMA attacks
inject a payload into encrypted volume → pwned
- One-time physical access can be equal to the repeated one
in the TPM-only setups (seamless boot up to the OS login screen)
- Dual-boot can lead to repeated remote access
to unmounted encrypted volumes
not relevant if an encrypted volume is mounted (→ self-pwn)



Encryption & Decryption Wrappers



BestCrypt Volume Encryption

- Plaintext BCVE metadata is treated as is on a mounted volume
(Not encrypted/decrypted.)

Python equivalent (greatly simplified)

```
def Encrypt(Buffer, LBA, Key, Tweak):  
    if Buffer.startswith(MAGIC_SIGNATURE):  
        return Buffer  
  
    return AESXTS_Encrypt(Buffer, LBA, Key, Tweak)  
  
def Decrypt(Buffer, LBA, Key, Tweak):  
    if Buffer.startswith(MAGIC_SIGNATURE):  
        return Buffer  
  
    return AESXTS_Decrypt(Buffer, LBA, Key, Tweak)
```



BestCrypt Volume Encryption (cont.)

- Write path: partial plaintext disclosure
Data in a single sector can belong to different security domains!
- Read path: plaintext injection
- This is **NotCVE-2026-0004**
Discovered and re-discovered by:
Juraj Horňák (2016), Jan Vojtěšek (2017), Maxim Suhanov (2025)
- The XTS guarantees were thrown away by the vendor



BestCrypt Volume Encryption (cont.)

On-disk:

```
0238A9A0: 61 6D 23 CD 93 76 EE 0C A8 0E D1 3C 17 6F 78 1C am#...v...<.ox.  
0238A9B0: 48 15 FF 43 02 65 C5 8A 01 1A FF CC 1A 3F C2 58 H..C.e.....?..X  
0238A9C0: B8 80 43 1F 96 E3 E2 AA DA 88 00 85 2B 94 3C AE ..C.....+<..  
0238A9D0: D1 6C EB 32 13 9D 51 E9 2A FF 0E DA 8C 43 06 37 .l.2..Q.*...C.7  
0238A9E0: 09 41 2D 15 2F 7B 4C CF A6 37 3D 52 69 D0 65 DA .A-./{L..7=ri.e.  
0238A9F0: 25 97 70 5F 33 94 FD C4 B2 49 63 B7 38 F9 AA 45 %p_3....Ic.8..E  
0238AA00: DD A2 6A 7E 3A 59 FF 45 3E 35 0A 44 BC B4 CD D5 j~!Y.E>5.D...  
0238AA10: 64 64 64 64 64 64 64 64 64 64 64 64 64 64 64 ddddddddddddddd  
0238AA20: 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 aaaaaaaaaaaaaaaaa  
0238AA30: 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 aaaaaaaaaaaaaaaaa  
0238AA40: 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 aaaaaaaaaaaaaaaaa  
0238AA50: 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 aaaaaaaaaaaaaaaaa  
0238AA60: 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 aaaaaaaaaaaaaaaaa  
0238AA70: 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 aaaaaaaaaaaaaaaaa  
0238AA80: 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 aaaaaaaaaaaaaaaaa  
0238AA90: 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 aaaaaaaaaaaaaaaaa  
0238AAA0: 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 aaaaaaaaaaaaaaaaa  
0238AAB0: 61 61 61 61 61 61 61 61 61 61 61 61 61 61 61 aaaaaaaaaaaaaaaaa  
0238AAC0: 62 62 62 62 62 62 62 62 62 62 62 62 62 62 62 bbbbbbbbbbbbbbbbb  
0238AAD0: 62 62 62 62 62 62 62 62 62 62 62 62 62 62 62 bbbbbbbbbbbbbbbbb  
0238AAE0: 62 62 62 62 62 62 62 62 62 62 62 62 62 62 62 bbbbbbbbbbbbbbbbb  
0238AAF0: 62 62 62 62 62 62 62 62 62 62 62 62 62 62 62 bbbbbbbbbbbbbbbbb  
0238AB00: 62 62 62 62 62 62 62 62 62 62 62 62 62 62 62 bbbbbbbbbbbbbbbbb  
0238AB10: 62 62 62 62 62 62 62 62 62 62 62 62 62 62 62 bbbbbbbbbbbbbbbbb  
0238AB20: 62 62 62 62 62 62 62 62 62 62 62 62 62 62 62 bbbbbbbbbbbbbbbbb  
0238AB30: 62 62 62 62 62 62 62 62 62 62 62 62 62 62 62 bbbbbbbbbbbbbbbbb  
0238AB40: 62 62 62 62 62 62 62 62 62 62 62 62 62 62 62 bbbbbbbbbbbbbbbbb
```

Mounted:

The screenshot shows a Windows File Explorer window with the address bar set to 'This PC > BC_TEST (E:) > test'. The 'test' folder contains a file named 'test.txt'. The 'test.txt' file is open, showing a list of 'ENCRYPTED TEXT' entries. A blue arrow points from the 'test.txt' file in the File Explorer to the 'test.txt' file in the BestCrypt Volume Encryption application window. The BestCrypt Volume Encryption application window shows the volume status as 'Encrypted and mounted' and provides information about the license and version.

Volume	Status	Capacity	Free space	File system	System or Boot	Volume type	Algorithm
(C:)	Not encrypted	63.25 GB	42.14 GB	NTFS	Boot	Basic	
BC_TEST (E:)	Encrypted and mounted	15.98 GB	15.94 GB	NTFS		Basic	AES (XTS mode)

About BestCrypt Volume Encryption

BestCrypt Volume Encryption
Copyright © 2005 - 2025 Jetico Inc. Oy

License information
This copy of BestCrypt Volume Encryption is demo version!
Expiration date:
Monday, May 5, 2025
The license allows all features of commercial version within the evaluation period.

Version information
User Interface version: 5.13.6
Driver version: 4.54

Vendor information
Web page: <http://www.jetico.com>
E-Mail address: <mailto:support@jetico.com>

Plaintext injection
via one-time offline attack

BestCrypt Volume Encryption (cont.)

Mounted:

Remotely spray rows into DB

```
000134E0: 3A 59 FF 45 3E 35 0A 44 BC B4 CD D5 32 00 30 00 :Y.E>5.D...2.0.
000134F0: 38 00 34 00 33 00 31 00 38 00 36 00 30 00 35 00 8.4.3.1.8.6.0.5.
00013500: 37 00 30 00 37 00 38 00 31 00 36 00 31 00 33 00 7.0.7.8.1.6.1.3.
00013510: 37 00 4F 87 08 07 00 51 59 00 00 00 4A 00 6F 00 J.o.h.n...D.o.e.
00013520: 68 00 6E 00 20 00 44 00 6F 00 65 00 20 00 35 00 h.n...D.o.e.
00013530: 38 00 30 00 39 00 36 00 35 00 30 00 30 00 38 00 8.0.9.6.5.0.0.8.
00013540: 35 00 33 00 30 00 39 00 39 00 37 00 35 00 34 00 5.3.0.9.9.7.5.4.
00013550: 39 00 36 00 35 00 32 00 38 00 36 00 34 00 30 00 9.6.5.2.8.6.4.0.
00013560: 37 00 35 00 3D 87 07 07 00 2D 59 00 00 00 DD A2 7.5.=...-Y.
00013570: 6A 7E 3A 59 FF 45 3E 35 0A 44 BC B4 CD D5 33 00 j~:Y.E>5.D...3.
00013580: 35 00 32 00 38 00 34 00 33 00 39 00 30 00 38 00 5.2.8.4.3.9.0.8.
00013590: 34 00 36 00 32 00 32 00 33 00 31 00 32 00 39 00 4.6.2.2.3.1.2.9.
000135A0: 39 00 34 00 4F 87 06 07 00 51 59 00 00 00 4A 00 9.4.0...QY...J.
000135B0: 6F 00 68 00 6E 00 20 00 44 00 6F 00 65 00 20 00 o.h.n...D.o.e.
000135C0: 38 00 38 00 35 00 30 00 36 00 32 00 34 00 37 00 8.8.5.0.6.2.4.7.
000135D0: 33 00 34 00 30 00 32 00 36 00 37 00 32 00 30 00 3.4.0.2.6.7.2.0.
000135E0: 30 00 31 00 35 00 30 00 39 00 31 00 33 00 36 00 0.1.5.0.9.1.3.6.
000135F0: 35 00 33 00 34 00 3D 87 05 07 00 2D 59 00 00 00 5.3.4.=...-Y.
00013600: DD A2 6A 7E 3A 59 FF 45 3E 35 0A 44 BC B4 CD D5 j~:Y.E>5.D...
00013610: 33 00 30 00 30 00 34 00 31 00 36 00 33 00 33 00 3.0.0.4.1.6.3.3.
00013620: 39 00 38 00 32 00 32 00 37 00 32 00 34 00 33 00 9.8.2.2.7.2.4.3.
00013630: 36 00 37 00 39 00 4F 87 04 07 00 51 59 00 00 00 6.7.9.0...QY...
00013640: 4A 00 6F 00 68 00 6E 00 20 00 44 00 6F 00 65 00 J.o.h.n...D.o.e.
00013650: 20 00 32 00 34 00 30 00 32 00 38 00 39 00 39 00 2.4.0.2.8.9.9.
00013660: 35 00 35 00 33 00 32 00 35 00 32 00 38 00 38 00 5.5.3.2.5.2.8.8.
00013670: 33 00 34 00 35 00 39 00 36 00 38 00 37 00 37 00 3.4.5.9.6.8.7.7.
00013680: 36 00 37 00 30 00 37 00 3D 87 03 07 00 2D 59 00 6.7.0.7.=...-Y.
00013690: 00 00 DD A2 6A 7E 3A 59 FF 45 3E 35 0A 44 BC B4 j~:Y.E>5.D...
000136A0: CD D5 36 00 33 00 31 00 30 00 39 00 37 00 34 00 6.3.1.0.9.7.4.
000136B0: 39 00 38 00 32 00 37 00 37 00 33 00 33 00 33 00 9.8.2.7.7.3.3.3.
000136C0: 39 00 38 00 31 00 32 00 4F 87 02 07 00 51 59 00 9.8.1.2.0...QY.
000136D0: 00 00 4A 00 6F 00 68 00 6E 00 20 00 44 00 6F 00 J.o.h.n...D.o.
000136E0: 65 00 20 00 32 00 37 00 39 00 39 00 34 00 32 00 e...2.7.9.9.4.2.
000136F0: 38 00 32 00 38 00 38 00 30 00 35 00 30 00 31 00 8.2.8.8.0.5.0.1.
00013700: 32 00 39 00 39 00 35 00 37 00 37 00 31 00 36 00 2.9.9.5.7.7.1.6.
00013710: 33 00 30 00 31 00 32 00 33 00 3D 87 01 07 00 2D 3.0.1.2.3.=...-
00013720: 59 00 00 00 DD A2 6A 7E 3A 59 FF 45 3E 35 0A 44 Y....j~:Y.E>5.D
00013730: BC B4 CD D5 37 00 37 00 38 00 31 00 32 00 33 00 7.7.8.1.2.3.
00013740: 39 00 35 00 31 00 31 00 32 00 30 00 36 00 35 00 9.5.1.1.2.0.6.5.
00013750: 36 00 31 00 34 00 30 00 38 00 4D 87 00 07 00 51 6.1.4.0.8.M...Q
00013760: 55 00 00 00 4A 00 6F 00 68 00 6E 00 20 00 44 00 U...J.o.h.n...D.
00013770: 6F 00 65 00 20 00 38 00 30 00 30 00 31 00 33 00 o.e...8.0.0.1.3.
00013780: 36 00 31 00 32 00 31 00 38 00 33 00 35 00 34 00 6.1.2.1.8.3.5.4.
00013790: 35 00 38 00 38 00 39 00 31 00 36 00 36 00 35 00 5.8.8.9.1.6.6.5.
000137A0: 33 00 34 00 35 00 33 00 31 00 3D 86 7F 07 00 2D 3.4.5.3.1.=...-
```

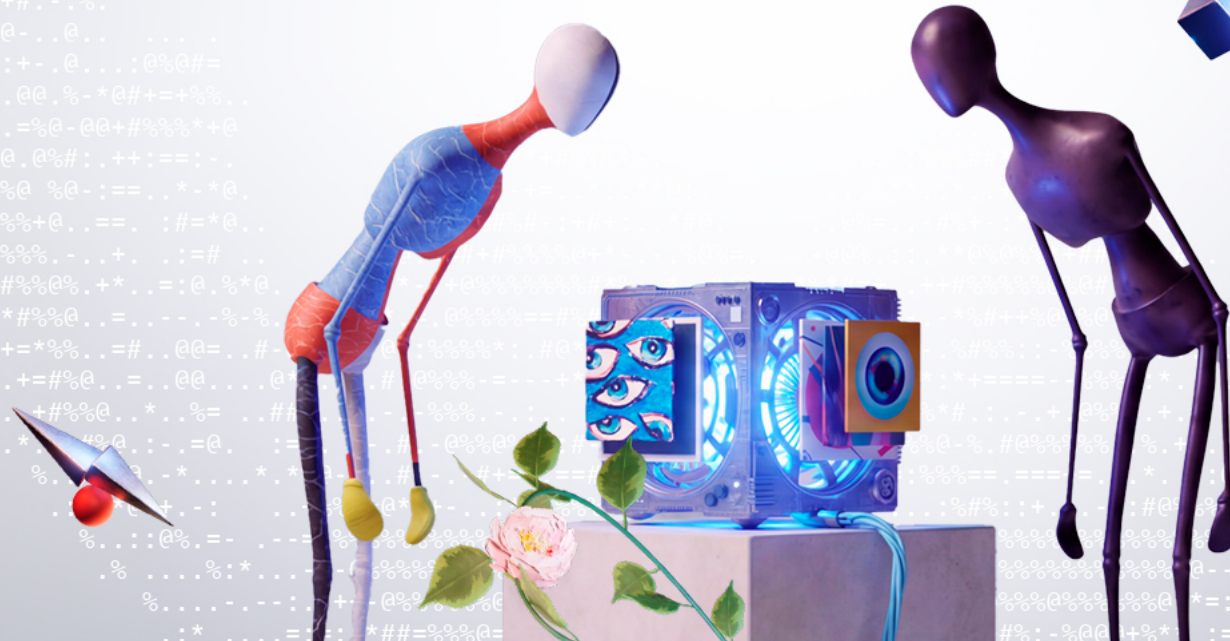
On-disk:

Nearby data is no longer encrypted

```
21D7884D0: BB 3F 86 68 01 5D F3 38 08 1A DD 0D 40 0B 7F 79 .?..h.].8....@..y
21D7884E0: D9 11 E6 6E 01 D5 6E 8F 87 F8 D0 F9 CE B6 5D 54 ....n..n.....]T
21D7884F0: 3A A6 17 13 FD 68 9B D8 DB B6 67 C9 6D 7B AD 58 ....h....g.m{.X
21D788500: 91 28 94 D4 A5 BC 7A 60 18 EB 19 67 A4 4B 5D 9B .(....z'....g.K]
21D788510: F0 78 3A 44 43 9A 3F D7 DB 10 EE 33 9B C8 B2 4D .x:DC.?....3...M
21D788520: D0 FD 6B 0F 5A 1E 77 F3 C6 04 5A DC F4 F8 A3 24 ...k.Z.w...Z...$
21D788530: F7 B1 2D F3 27 73 CC 98 D7 00 21 7E A2 E8 C9 B4 ...-'s.....!~...
21D788540: 33 6B 21 96 93 D6 EF DE 2A E2 C7 0A A3 20 1E 79 3k!.....*.....y
21D788550: 23 6F 87 0C 8A 9A EA 50 0E 3A 78 D1 03 FD D4 39 #o.....P.:x...9
21D788560: 56 D7 A4 A6 56 43 37 0B 08 3E B5 11 42 89 9C E3 V...VC7...>..B...
21D788570: 44 2A EA 2E 36 5F F2 E1 F3 03 9B 88 85 9E D0 67 D*..6_.....g
21D788580: 08 D9 BC 3E 45 F7 FB BB 9F 11 09 17 F8 44 7D 07 ...>E.....D}
21D788590: C4 5B 89 6C 92 B7 CE F9 FF B8 91 82 48 AD 3A 7F [.l.....H.:
21D7885A0: 1A 8C 89 C9 A4 F8 C5 B0 1C 0D A5 69 71 4C F1 D7 .....iQL
21D7885B0: FF CE B2 6F 87 06 93 96 BD D6 08 D3 E5 DC 21 CC ...o.....!
21D7885C0: 58 28 5A D0 2B 25 1D FD D2 3C A5 F1 34 C4 69 2B X(Z+%...<..4.i+
21D7885D0: 83 D0 53 E9 5A 5B 7E 78 EA 94 85 F9 72 AD D9 13 ...S.Z[~x...r...
21D7885E0: 39 57 6A CB 0E 56 B3 9B 04 5E 3B 10 56 B0 1C 12 9Wj..V...^;..u..
21D7885F0: DE 15 09 20 C0 B3 69 FF 19 CB 3D BD 75 A1 A6 14 ...i...=..u...
21D788600: DD A2 6A 7E 3A 59 FF 45 3E 35 0A 44 BC B4 CD D5 j~:Y.E>5.D...
21D788610: 33 00 30 00 30 00 34 00 31 00 36 00 33 00 33 00 3.0.0.4.1.6.3.3.
21D788620: 39 00 38 00 32 00 32 00 37 00 32 00 34 00 33 00 9.8.2.2.7.2.4.3.
21D788630: 36 00 37 00 39 00 4F 87 04 07 00 51 59 00 00 00 6.7.9.0...QY...
21D788640: 4A 00 6F 00 68 00 6E 00 20 00 44 00 6F 00 65 00 J.o.h.n...D.o.e.
21D788650: 20 00 32 00 34 00 30 00 32 00 38 00 39 00 39 00 2.4.0.2.8.9.9.
21D788660: 35 00 35 00 33 00 32 00 35 00 32 00 38 00 38 00 5.5.3.2.5.2.8.8.
21D788670: 33 00 34 00 35 00 39 00 36 00 38 00 37 00 37 00 3.4.5.9.6.8.7.7.
21D788680: 36 00 37 00 30 00 37 00 3D 87 03 07 00 2D 59 00 6.7.0.7.=...-Y.
21D788690: 00 00 DD A2 6A 7E 3A 59 FF 45 3E 35 0A 44 BC B4 j~:Y.E>5.D...
21D7886A0: CD D5 36 00 33 00 31 00 30 00 39 00 37 00 34 00 6.3.1.0.9.7.4.
21D7886B0: 39 00 38 00 32 00 37 00 37 00 33 00 33 00 33 00 9.8.2.7.7.3.3.3.
21D7886C0: 39 00 38 00 31 00 32 00 4F 87 02 07 00 51 59 00 9.8.1.2.0...QY.
21D7886D0: 00 00 4A 00 6F 00 68 00 6E 00 20 00 44 00 6F 00 J.o.h.n...D.o.
21D7886E0: 65 00 20 00 32 00 37 00 39 00 39 00 34 00 32 00 e...2.7.9.9.4.2.
21D7886F0: 38 00 32 00 38 00 38 00 30 00 35 00 30 00 31 00 8.2.8.8.0.5.0.1.
21D788700: 32 00 39 00 39 00 35 00 37 00 37 00 31 00 36 00 2.9.9.5.7.7.1.6.
21D788710: 33 00 30 00 31 00 32 00 33 00 3D 87 01 07 00 2D 3.0.1.2.3.=...-
21D788720: 59 00 00 00 DD A2 6A 7E 3A 59 FF 45 3E 35 0A 44 Y....j~:Y.E>5.D
21D788730: BC B4 CD D5 37 00 37 00 38 00 31 00 32 00 33 00 7.7.8.1.2.3.
21D788740: 39 00 35 00 31 00 31 00 32 00 30 00 36 00 35 00 9.5.1.1.2.0.6.5.
21D788750: 36 00 31 00 34 00 30 00 38 00 4D 87 00 07 00 51 6.1.4.0.8.M...Q
21D788760: 55 00 00 00 4A 00 6F 00 68 00 6E 00 20 00 44 00 U...J.o.h.n...D.
21D788770: 6F 00 65 00 20 00 38 00 30 00 30 00 31 00 33 00 o.e...8.0.0.1.3.
21D788780: 36 00 31 00 32 00 31 00 38 00 33 00 35 00 34 00 6.1.2.1.8.3.5.4.
21D788790: 35 00 38 00 38 00 39 00 31 00 36 00 36 00 35 00 5.8.8.9.1.6.6.5.
```


BestCrypt Volume Encryption (cont.)

- **NEVER** use BestCrypt Volume Encryption when hosting Internet-facing databases
- Even when publishing indirectly (e.g., via web apps)



Defusing the Elephant diffuser (BitLocker)

- This bug is present since Vista introduced the Elephant diffuser
- **Not exploitable**, affects the boot loader only and cryptsetup before 68d4749 (2025-08)

Python equivalent (greatly simplified)

```
def bootmgr_ReadAndDecrypt(LBA, ...):  
    if LBA in metadata_loc: # 'metadata_loc' is taken from the volume header,  
                           # which isn't authenticated in a no-TPM mode
```

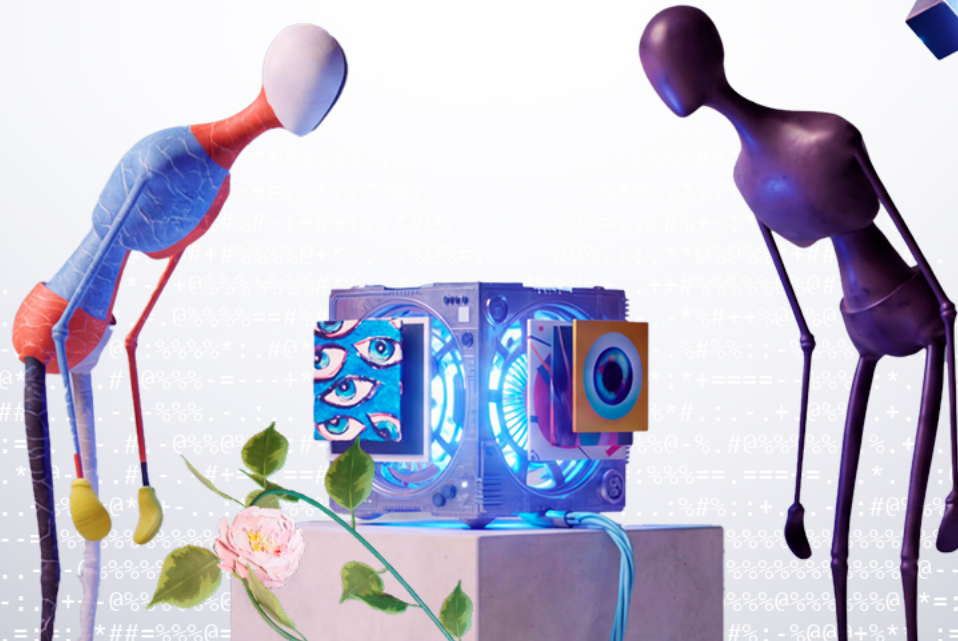
```
        return b'\x00' * 512
```

```
    Buffer = ReadFromDisk(LBA)
```

```
    return AESElephant_Decrypt(Buffer, LBA, ...)
```



BitLocker



Architecture of BitLocker

- Two encryption (filter) drivers:
fvevol.sys (usual)
dumpfve.sys (crash and hibernation)
- Two boot paths:
Normal (to the login screen)
WinRE (to the recovery app)
- Unmeasured ACE during the early boot → pwned
in the TPM-only and TPM+NK modes
- Failed to unseal the key? Need the recovery key!



BitLocker: Some Modes of Operation

- Cleartext Protector
 - encrypted but no secret required to mount
- PIN (password)
 - no TPM-based integrity protection for boot components
- TPM-only
 - seamless boot up to the login screen
 - usually bound to PCRs 7 (Secure Boot) and 11
- TPM + PIN
 - PIN alone isn't enough to decrypt
- TPM plus Network Key
 - seamless boot while in the trusted wired network
 - TPM plus PIN if the network unlock server is unreachable
- External Key (file), Smart Card



BitLocker: TPM plus Network Key

- Works via a proprietary extension to the DHCP
- 1st key encryption layer: unwrapped by the server
- 2nd key encryption layer: unwrapped by the TPM
- If that fails, another protector (TPM+PIN) is used



BitLocker: TPM plus Network Key

- Works via a proprietary extension to the DHCP
- 1st key encryption layer: unwrapped by the server
- 2nd key encryption layer: unwrapped by the TPM
- If that fails, another protector (TPM+PIN) is used
- A successful boot authenticates two sides:
computer and network
- A successful logon authenticates one more side:

user



BitLocker: WinRE

- Can recover from typical boot problems
 - Microsoft thinks so
- Can include OEM-specific recovery tools
- Automatic recovery is launched after two failed boot attempts
 - Otherwise, a simple menu is displayed to a user



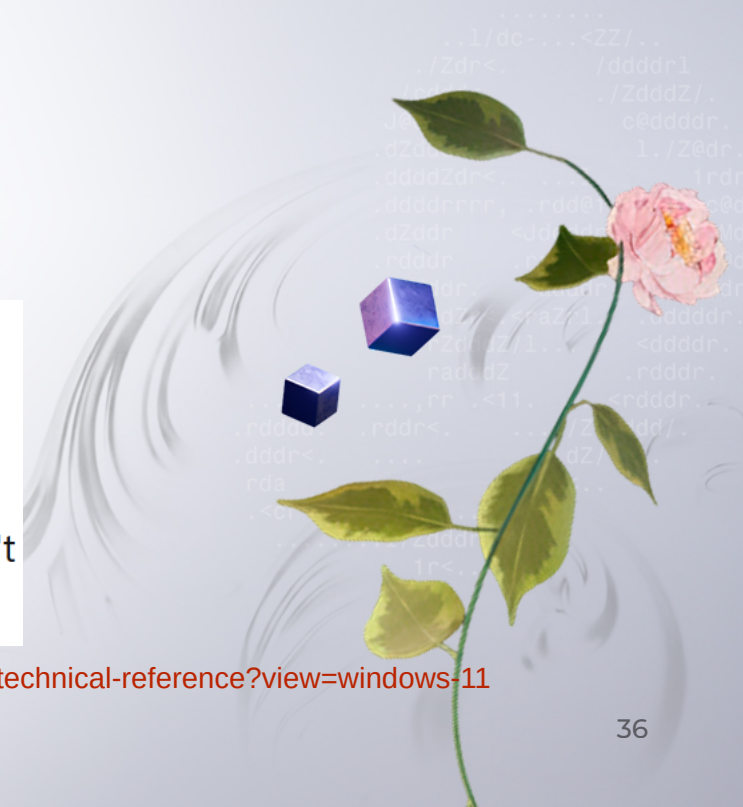
BitLocker: WinRE

- Can recover from typical boot problems
Microsoft thinks so
- Can include OEM-specific recovery tools
- Automatic recovery is launched after two failed boot attempts
Otherwise, a simple menu is displayed to a user

What's new with WinRE for Windows 11

- You can now run most tools within WinRE without selecting an administrator account and entering the password. When booted into the Recovery environment, encrypted files won't be accessible unless the user has the key to decrypt the volume.

<https://learn.microsoft.com/en-us/windows-hardware/manufacture/desktop/windows-recovery-environment--windows-re--technical-reference?view=windows-11>



BitLocker: WinRE and TPM

- If WIM is trusted:
the keyring is passed from the boot manager to WinRE
- Else:
the keyring is wiped before launching WinRE
- WIM is trusted when:
its hash is in the volume metadata
(AES-CCM-encrypted SHA-256)

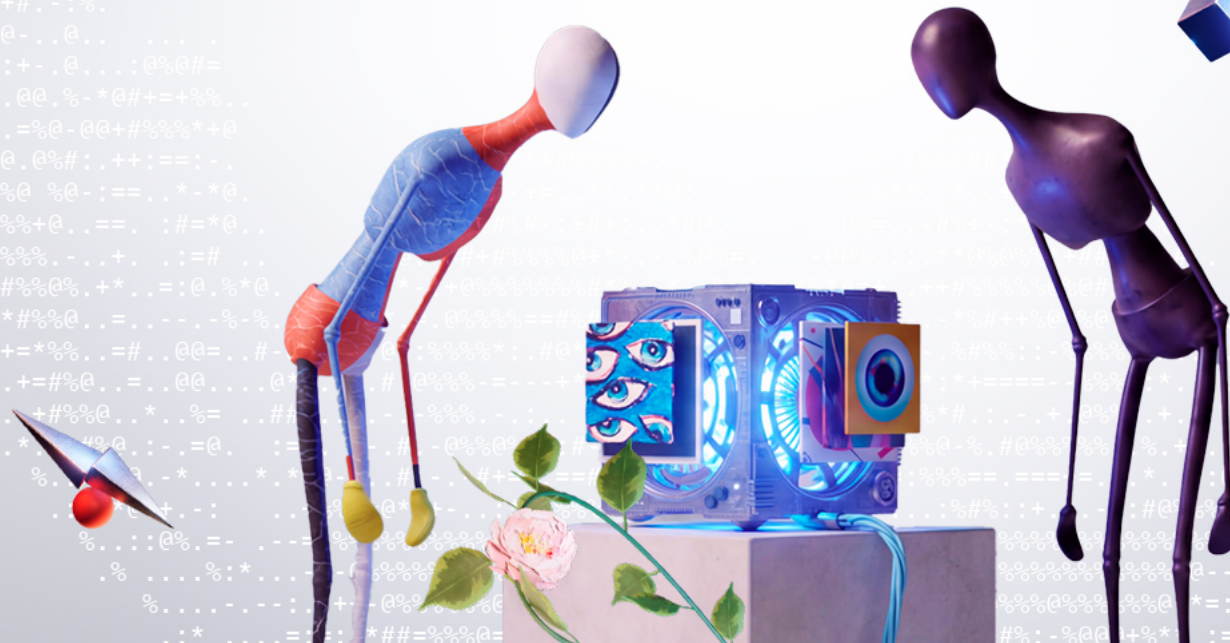


BitLocker: WinRE and TPM (cont.)

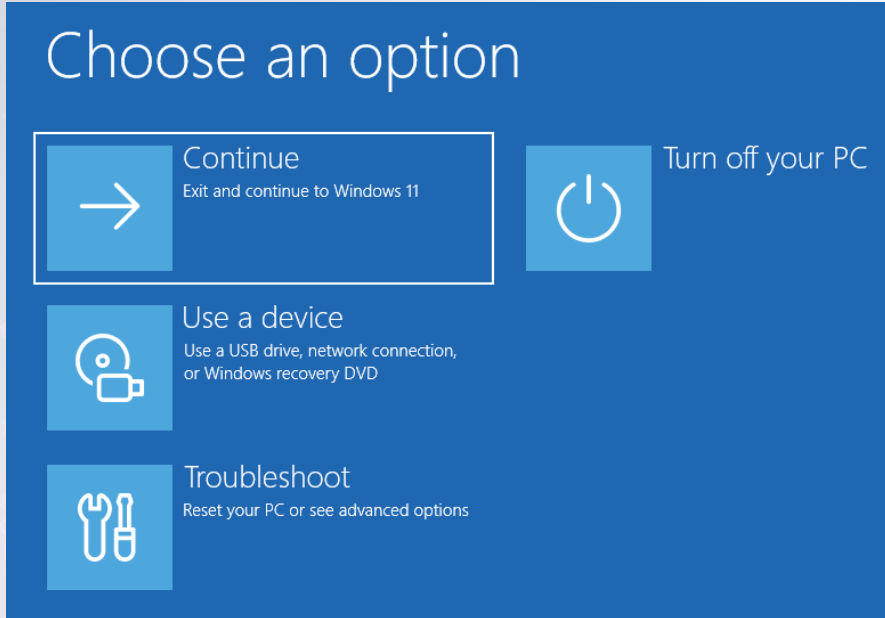
- The volume is locked after an unsafe action is requested

e.g., spawn the CMD shell

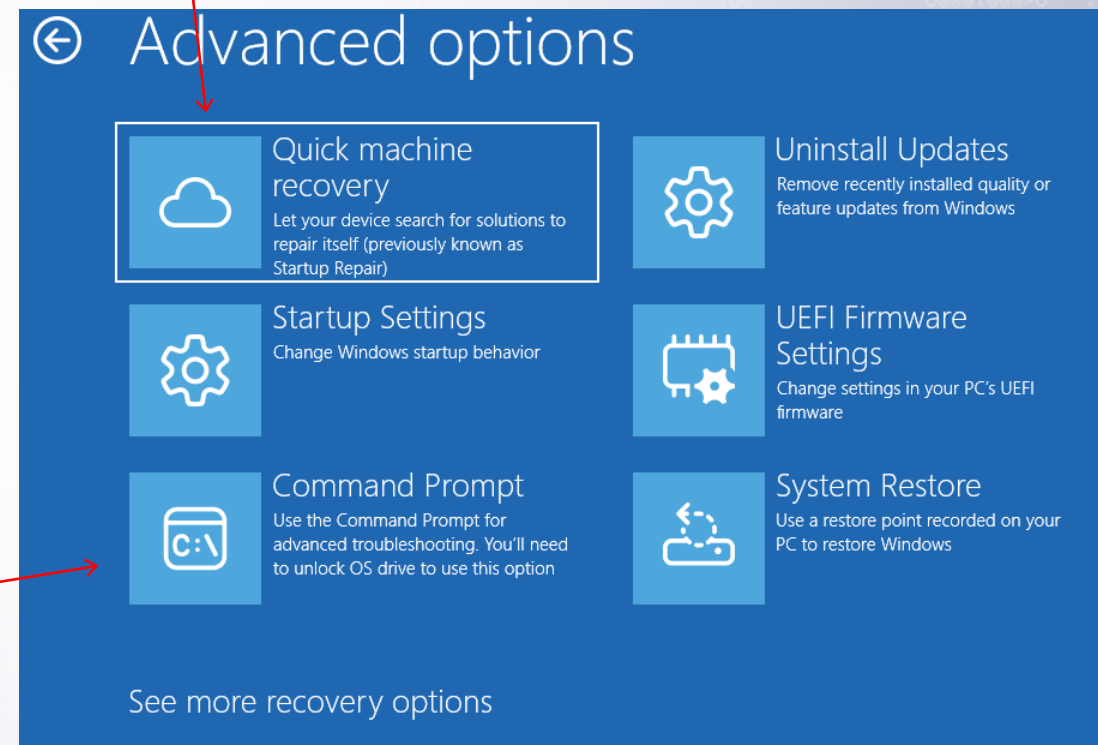
- The keyring is wiped too!



BitLocker: WinRE and TPM (cont.)



Safe



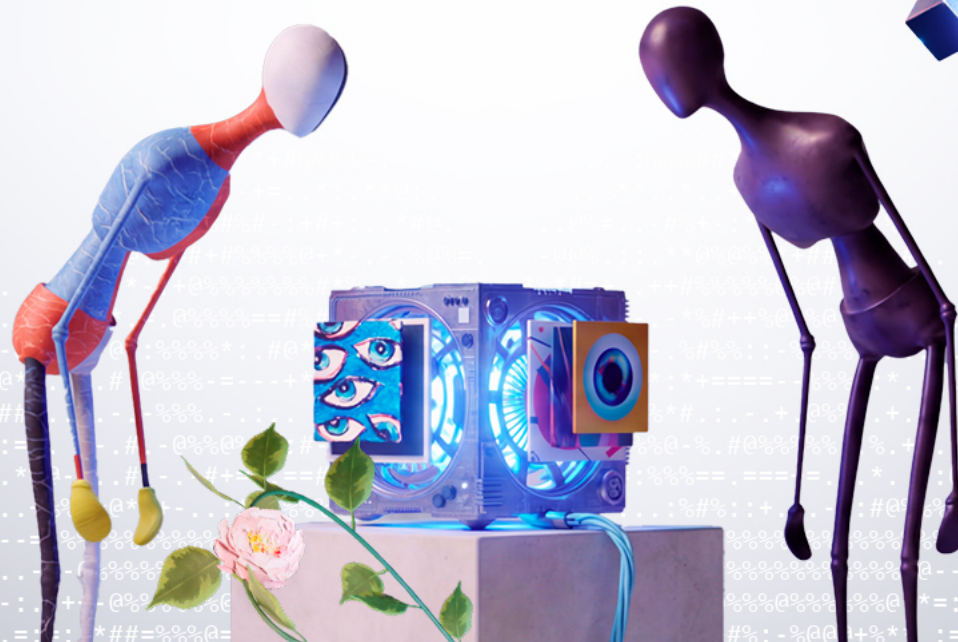
Not safe

BitLocker: Hardware Attacks & Protections

- Reboot (into another OS) and dump the RAM
 - The platform must implement the TCG Reset feature
- Sniff the channel between the host and the TPM
 - Doesn't work against firmware-based TPMs
- DMA attacks: drive-by and not
 - Protections must be enabled and measured (can't disable silently)
- Like a mobile device without *before first unlock*



CVEs!



CVE-2024-43513 (bitlockpick)

- WinRE fails to lock the system drive when spawning the CMD
- **Root cause:** the volume isn't locked when it's not mounted the loop iterates over mounted volumes only



CVE-2024-43513 (bitlockpick)

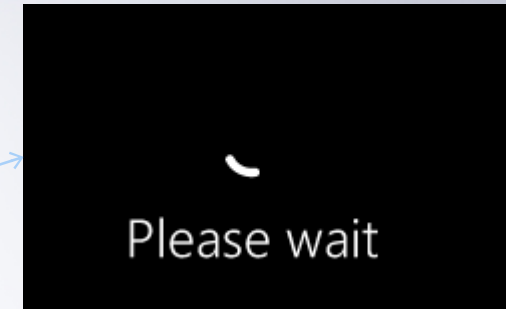
- WinRE fails to lock the system drive when spawning the CMD
- **Root cause:** the volume isn't locked when it's not mounted the loop iterates over mounted volumes only
- Re-format the EFI SP using the 2048-byte sector size files in this volume must be preserved
- Set the “noautomount” bit in the GPT for the encrypted partition
- Wipe the encrypted volume's start
- Boot, spawn CMD, the key is in-memory



CVE-2024-43513 (bitlockpick, cont.)

Another way:

- Clone the internal drive to any USB drive
- Wipe the partition table
 - both the MBR and GPT on the internal drive
- Boot from the USB drive into WinRE
- *When the spinning circle appears*, disconnect the drive
- Spawn CMD, re-connect the drive
 - `manage-bde -protectors -get C:`



CVE-2025-21202 (bitlockpick 2)

Root cause: this is an ugly logic bug

- Format a USB drive, encrypt
with a password or an external key
(no cleartext protector – important!)
- Remove the “noautomount” bit
if set for that external volume
- Attach, boot into WinRE
no unclean shutdowns (important!)
- Launch the Startup Repair process
but skip the recovery key prompt
- Spawn CMD
manage-bde -protectots -get E: (not a typo)

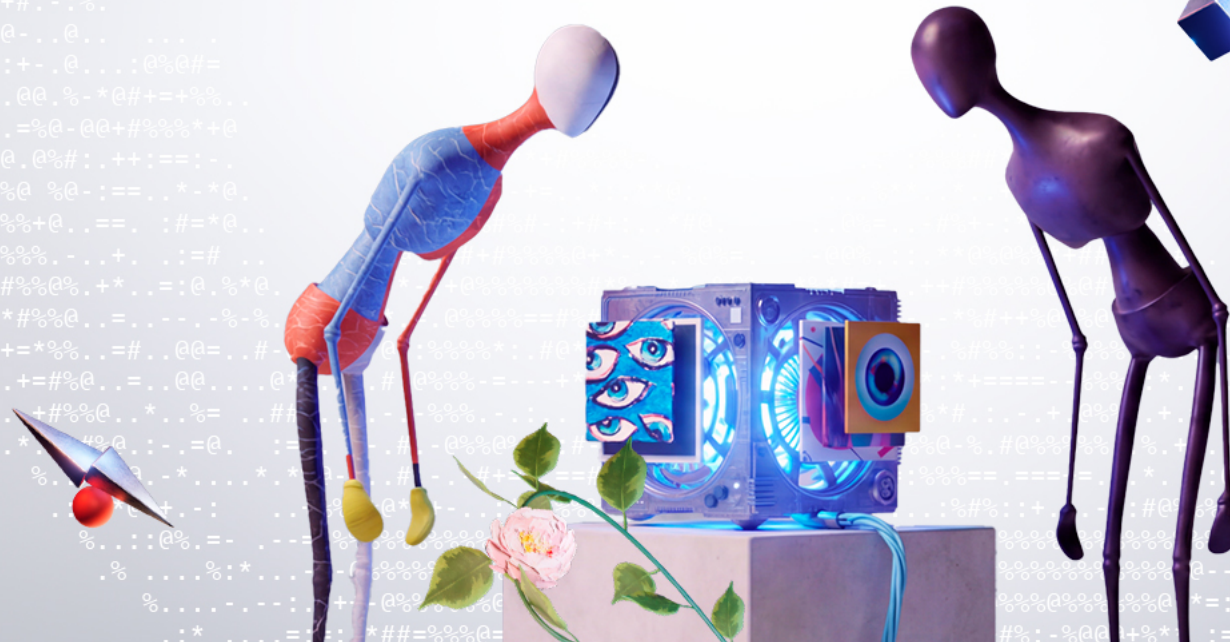


CVE-2025-21202 (bitlockpick 2, cont.)

- Many laptops ship with two encrypted volumes

C: (OS) and E: (data, set to auto-unlock after C:)

- They were vulnerable out of the box!



CVE-2026-20928

cold boot attack without cold and boot

Root cause: no memset() on free()

Locking the volume wipes the keyring, not other remnants

A reboot triggers the wipe
from the platform (firmware)

But no reboot required to
dump the RAM!

Address	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	ASCII
11931C9A0:	AA	00	00	00	08	22	64	04	01	00	00	00	01	00	64	04	"d.....d.
11931C9B0:	55	52	4C	04	30	20	64	04	20	00	00	00	76	6B	08	00	URL.0 d. ...vk..
11931C9C0:	04	00	00	80	00	00	00	00	04	00	00	00	01	00	65	00	e.
11931C9D0:	53	75	70	70	72	65	73	73	08	FF	FF	FF	76	6B	09	00	Suppress...vk..
11931C9E0:	04	00	00	80	00	00	00	00	04	00	00	00	01	00	3A	00	:
11931C9F0:	43	6F	6E	74	61	69	6E	65	72	00	35	00	64	00	34	00	Container.5.d.4.
11931CA00:	E0	FF	FF	FF	76	6B	07	00	04	00	00	80	00	00	00	00	vk.....
11931CA10:	04	00	00	00	01	00	39	00	49	6E	63	6C	75	64	65	00	9.Include.
11931CA20:	E0	FF	FF	FF	76	6B	08	00	04	00	00	80	00	00	00	00	vk.....
11931CA30:	04	00	00	00	01	00	38	00	53	75	70	70	72	65	73	73	8.Suppress
11931CA40:	E0	FF	FF	FF	76	6B	07	00	04	00	00	80	01	00	00	00	vk.....
11931CA50:	04	00	00	00	01	00	2A	00	44	65	66	61	75	6C	74	04	*.Default.
11931CA60:	08	00	00	00	28	A4	63	00	08	FF	FF	FF	76	6B	0C	00	(.c....vk..
11931CA70:	08	00	00	00	A0	B1	A8	00	03	00	00	00	01	00	5C	00	\.
11931CA80:	50	72	65	76	69	6F	75	73	48	77	69	64	73	00	5C	00	PreviousHwids.\.
11931CA90:	28	00	00	00	76	6B	07	00	04	00	00	80	01	00	00	00	(...vk.....
11931CAA0:	04	00	00	00	01	00	6E	00	44	65	66	61	75	6C	74	76	n.Defaultv
11931CAB0:	08	00	00	00	74	61	72	74	08	FF	FF	FF	76	6B	0C	00	tart...vk..
11931CAC0:	BC	00	00	00	58	98	63	00	01	00	00	00	01	00	74	00	X.c.....t.
11931CAD0:	73	65	63	72	65	74	5F	64	61	74	61	33	75	00	6D	00	secret_data3u.m.
11931CAE0:	D0	FF	FF	FF	76	6B	12	00	20	00	00	00	08	09	5E	04	vk.....^.
11931CAF0:	01	00	00	00	01	00	32	00	31	33	33	39	31	30	33	34	2.13391034
11931CB00:	31	37	33	36	30	32	35	32	38	35	64	00	00	00	49	64	1736025285d...Id
11931CB10:	F0	FF	FF	FF	6C	68	01	00	08	8B	5D	04	A3	E4	08	C7	lh....]
11931CB20:	40	FF	FF	FF	53	00	4F	00	46	00	54	00	57	00	41	00	@...S.O.F.T.W.A.
11931CB30:	52	00	45	00	5F	00	53	00	45	00	43	00	52	00	45	00	R.E._S.E.C.R.E.
11931CB40:	54	00	5F	00	44	00	41	00	54	00	41	00	5F	00	31	00	T._D.A.T.A._1.
11931CB50:	32	00	33	00	34	00	33	00	32	00	31	00	5F	00	53	00	2.3.4.3.2.1._S.
11931CB60:	45	00	43	00	52	00	45	00	54	00	5F	00	5F	00	53	00	E.C.R.E.T._.S.
11931CB70:	45	00	43	00	52	00	45	00	54	00	5F	00	44	00	41	00	E.C.R.E.T._.D.A.
11931CB80:	54	00	41	00	5F	00	31	00	32	00	33	00	34	00	33	00	T.A._1.2.3.4.3.
11931CB90:	32	00	31	00	5F	00	53	00	45	00	43	00	52	00	45	00	2.1._S.E.C.R.E.
11931CBA0:	54	00	5F	00	5F	00	53	00	45	00	43	00	52	00	45	00	T._.S.E.C.R.E.
11931CBB0:	54	00	5F	00	44	00	41	00	54	00	41	00	5F	00	31	00	T._D.A.T.A._1.
11931CBC0:	32	00	33	00	34	00	33	00	32	00	31	00	5F	00	53	00	2.3.4.3.2.1._S.
11931CBD0:	45	00	43	00	52	00	45	00	54	00	5F	00	5F	00	00	00	E.C.R.E.T._.



CVE-2026-20928

cold boot attack without cold and boot

Root cause: no memset() on free()

Locking the volume wipes the keyring, not other remnants

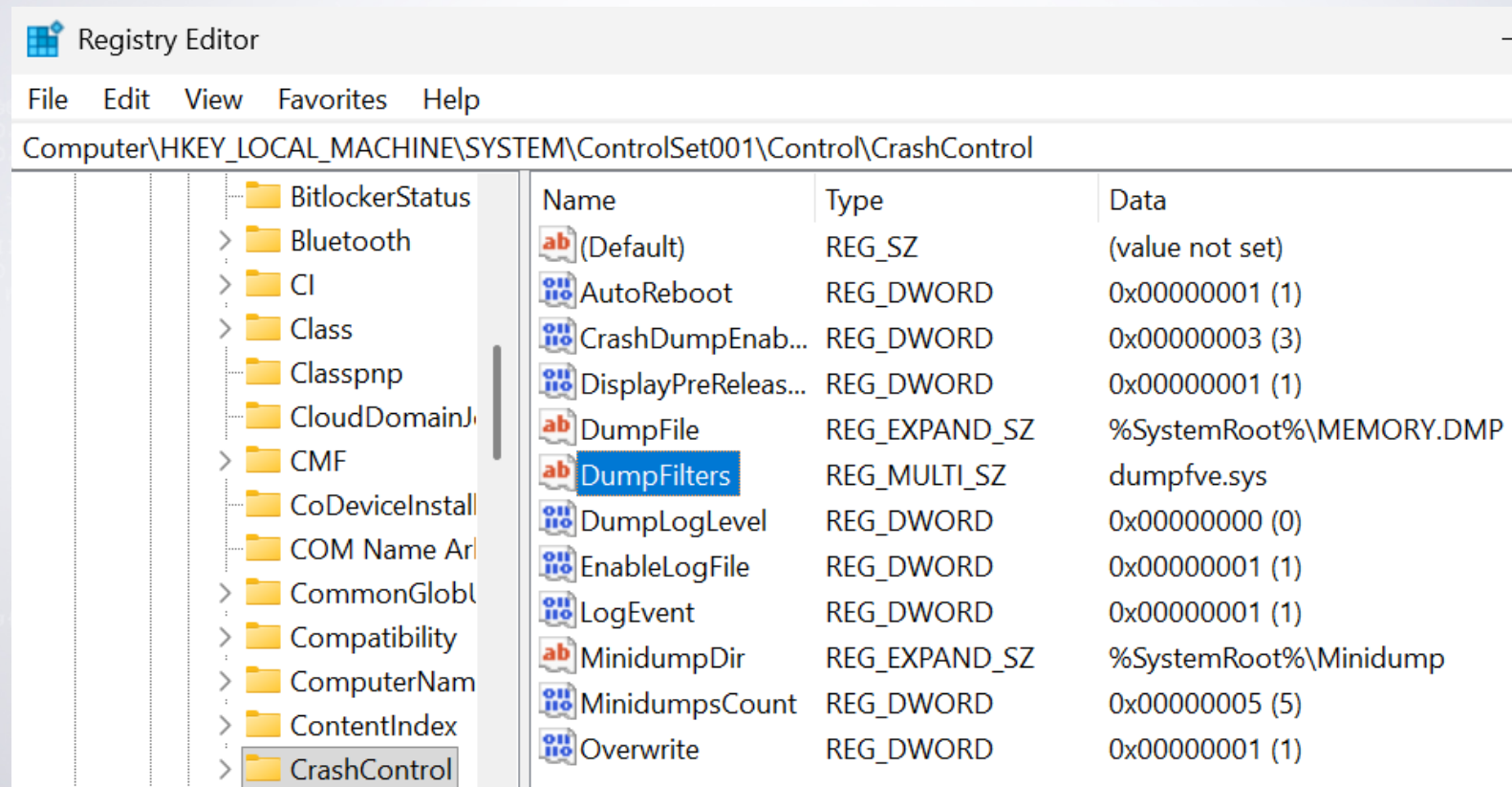
- Boot into WinRE
- Launch the Startup Repair process
 - encrypted data processed and freed, not wiped
- Spawn CMD
 - skip the recovery key prompt, only the key gets wiped
- Dump the RAM
 - e.g., using Belkasoft RAM Capturer
- Carve parts of \$MFT (NTFS) and registry hives
- Repeat



CVE-2025-21210: CrashXTS

Root cause: XTS traffic analysis, XTS randomization

Block the dumpfve.sys driver from being loaded, boot, then hibernate

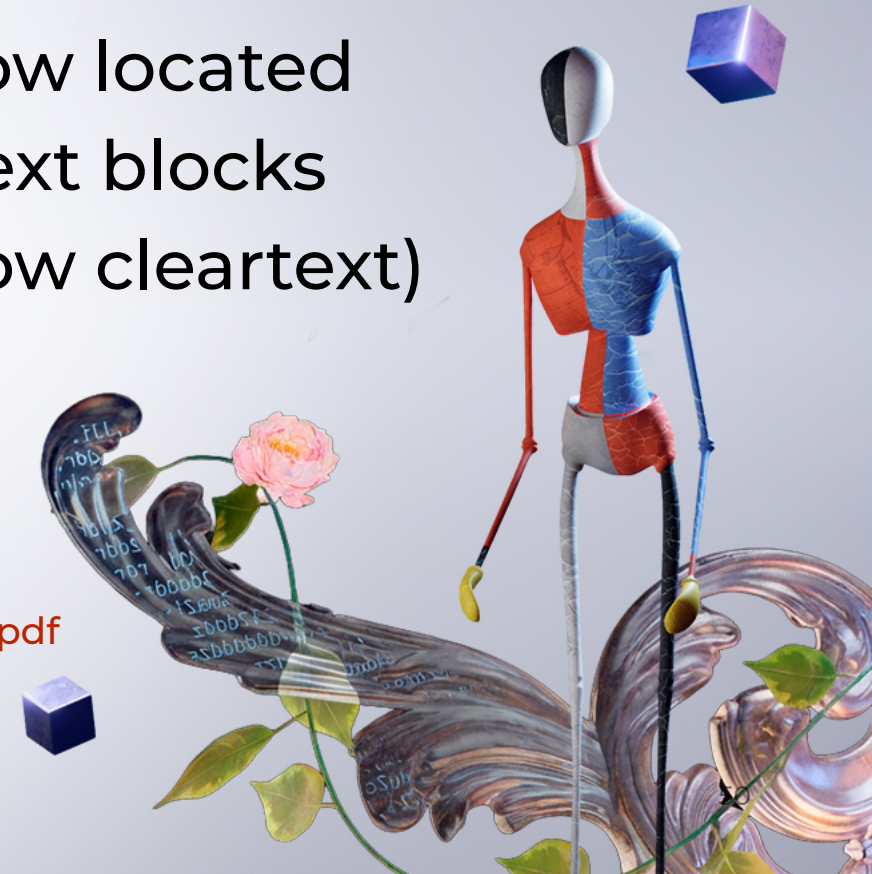


CVE-2025-21210: CrashXTS (cont.)

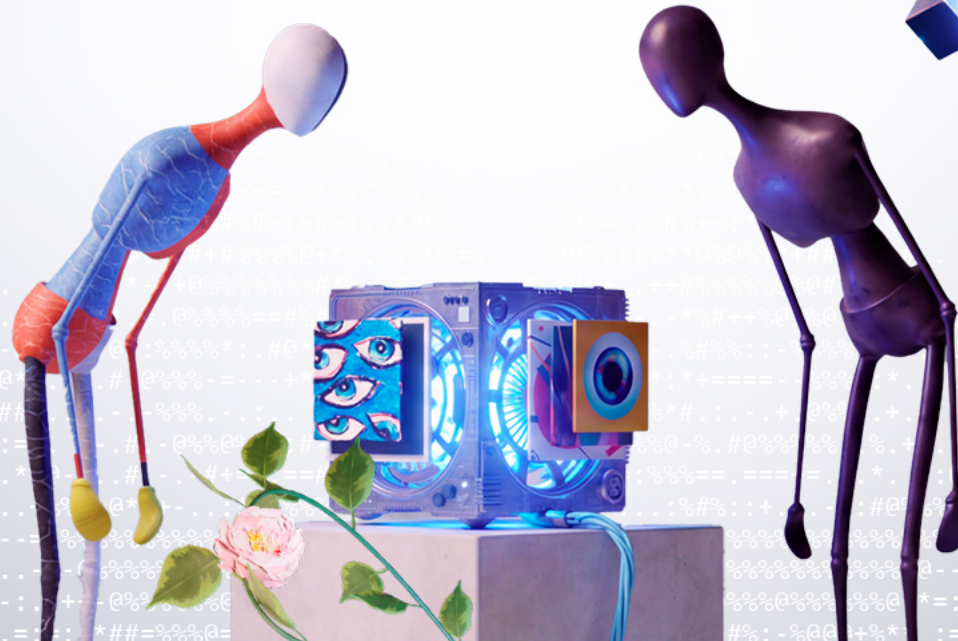
- Boot → Clean shutdown (1st state)
- Boot → Unclean shutdown (2nd state)
- Boot → BSOD (3rd state)
- Traffic analysis: the “CrashControl” key is now located
- Randomization: corrupt candidate ciphertext blocks
- Boot → Hibernate → pwned (hiberfil.sys is now cleartext)



<https://dfir.ru/wp-content/uploads/2025/01/crashxts.pdf>



Anything more like this?



Very Bad .reg from <Some Vendor> (2024, now fixed)

The screenshot shows a web browser displaying a support article titled "How to get a complete memory dump when Windows 10 crashes with a blue screen". The article is dated August 31, 2023, with ID 12402. It discusses a situation where Windows 10 crashes and a blue screen appears (BSOD - Blue Screen of Death). The article provides instructions on how to create a full memory dump and send it to the vendor for analysis. The instructions include: 1. Free up at least 25 GB of space on the system drive (most often, it's C:\). By default, the dump file is saved in the Windows folder on the system drive with the name MEMORY.DMP. 2. Download the CrashControl.zip archive and extract the files from it. 3. Run the CrashControl.reg file from the archive. 4. Wait until the file has been added to the registry and reboot the computer. 5. Reproduce the issue. 6. Go to the Windows folder on the system drive and make sure that the MEMORY.DMP file has been created. 7. Compress the file into an archive. For instructions, see this article. The article also mentions attaching the created dump file to a request to the vendor and uploading the archive to a cloud storage service.

Overlaid on the article is a Windows Registry Editor window titled "CrashControl.reg". The window shows the following registry values:

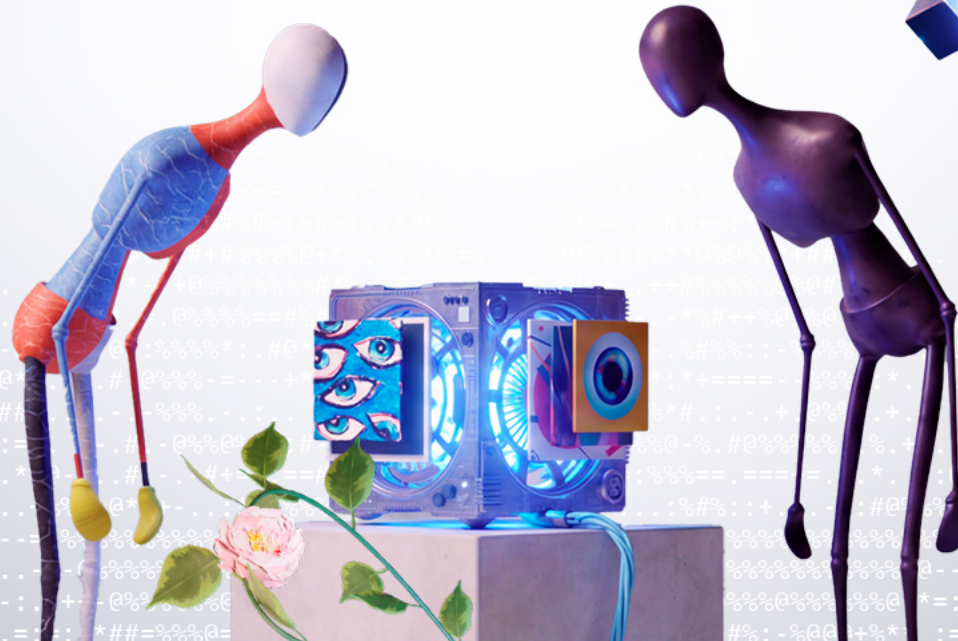
```
1 Windows Registry Editor Version 5.00
2
3 [HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\CrashControl]
4 "DumpFilters"=hex(7):64,00,75,00,6d,00,70,00,66,00,76,00,65,00,2e,00,73,00,79,\
5 00,73,00,00,00,00,00
6 "LogEvent"=dword:00000001
7 "MinidumpDir"=hex(2):25,00,53,00,79,00,73,00,74,00,65,00,6d,00,52,00,6f,00,6f,\
8 00,74,00,25,00,5c,00,4d,00,69,00,6e,00,69,00,64,00,75,00,6d,00,70,00,00,00
9 "Overwrite"=dword:00000001
10 "DumpFile"=hex(2):25,00,53,00,79,00,73,00,74,00,65,00,6d,00,52,00,6f,00,6f,\
11 74,00,25,00,5c,00,4d,00,45,00,4d,00,4f,00,52,00,59,00,2e,00,44,00,4d,00,50,\
12 00,00,00
13 "MinidumpsCount"=dword:00000032
14 "AutoReboot"=dword:00000001
15 "CrashDumpEnabled"=dword:00000001
16 "AlwaysKeepMemoryDump"=dword:00000001
```

A blue arrow points from the text "No 3rd-party drivers (VeraCrypt, BCVE, etc.)" to the "DumpFilters" value in the registry editor.

Below the registry editor is a file explorer window titled "CrashControl.zip". It shows the contents of the archive, which include the "CrashControl.reg" file.

No 3rd-party
drivers
(VeraCrypt,
BCVE, etc.)

Back to BitLocker!



CVE-2025-21214: Root File System “Spoofing”

- Security implications known since the 00's
- This bug is a feature from the 90's
- I discovered an attack against BitLocker in 2015-{01|02}
- It's constantly re-discovered in other products as well

QSB-021-2015 in Qubes OS

also: https://gitlab.tails.boum.org/tails/tails/-/work_items/7475#note_102471

and: some fixes in GRUB

(and many others, re-discovered approximately every year)



CVE-2025-21214 (cont.)

- BIOS drives:
0x80 (boot drive), 0x81
- OS drives (native access, Linux-style):
sda, sdb, sdc
- Where is the boot drive?
0x80 = sda? sdb? sdc? or none of them?
- Same for UEFI drives
- Operating systems try to guess...
... based on what was read from the boot drive
via the firmware (e.g., via INT 13h)
 - Disk signature, MBR checksum
 - Disk GUID

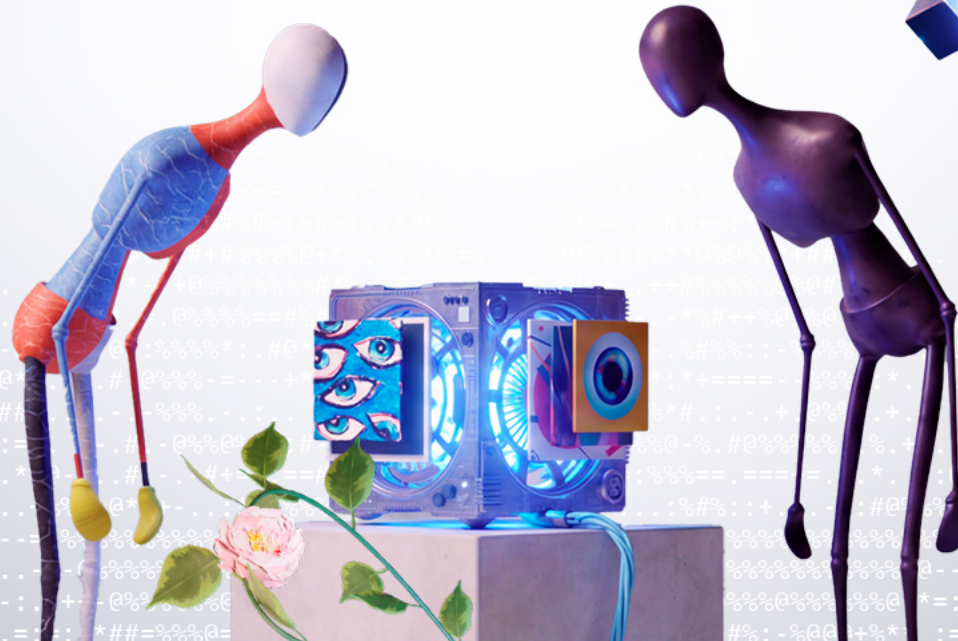


CVE-2025-21214 (cont.)

- Attach a 2nd drive with the same partition layout same GUIDs, offsets and sizes, but the system volume is unencrypted
- If EFI enumerates that drive after the real one and the OS enumerates it before the real one (hint: the former part depends on the UEFI firmware)
- Then the boot manager picks the real (encrypted) drive and unlocks it, loads the NT kernel and drivers
- Then the NT kernel picks a wrong (unencrypted) drive and boots into the unencrypted environment but the keyring is in-memory



LUKS (in GRUB)



CVE-2025-4382

- Seamless boot is implemented via GRUB in openSUSE
- Unlocking occurs in GRUB, the key is exposed to the Linux via a virtual file (no second unlock is required)

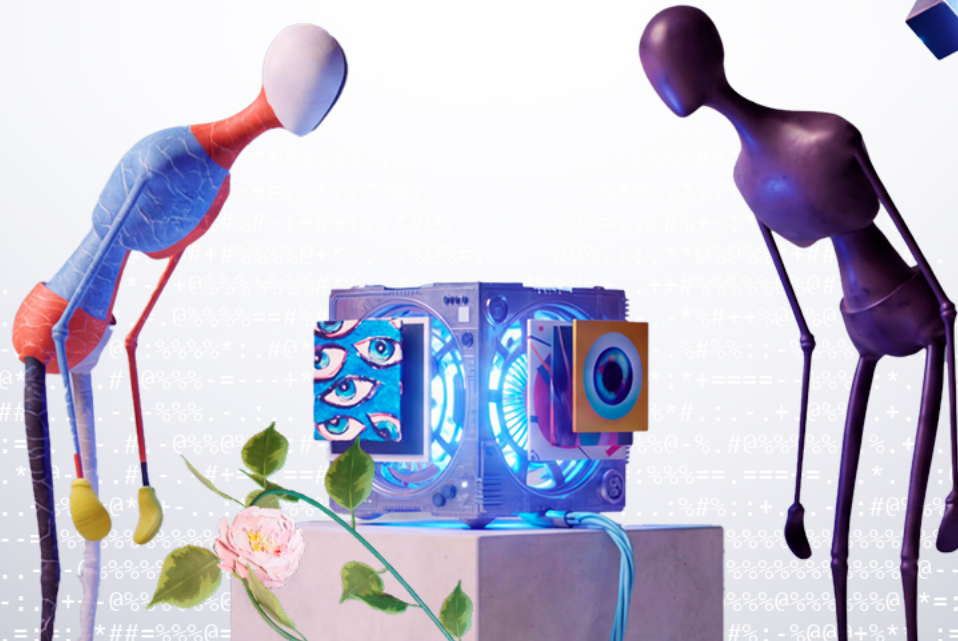


CVE-2025-4382

- Seamless boot is implemented via GRUB in openSUSE
- Unlocking occurs in GRUB, the key is exposed to the Linux via a virtual file (no second unlock is required)
- Corrupt the encrypted file system's superblock just wipe first megabytes of the partition
- Boot
- You get the GRUB rescue shell with full access
error condition → rescue shell
encryption keys are readable from the shell



Back to BitLocker



CVE-2025-21215: Capturing Your PIN

- Unusual starting point: the storage device
 - your HDD/SSD is backdoored
 - your ESX datastore is hacked (but not ESX itself)
 - your FC switch is hacked
 - admin-to-confidential-VM
- Otherwise, you need to write-protect a single sector somehow (easy for virtual disks in some clouds)
- Data-only, no code injection into the boot path
- Obtains the BitLocker PIN as keystrokes
 - before they get erased
 - but sometimes captures the main key (FVEK)

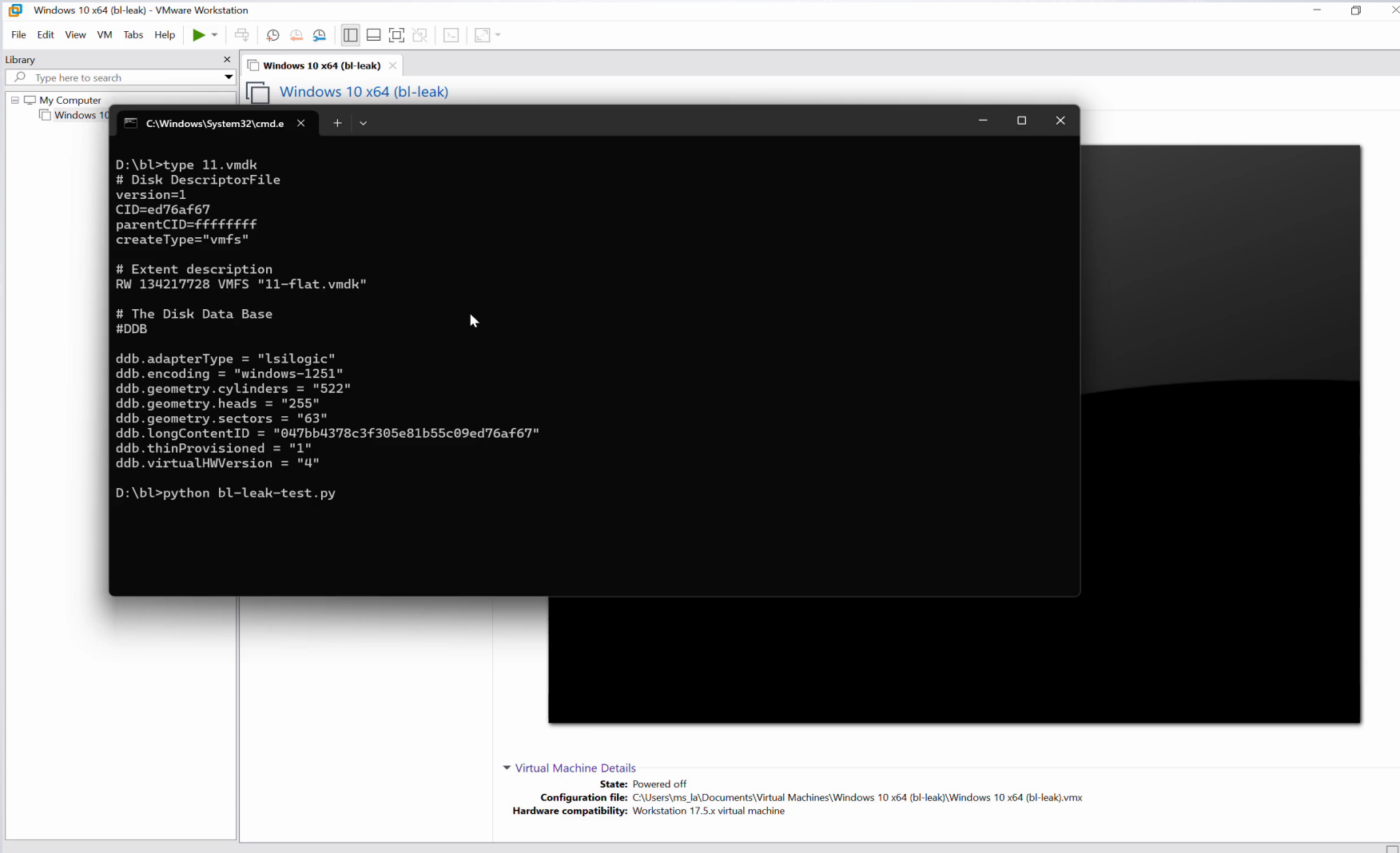


CVE-2025-21215 (cont.)

- The actual bug is the out-of-bounds read in the NTFS code `bootmgfw.NtfsReadWriteAttribute()`
 - no check that the resident data fits the allocation
- ~~It's just an OOB read! No way to expose the bytes over-read~~
- Writing the leaked buffer to the disk via an old-style registry transaction log file delivered into the BCD hive bytes



CVE-2025-21215 (demo)



The screenshot shows a VMware Workstation window titled "Windows 10 x64 (bl-leak) - VMware Workstation". The interface includes a menu bar (File, Edit, View, VM, Tabs, Help), a toolbar, and a Library pane on the left. The main window displays a Windows 10 desktop. A command prompt window is open, showing the following text:

```
D:\bl>type 11.vmdk
# Disk DescriptorFile
version=1
CID=ed76af67
parentCID=ffffffff
createType="vmfs"

# Extent description
RW 134217728 VMFS "11-flat.vmdk"

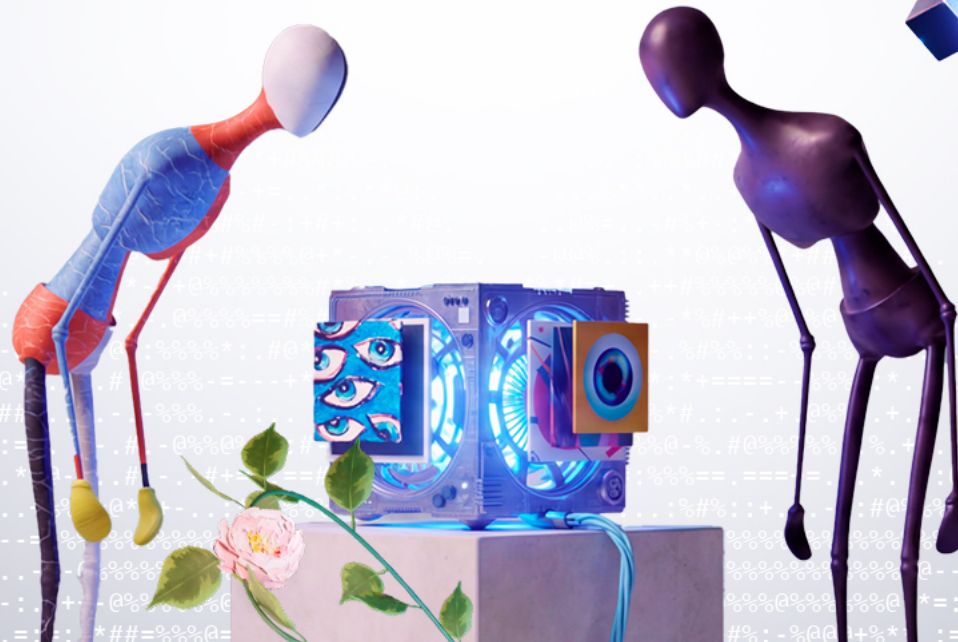
# The Disk Data Base
#DDB

ddb.adapterType = "lsilogic"
ddb.encoding = "windows-1251"
ddb.geometry.cylinders = "522"
ddb.geometry.heads = "255"
ddb.geometry.sectors = "63"
ddb.longContentID = "047bb4378c3f305e81b55c09ed76af67"
ddb.thinProvisioned = "1"
ddb.virtualHWVersion = "4"

D:\bl>python bl-leak-test.py
```

At the bottom of the VMware window, the "Virtual Machine Details" section is visible, showing the state as "Powered off", the configuration file path, and hardware compatibility information.

Our future



Wide-Block Ciphers & Modes

- Adiantum (cipher) and others
block size can be 4096 bytes
- See CVE-2025-4382
- Randomizing more than a megabyte can lead to a vulnerability



Authenticated Encryption

- Auth tags consume disk space
- Userspace isn't ready for read errors on auth failures
garbage (random data) vs. read()/readdir() error

Perl / perl5 ▾

Issues 2.2k Pull requests 169 Actions Projects Wiki Security and quality Insights

readdir does not report an eventual error #17907

Open

rdiez opened on Jun 28, 2020

readdir does not report an eventual error

The documentation from Perl's readdir does not state how you get an eventual error back.

This issue in Perl's readdir has been discussed in the past:

[Testing for readdir failure](#)

Linux documents that readdir() can fail, and how to get the error code. It involves setting errno to 0 before calling readdir(). Other operating systems may also fail this kind of directory scanning in other interesting ways.



Authenticated Encryption (cont.)

Maxim Suhanov 2025-06-11 15:32:51 UTC

[Description](#)

Hello.

Problem

When running the 'ls' command against an unreadable directory, its exit code is zero, and no read error is printed.

Example:

```
root@testhost:~ # mount | grep -F /mnt
/dev/dalpl1.eli on /mnt (ufs, local, soft-updates)
root@testhost:~ # ls /mnt
.snap  test  test2
root@testhost:~ # ls /mnt/test2
1.txt
root@testhost:~ # ls /mnt/test
root@testhost:~ # echo $?
0
root@testhost:~ # truss ls -l /mnt/test/ 2>&1 | tail
fstat(4,{ mode=drwxr-xr-x ,inode=98304,size=26624,blksize=32768 }) = 0 (0x0)
fchdir(0x4) = 0 (0x0)
getdirentries(4,0x5743bd843000,4096,0x5743bd840028) ERR#97 'Integrity check failed'
close(4) = 0 (0x0)
fchdir(0x3) = 0 (0x0)
fstat(1,{ mode=p----- ,inode=708,size=0,blksize=4096 }) = 0 (0x0)
total 0
write(1,"total 0\n",8) = 8 (0x8)
exit(0x0)
process exit, rval = 0
root@testhost:~ #
```

Compare this to the 'find' command (which reports the error properly):

```
root@testhost:~ # find /mnt/
/mnt/
/mnt/.snap
/mnt/test
find: /mnt/test: Integrity check failed
/mnt/test2
/mnt/test2/1.txt
root@testhost:~ # echo $?
1
root@testhost:~ #
```

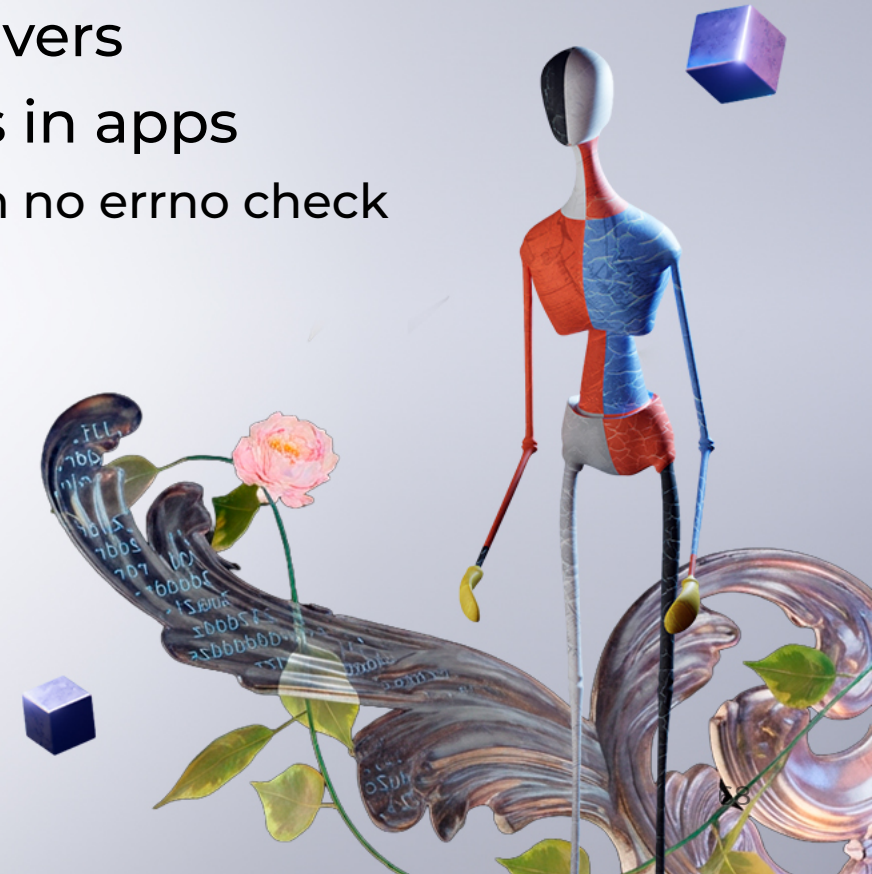
Root cause

The bug is in 'bin/ls/ls.c':

And many others!

- SMB clients
- File system drivers
- Config readers in apps

readline() with no errno check



**OFFZONE
2026**

**Thank you!
Questions?**



OFFZONE
2026

MS



@ERRNO_FAIL

Blog: dfir.ru

