

As requested, I have now demonstrated both vulnerabilities (BA2022 & BA2015) with practical exploitation scenarios and PoC evidence, which shows how an attacker could abuse these issues in a real-world context.

#### 📌 1. [BA2022] Weak Hash in Code Signing – SHA-1 Exploitation Demonstration

Although the original CyberGhostVPNSetup.exe appears signed and trusted, I have successfully created a fake installer (fakeCyberGhost.exe) using a self-signed certificate with SHA-1 hashing.

##### ✅ Technical Proof:

Used OpenSSL to generate a fake CA and SHA-1 certificate

Signed a fake binary using signtool.exe with that certificate

Windows still marked it as “Signed” (see sigcheck output)

Signature chain was accepted despite being forged with SHA-1

##### 🔥 Security Impact:

This shows that an attacker could:

Forge a malicious payload using a SHA-1-based certificate chain

Mimic the CyberGhost vendor name and product

Trick users into installing malware disguised as a trusted installer

This kind of issue is highly relevant in supply chain attacks and APT-level scenarios, especially in environments that do not enforce strict trust validation or SmartScreen policies.

```

C:\Users\Fatih Bülbul\Bug-Bounty\Sigcheck>sigcheck -i CyberGhostVPNSetup.exe

Sigcheck v2.90 - File version and signature viewer
Copyright (C) 2004-2022 Mark Russinovich
Sysinternals - www.sysinternals.com

C:\Users\Fatih B'lb'l\Bug-Bounty\Sigcheck\CyberGhostVPNSetup.exe:
Verified: Signed
Link date: 12:25 13.09.2023
Signing date: 12:25 13.09.2023
Catalog: C:\Users\Fatih B'lb'l\Bug-Bounty\Sigcheck\CyberGhostVPNSetup.exe
Signers:
  CyberGhost S.R.L.
    Cert Status: This certificate or one of the certificates in the certificate chain is not time valid.
    Valid Usage: Code Signing
    Cert Issuer: Sectigo Public Code Signing CA R36
    Serial Number: 10 80 11 D9 0D 01 0F 95 9E C9 66 41 E6 8F 91 8F
    Thumbprint: 7F5F5D77549F37157E49A1257CFD77C6AEC97001
    Algorithm: sha384RSA
    Valid from: 03:00 03.06.2022
    Valid to: 02:59 03.06.2024
  Sectigo Public Code Signing CA R36
    Cert Status: Valid
    Valid Usage: Code Signing
    Cert Issuer: Sectigo Public Code Signing Root R46
    Serial Number: 62 1D 6D 0C 52 01 9E 3B 90 79 15 20 89 21 1C 0A
    Thumbprint: 0BC5E76773D2E44FC9903D4DFE451553BBEC4A
    Algorithm: sha384RSA
    Valid from: 03:00 22.03.2021
    Valid to: 02:59 22.03.2036
  Sectigo Public Code Signing Root R46
    Cert Status: Valid
    Valid Usage: Code Signing
    Cert Issuer: AAA Certificate Services
    Serial Number: 48 FC 93 B4 60 55 94 8D 36 A7 C9 8A 89 D6 94 16
    Thumbprint: 329B78A5C9EBC2043242DE90CE1B7C6B1BA6C692
    Algorithm: sha384RSA
    Valid from: 03:00 25.05.2021
    Valid to: 02:59 01.01.2029
  Sectigo (AAA)
    Cert Status: Valid
    Valid Usage: Client Auth, Code Signing, EFS, Email Protection, IPSEC Tunnel, IPSEC User, Server Auth, Timestamp Signing
    Cert Issuer: AAA Certificate Services
    Serial Number: 01
    Thumbprint: 01E723406D17D68FD92564C2F1F1601764D8E349
    Algorithm: sha1RSA
    Valid from: 03:00 01.01.2004
    Valid to: 02:59 01.01.2029
Counter Signers:

```

```

C:\Users\Fatih Bülbul\Bug-Bounty>copy C:\Windows\System32\notepad.exe fakeCyberGhost.exe
1 file(s) copied.

```

```

C:\Users\Fatih Bülbul\Bug-Bounty>C:\Program Files (x86)\OpenSSL-Win32\bin\openssl.exe req -x509 -newkey rsa:2048 -keyout fake.key -out fake.crt -sha1 -days 365 -nodes -subj "/CN=Fake CyberGhost CA"
Generating a RSA private key
.....+++++
.....+++++
writing new private key to 'fake.key'
-----

C:\Users\Fatih Bülbul\Bug-Bounty>C:\Program Files (x86)\OpenSSL-Win32\bin\openssl.exe pkcs12 -export -out fake.pfx -inkey fake.key -in fake.crt -passout pass:1234

C:\Users\Fatih Bülbul\Bug-Bounty>dir
Volume in drive C is Windows
Volume Serial Number is 3A74-5E28

Directory of C:\Users\Fatih Bülbul\Bug-Bounty

20.05.2025 10:56 <DIR> .
20.05.2025 10:14 <DIR> ..
20.05.2025 10:55 1.158 fake.crt
20.05.2025 10:55 1.736 fake.key
20.05.2025 10:56 2.397 fake.pfx
13.05.2025 23:30 360.448 fakeCyberGhost.exe
20.05.2025 10:16 <DIR> Sigcheck
20.05.2025 10:14 689.079 Sigcheck.zip
20.05.2025 10:45 5.755.984 Win64OpenSSL_Light-3_0_0.exe
6 File(s) 6.801.802 bytes
3 Dir(s) 118.580.809.728 bytes free

C:\Users\Fatih Bülbul\Bug-Bounty>

```

```

C:\Users\Fatih Bülül\Bug-Bounty>"C:\Program Files (x86)\Windows Kits\10\bin\10.0.26100.0\x64\signtool.exe" sign /f fake.pfx /fd SHA1 /p 1234 fakeCyberGhost.exe
Done Adding Additional Store
Successfully signed: fakeCyberGhost.exe

C:\Users\Fatih Bülül\Bug-Bounty>Sigcheck\sigcheck64.exe -i fakeCyberGhost.exe

Sigcheck v2.90 - File version and signature viewer
Copyright (C) 2004-2022 Mark Russinovich
Sysinternals - www.sysinternals.com

C:\Users\Fatih Bülül\Bug-Bounty>fakeCyberGhost.exe:
Verified:      Bir sertifika zinciri do?ru olarak i?lendi ancak g'ven sa?lay?c?s? taraf?ndan g'venilmeyen bir k+k sertifikas?nda sona erdi.
Link date:     18:14 06.04.1909
Signing date:  11:19 20.05.2025
Catalog:      C:\Users\Fatih Bülül\Bug-Bounty\fakeCyberGhost.exe
Signer:
  Fake CyberGhost CA
  Cert Status:  The certificate or certificate chain is based on an untrusted root.
  Valid Usage:  All
  Cert Issuer:  Fake CyberGhost CA
  Serial Number: 78 F1 3F 3C 25 4F F2 E6 04 4C 41 0D 76 C6 75 DB C2 22 91 B8
  Thumbprint:    E5AF858BD55AA829C46CEFD6C13CD514125FB8C4
  Algorithm:    sha1RSA
  Valid from:    10:55 20.05.2025
  Valid to:      10:55 20.05.2026
  Company:      Microsoft Corporation
  Description:   Notepad
  Product:       Microsoft® Windows® Operating System
  Prod version:  10.0.22621.5262
  File version:  10.0.22621.5262 (WinBuild.160101.0800)
  MachineType:  64-bit

C:\Users\Fatih Bülül\Bug-Bounty>

```

## ✦ 2. [BA2015] Missing High Entropy ASLR – Predictable Memory Layout

I have used WinDbg to run CyberGhostVPNSetup.exe multiple times and observed that the binary consistently loads into low-memory base addresses, such as:

0x00C90000

0x00630000

On 64-bit systems, properly compiled binaries should load into high-memory randomized regions (e.g., 0x00007FF6xxxx0000) when High Entropy ASLR is enabled.

### ✓ Technical Proof:

WinDbg output shows low, non-randomized base addresses

Confirms that /HIGHENTROPYVA is not in use

Therefore, attackers can predict memory layout

Increases success of ROP (Return-Oriented Programming) and memory corruption exploits

C:\Users\Fatih Bülbul\Bug-Bounty\Sigcheck\CyberGhostVPNSetup.exe - WinDbg 1.2504.15001.0 (Administrator)

Command

```

----> Repository : LocalInstalled, Enabled: true, Packages count: 44

Microsoft (R) Windows Debugger Version 10.0.27829.1001 AMD64
Copyright (c) Microsoft Corporation. All rights reserved.

CommandLine: C:\Users\Fatih Bülbul\Bug-Bounty\Sigcheck\CyberGhostVPNSetup.exe

***** Path validation summary *****
Response      Time (ms)      Location
Deferred
Symbol search path is: srv*
Executable search path is:

+-----+
| This target supports Hardware-enforced Stack Protection. A HW based |
| "Shadow Stack" may be available to assist in debugging and analysis. |
| See aka.ms/userhsp for more info. |
+-----+

dps @ssp

ModLoad: 00000000`00c90000 00000000`00cb2000 WebBootstrapper.exe
ModLoad: 00007ffb`e2e30000 00007ffb`e3047000 ntdll.dll
ModLoad: 00007ffb`d4980000 00007ffb`d49eb000 C:\WINDOWS\SYSTEM32\MSCOREE.DLL
ModLoad: 00007ffb`e1ce0000 00007ffb`e1da4000 C:\WINDOWS\System32\KERNEL32.dll
ModLoad: 00007ffb`e0190000 00007ffb`e0563000 C:\WINDOWS\System32\KERNELBASE.dll
ModLoad: 00007ffb`dc0a0000 00007ffb`dc137000 C:\WINDOWS\SYSTEM32\apphelp.dll
(4adc.1360): Break instruction exception - code 80000003 (first chance)
ntdll!LdrpDoDebuggerBreak+0x30:
00007ffb`e2f0cb04 cc int 3

0:000>

```

| Name   | Value | Type | Location |
|--------|-------|------|----------|
| Locals |       |      |          |

| TID    | Index | Thread            |
|--------|-------|-------------------|
| 0x1360 | 0x0   | 00400000`0002b89e |

C:\Users\Fatih Bülbul\Bug-Bounty\Sigcheck\CyberGhostVPNSetup.exe - WinDbg 1.2504.15001.0 (Administrator)

Command

```

***** Path validation summary *****
Response      Time (ms)      Location
Deferred
Symbol search path is: srv*
Executable search path is:

+-----+
| This target supports Hardware-enforced Stack Protection. A HW based |
| "Shadow Stack" may be available to assist in debugging and analysis. |
| See aka.ms/userhsp for more info. |
+-----+

dps @ssp

ModLoad: 00000000`00630000 00000000`00652000 WebBootstrapper.exe
ModLoad: 00007ffb`e2e30000 00007ffb`e3047000 ntdll.dll
ModLoad: 00007ffb`d4980000 00007ffb`d49eb000 C:\WINDOWS\SYSTEM32\MSCOREE.DLL
ModLoad: 00007ffb`e1ce0000 00007ffb`e1da4000 C:\WINDOWS\System32\KERNEL32.dll
ModLoad: 00007ffb`e0190000 00007ffb`e0563000 C:\WINDOWS\System32\KERNELBASE.dll
ModLoad: 00007ffb`dc0a0000 00007ffb`dc137000 C:\WINDOWS\SYSTEM32\apphelp.dll
(1bb4.5324): Break instruction exception - code 80000003 (first chance)
ntdll!LdrpDoDebuggerBreak+0x30:
00007ffb`e2f0cb04 cc int 3

0:000> g
ModLoad: 00007ffb`c9f80000 00007ffb`c9a4b000 C:\WINDOWS\system32\TmUhl\20019\AddOn\8.55.0.1350\TmUhlEvt64.dll
ModLoad: 00007ffb`e0bc0000 00007ffb`e0bc8000 C:\WINDOWS\System32\PSAPI.DLL
ModLoad: 00007ffb`e0ac0000 00007ffb`e0b71000 C:\WINDOWS\System32\ADVAPI32.dll
ModLoad: 00007ffb`e0fa0000 00007ffb`e1047000 C:\WINDOWS\System32\msvcrt.dll
ModLoad: 00007ffb`e1100000 00007ffb`e11a7000 C:\WINDOWS\System32\sechost.dll
ModLoad: 00007ffb`e0730000 00007ffb`e0758000 C:\WINDOWS\System32\bcrypt.dll
ModLoad: 00007ffb`c23f0000 00007ffb`c2464000 C:\WINDOWS\System32\bcrypt.dll

```

## ★ Combined Business Risk

These two weaknesses, when combined, create a dangerous scenario:

Trust Bypass: The SHA-1 signature flaw allows an attacker to sign a fake binary that Windows accepts as “Signed”

Exploit Reliability: The predictable memory layout increases exploitability of memory-based vulnerabilities

Supply Chain Threat: End users are at risk of installing malware that appears legitimate

#### Supporting Evidence

Attached screenshots of:

SHA-1 forged certificate creation and signing

Sigcheck verification (shows “Signed” despite fake cert)

WinDbg memory base address outputs proving ASLR weakness

Each step has been performed on a clean Windows environment using official tools (BinSkim, WinDbg, signtool, OpenSSL)

#### Conclusion

This is not just a theoretical misconfiguration — it is a practical attack vector that shows how a motivated adversary could bypass trust and memory protections. The attached PoCs directly demonstrate real-world exploitability.