Source and Executable

Mike Lai, mikelai@microsoft.com
Microsoft Corp

```
main()
{
        cout <<
"Hello World\n";

        return 0;
}
```

HelloWorld.cpp

Human readable
C++ source code
file


```
4D 5A 90 00 03 00 00 00  04 00 00 00 FF FF 00 00
B8 00 00 00 00 00 00 00  40 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 00 00 F0 00 00 00
0E 1F BA 0E 00 B4 09 CD  21 B8 01 4C CD 21 54 68
69 73 20 70 72 6F 67 72  61 6D 20 63 61 6E 6E 6F
74 20 62 65 20 72 75 6E  20 69 6E 20 44 4F 53 20
6D 6F 64 65 2E 0D 0D 0A  24 00 00 00 00 00 00 00
48 B6 98 3C 0C D7 F6 6F  0C D7 F6 6F 0C D7 F6 6F
84 B0 F3 6E 14 D7 F6 6F  84 B0 F2 6E 07 D7 F6 6F
84 B0 F5 6E 0E D7 F6 6F  84 B0 F7 6E 09 D7 F6 6F
69 B1 F7 6E 08 D7 F6 6F  0C D7 F7 6F 5E D7 F6 6F
0D BA F3 6E 0E D7 F6 6F  0D BA 09 6F 0D D7 F6 6F
0D BA F4 6E 0D D7 F6 6F  52 69 63 68 0C D7 F6 6F
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
```

HelloWorld.exe

Machine readable
(binary) executable
file

Customer Question: Was the executable running in my machine really built from the same source code files that I or your 3rd party evaluators / auditors have reviewed / analyzed ?

Microsoft Response: Would you feel more comfortable if there are additional artifacts to resolve the Customer Question?

Is that really the source code for this software?

- A generic issue of software -- including Open Source Software
  - https://blogs.kde.org/2013/06/19/really-source-code-software
  - https://blog.torproject.org/blog/deterministic-builds-part-one-cyberwar-and-global-compromise
  - https://wiki.debian.org/ReproducibleBuilds
  - https://reproducible-builds.org/
  - Video: https://www.sfscon.it/talks/you-think-youre-not-a-target-a-tale-of-three-developers/
  - Video: Hardware implants in the supply-chain

https://media.ccc.de/v/35c3-9597-modchips_of_the_state#t=106

The source / executable correspondence question is asked more often than we think

equivalence can be demonstrated at scale. The NCSC has advised the Oversight Board that the priority should be to rectify these underlying flaws as part of Huawei's transformation plan. Unless and until this is done it is not possible to be confident that the source code examined by HCSEC is precisely that used to build the binaries running in the UK networks.

https://www.gov.uk/government/publications/huawei-cyber-security-evaluation-centre-oversight-board-annual-report-2019

Our approach

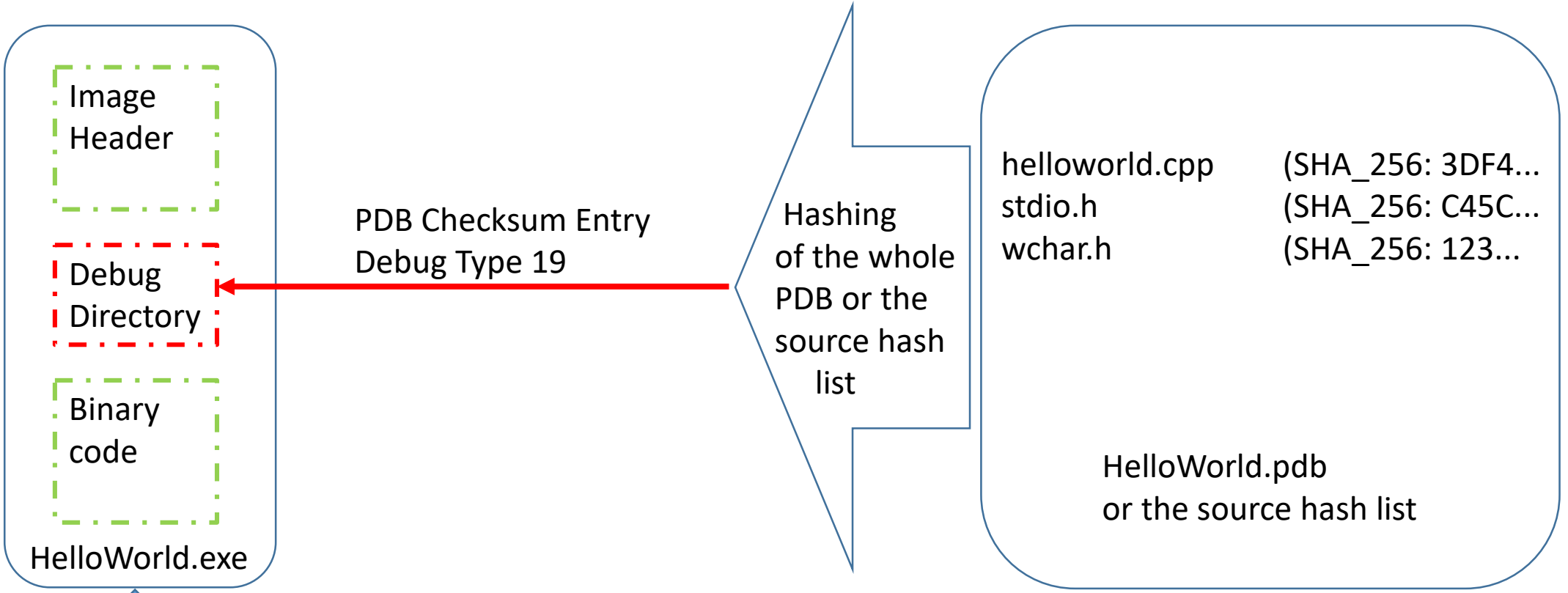
Based on the artifacts being generated by compiler/linker toolchains

Compiler Generated Artifacts

- Compiler reads source code files
- Compiler outputs a pair of files:
  1. Executable (exe, dll, sys, etc) containing the machine instructions
  2. Program Database (PDB) storing the debugging information
- Compiler computes strong hash values of source files
- Complier stores the hash values and the path names of source files in the PDB
- U computes the strong hash of the PDB or list of src hash values
- U stores the PDB or src hash list hash in a dir of the executable

❑ See https://msdn.microsoft.com/en-us/magazine/mt795185

Artifact Interrelationship (following ECMA-335)



On top of a stronger binding between the exe and PDB pair, your digital signing of the exe enables the integrity and authenticity of the PDB (which includes the source file hashes) as well as the exe (which contains the actual compiled binary code)

See https://bit.ly/2EcdPEp for the detail of PDB checksum in Debug Directory

Confirm that a source file was used during the compilation of an executable

- You compute the hash value of the source file in question (certutil –hashfile from system32)
- You discover the PDB file pairing with the executable in question (symchk from MSDN)
- You verify that the hash value that you just computed matches the hash value with the same source file name in the PDB that you just discovered
- See KB 3195907 (http://bit.ly/2gNthQk)
- Exactly what WinDbg does before using the source file

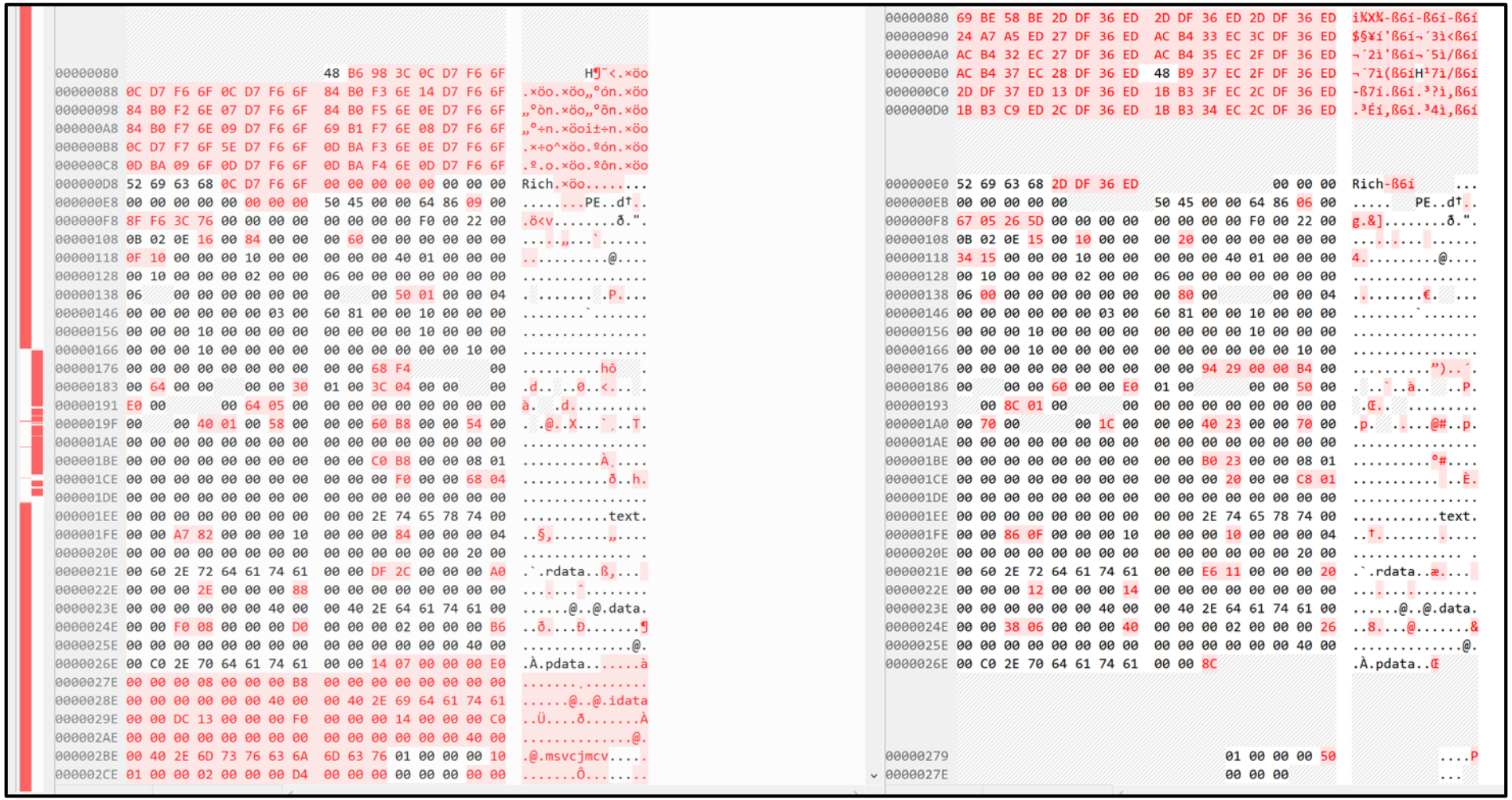
Still not convinced ?

Customer request: I want to recompile your sources to reproduce the executable files myself and to compare the reproduced with the original, as in the case of OSS

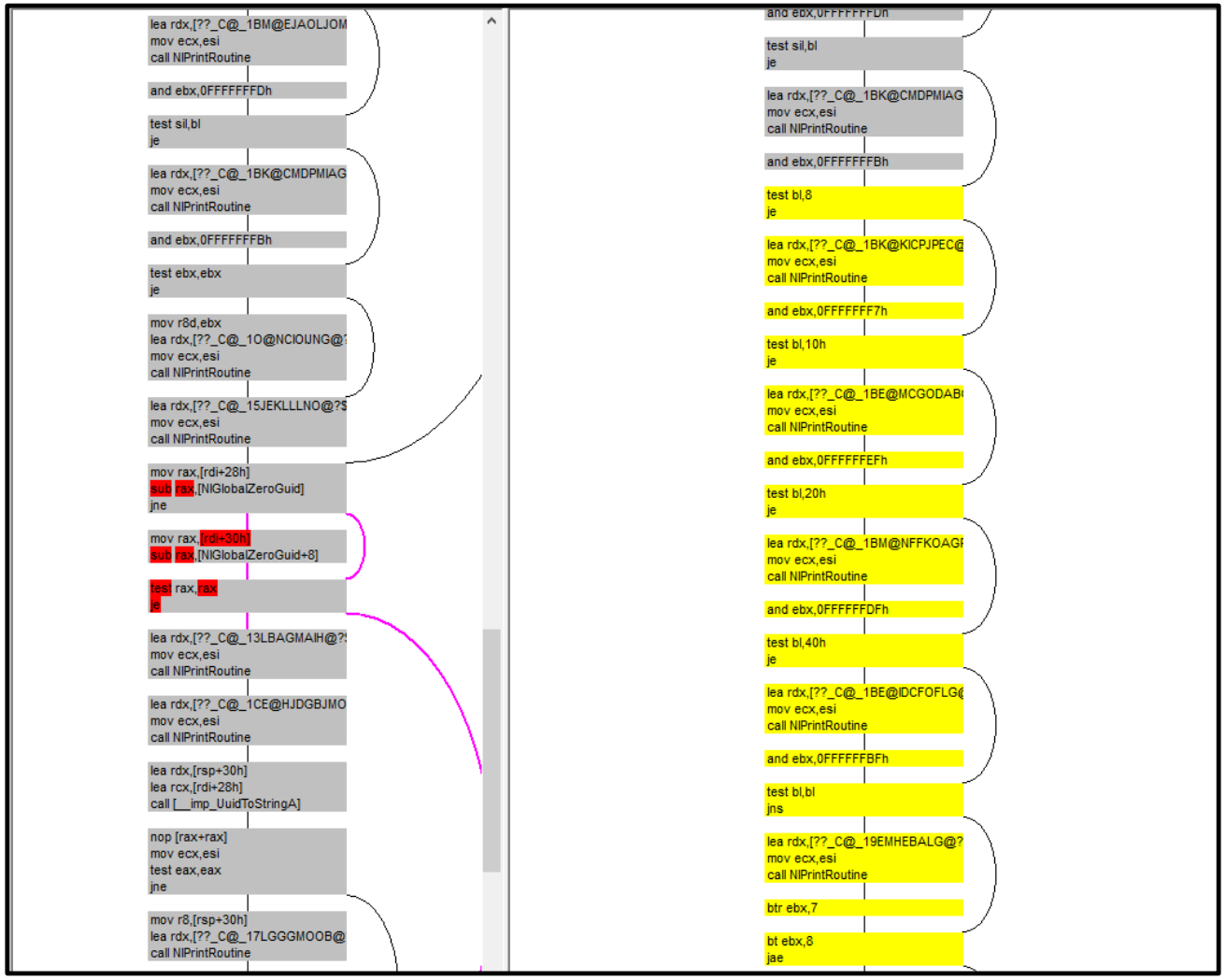
Microsoft Response: Please stop by one of our Transparency Centers

See https://www.microsoft.com/en-us/securityengineering/gsp for details

Not as easy as one thought (1) – hex compare



Not as easy as one thought (2) – machine instruction compare



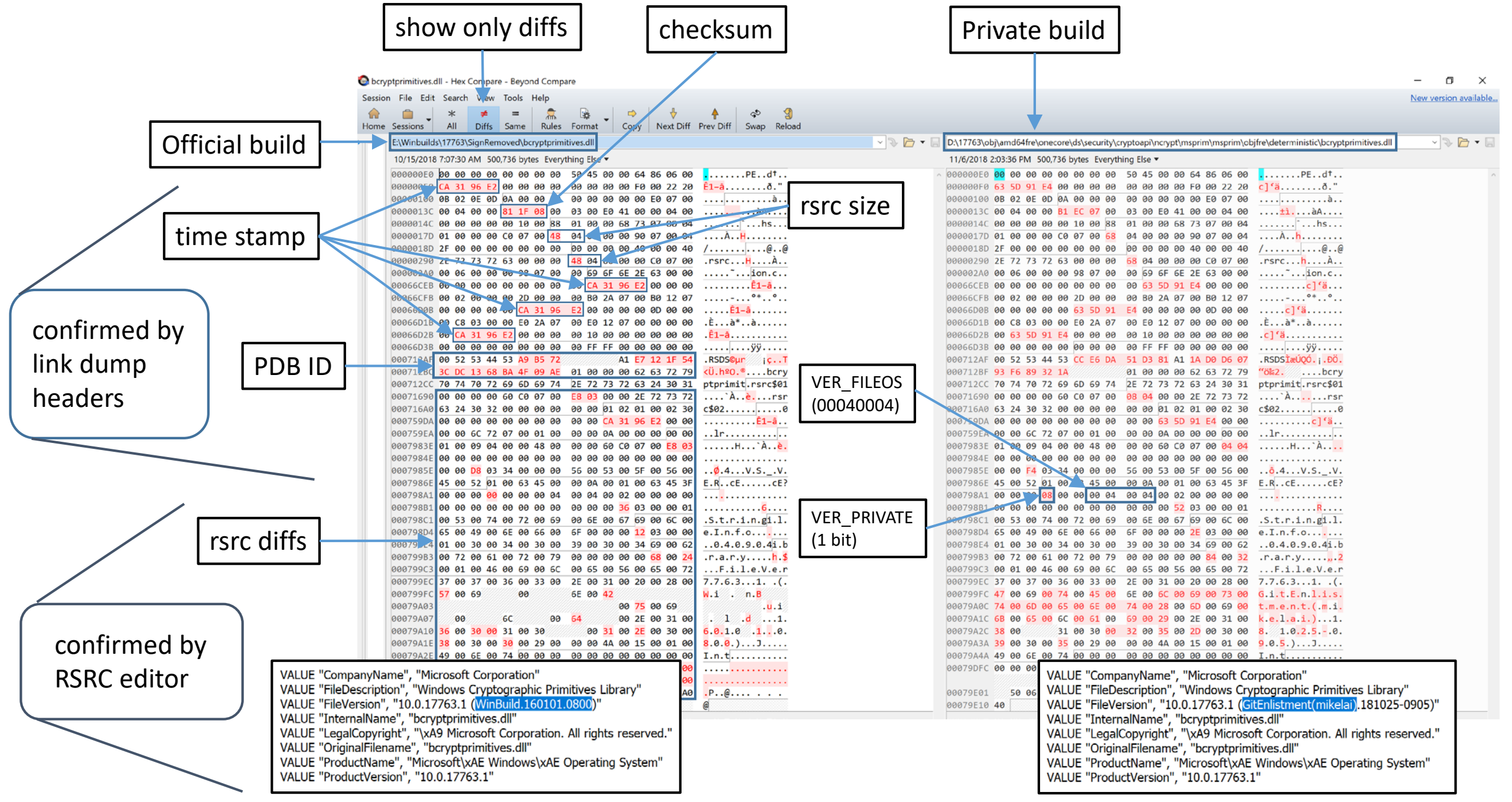
Need special support from compiler/linker toolchains

- 1980's Apollo DSEE (Domain Software Engineering Environment) reproduced an executable in a bit-by-bit identical manner
- Visual Studio 2019 C++ compiler/linker supports "/experimental:deterministic" and "/pathmap:<SOURCE>=<DEST>" flags
- Visual Studio 2019 CSC compiler supports "-deterministic" and "-pathmap:<SOURCE>=<DEST>" flags
- LLVM Clang toolchain, GCC toolchain -- see https://reproducible-builds.org/ for information

How do I discover that a Windows 10 based executable file is reproducible?

Use “link /dump /headers” against the executable file in question to inspect the Debug Directories in its image header

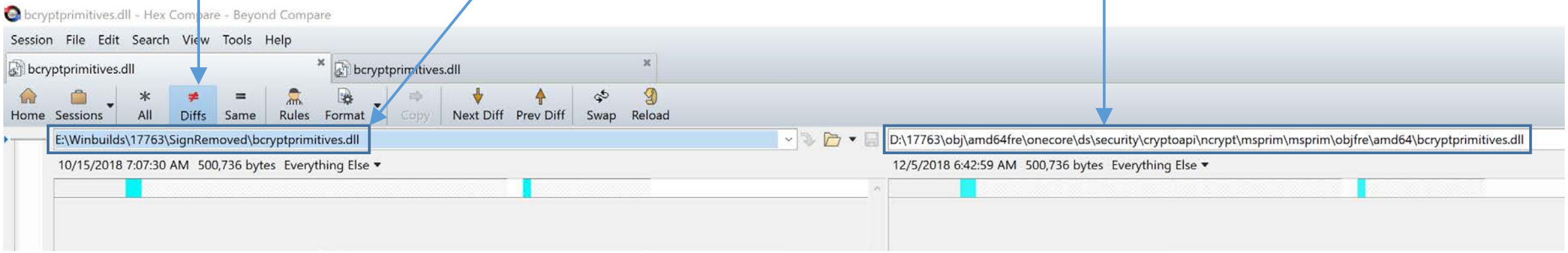
```
Debug Directories

    Time Type        Size      RVA  Pointer
-------- ------- -------- -------- --------
35707659 cv            25 003A3A40   3A0440    Format: RSDS, {11BC9A51-3F11-40CA-359E-CDF50F0122C1}, 1, ntkrnlmp.pdb
35707659 coffgrp     145C 003A3A68   3A0468    50475500 (PGU)
35707659 repro         24 003A4EEC   3A18EC    51 9A BC 11 11 3F CA 40 35 9E CD F5 0F 01 22 C1 82 9F 4D 74 1A 84 81 FD 12 13 E7 04 59 76 70 35
```


show only diffs
checksum
Private build
Official build
rsrc size
time stamp
confirmed by link dump headers
PDB ID
VER_FILEOS (00040004)
VER_PRIVATE (1 bit)
rsrc diffs
confirmed by RSRC editor
bcryptprimitives.dll - Hex Compare - Beyond Compare
E:\Winbuilds\17763\SignRemoved\bcryptprimitives.dll
D:\17763\obj\amd64fre\onecore\ds\security\cryptoapi\ncrypt\msprim\msprim\objfre\deterministic\bcryptprimitives.dll
VALUE "CompanyName", "Microsoft Corporation"
VALUE "FileDescription", "Windows Cryptographic Primitives Library"
VALUE "FileVersion", "10.0.17763.1 (WinBuild.160101.0800)"
VALUE "InternalName", "bcryptprimitives.dll"
VALUE "LegalCopyright", "\xA9 Microsoft Corporation. All rights reserved."
VALUE "OriginalFilename", "bcryptprimitives.dll"
VALUE "ProductName", "Microsoft\xAE Windows\xAE Operating System"
VALUE "ProductVersion", "10.0.17763.1"
VALUE "CompanyName", "Microsoft Corporation"
VALUE "FileDescription", "Windows Cryptographic Primitives Library"
VALUE "FileVersion", "10.0.17763.1 (GitEnlistment(mikelai).181025-0905)"
VALUE "InternalName", "bcryptprimitives.dll"
VALUE "LegalCopyright", "\xA9 Microsoft Corporation. All rights reserved."
VALUE "OriginalFilename", "bcryptprimitives.dll"
VALUE "ProductName", "Microsoft\xAE Windows\xAE Operating System"
VALUE "ProductVersion", "10.0.17763.1"

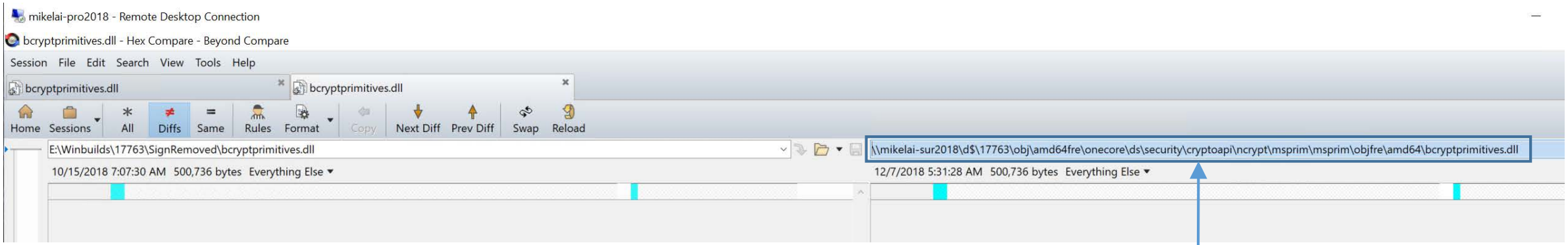
show only diffs

Official build

Private build



Showing empty – i.e. no difference

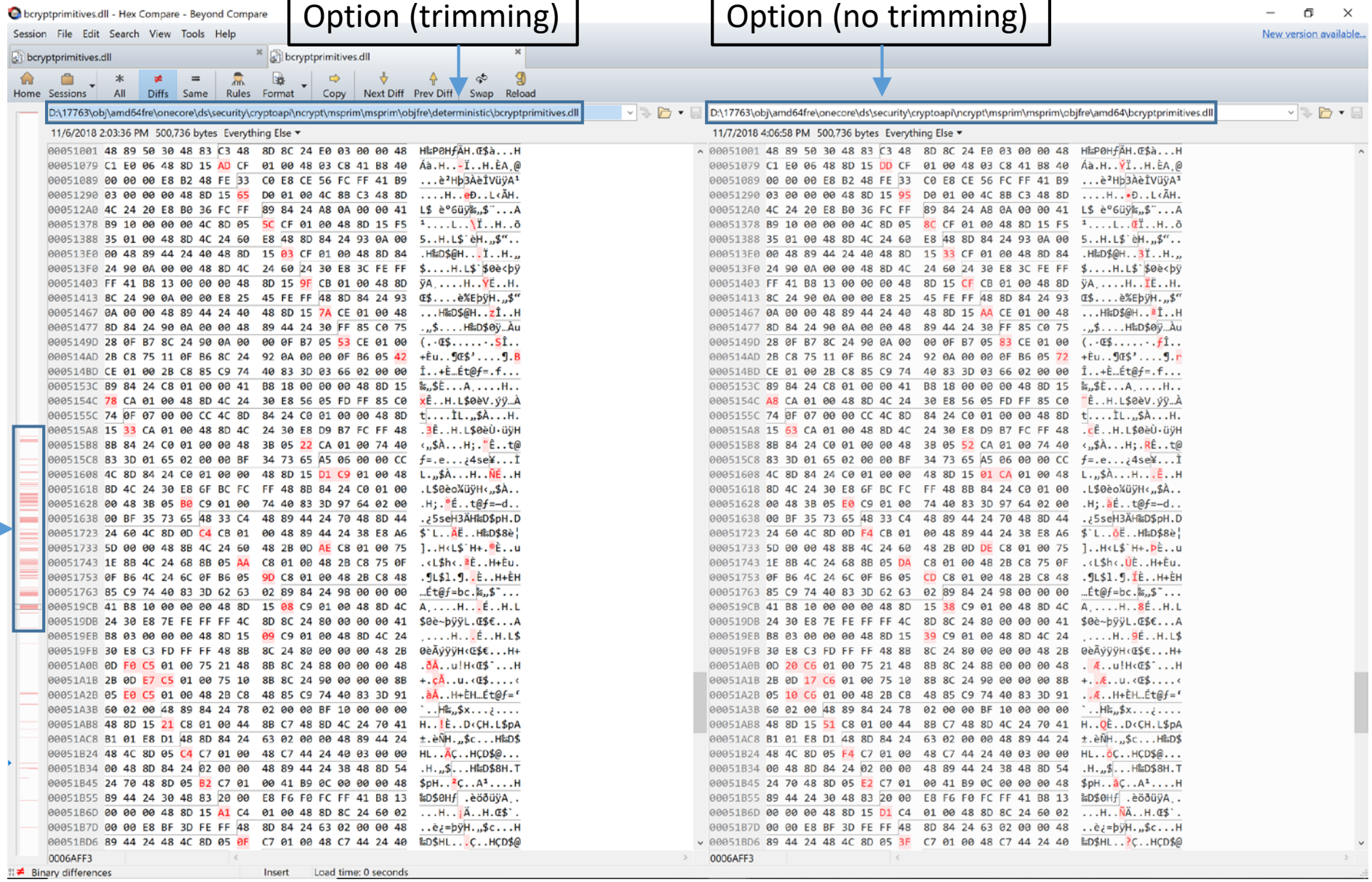


Also works when private build occurs in a remote machine

Private build with deterministic Option (trimming)

Private build without deterministic Option (no trimming)

More diffs to explain



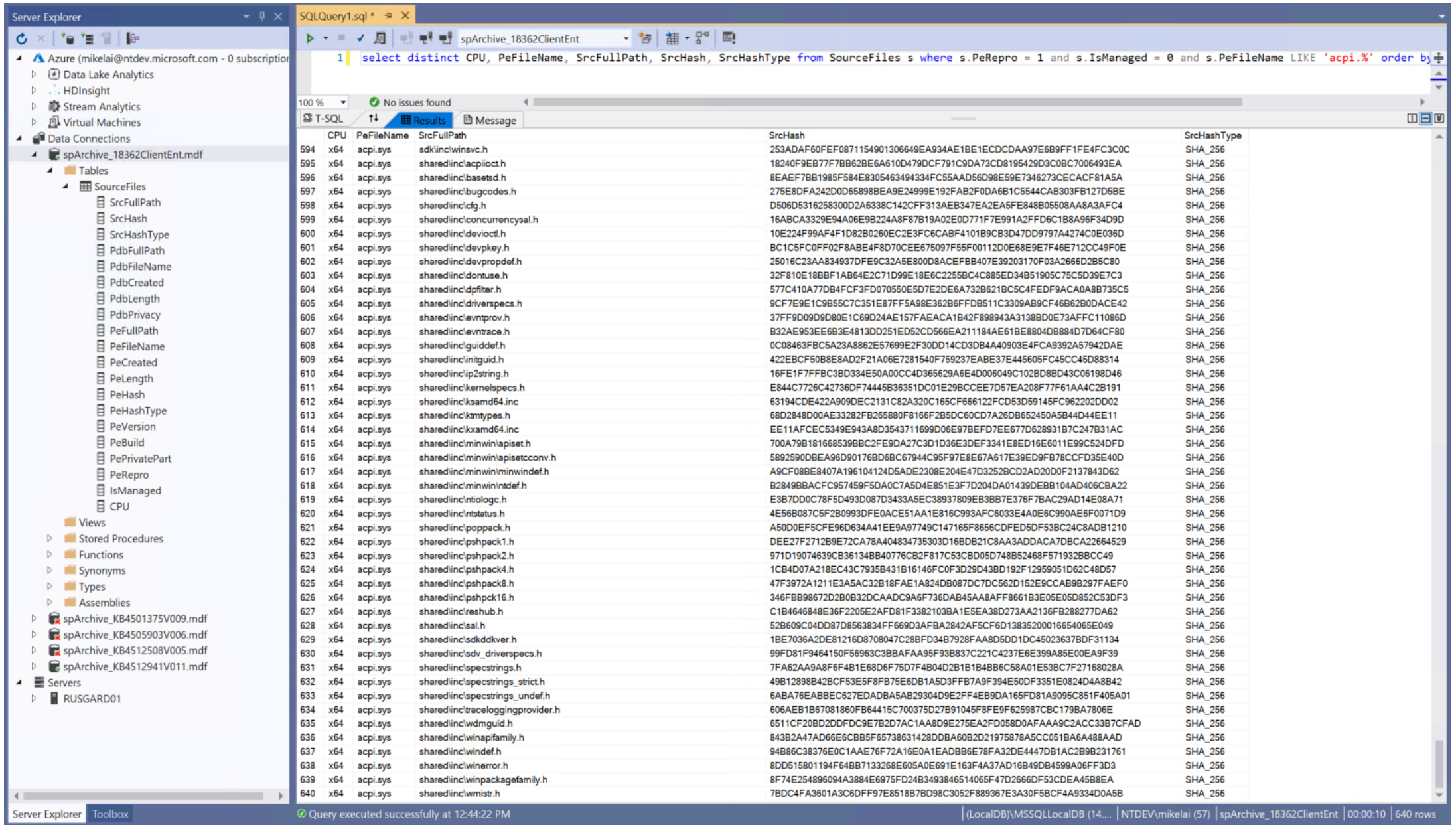
Enterprise

Change Edition

Attribute Name	Attribute Value
Product Group	windows-desktop
Edition Type	Official
Edition Family	Client
Is Virtual Edition?	True
Parent Edition	Professional
Disk Footprint	15,255,828,860
Device Images (FFUs)	0
Neutral Packages	1282
Hosted APIsets	701
Binaries	6465
WoW Binaries	495
All Files	16085

Usage scenarios

Source hash database for query



Compatible with SARIF (Static Analysis Results Interchange Format)

According to https://docs.oasis-open.org/sarif/sarif/v2.1.0/sarif-v2.1.0.html

EXAMPLE:

```
"artifacts": [
  {
    "location": {
      "uri": "file:///C:/Code/main.c"
    },
    "sourceLanguage": "c",
    "hashes": {
      "sha-256": "b13ce2678a8807ba0765ab94a0ecd394f869bc81"
    }
  }
}
```

Identify delta source files for an update

2	Old PE:	d:\media\18362clientent\mountidx3\windows\system32\drivers\acpi.sys (x64)		
3	New PE:	d:\media\kb4512941v011\windows10.0-kb4512941-x64\amd64_dual_acpi.inf_31bf3856ad364e35_10.0.18362.329_none_b5ad09708ddd3578\acpi.sys (x64)		
4				
5	**Label**	**CPU**	**Source Hash**	**Source Path**
6	18362ClientEnt	x64	C8189D3CB51B99B57932E0B4994D6B05FC306349E475B2307C94FAA6C6027C7B	ddk\inc\ntddk.h
7	KB4512941V011	x64	9F64E685979B4340C9ECB0863B50C4E9C24C2AF8903AFEC9E37246943C99450F	ddk\inc\ntddk.h
8	18362ClientEnt	x64	FE4A0FF22FC952F276D09BFA44D5F65C24CD6FAD4D6660D8223C30B5A75CDB50	ddk\inc\wdm.h
9	KB4512941V011	x64	7C3C6F8E254FC81C2682813B6F7C723614E96E0DEA712D3BB285EF6A6E3A6326	ddk\inc\wdm.h
10	18362ClientEnt	x64	073796FB5EF24AB3B8EE0383F658A537D268A2B105F6DC5307A31B58BCA4E92F	internal\minwin\priv_sdk\inc\hvgdk_mini.h
11	KB4512941V011	x64	DA7936DA0F48EC2F6CDCFABA7C9971F1A33C90206F6EACABB13EF8305EEAEF81	internal\minwin\priv_sdk\inc\hvgdk_mini.h
12	18362ClientEnt	x64	FDC299B0700CA24D2F9625228DC354348E68E309A59E3513186442C482D9136C	internal\minwin\priv_sdk\inc\iumtypes.h
13	KB4512941V011	x64	30A6C17473338D4B54DB23A3A50087823F8CA102A04CA03FD60D4D7FB5DE8BDA	internal\minwin\priv_sdk\inc\iumtypes.h
14	18362ClientEnt	x64	278D777B40BB49DDF5A71DB34CB22D4421CD7F9E060ED13D1E94E1E5AB8FAF5C	internal\minwin\priv_sdk\inc\kernel-processor-power-events.h
15	KB4512941V011	x64	90217EBDB320D953A96779D8F2C2FDC0F2073223C313BA5E657B1C513CC891E5	internal\minwin\priv_sdk\inc\kernel-processor-power-events.h
16	18362ClientEnt	x64	D009EAE94F996512FA0BA3AC036903715B5EFE566F8B10A49123BB3F15241BF1	internal\minwin\priv_sdk\inc\ntpoapi_p.h
17	KB4512941V011	x64	54183CC924B9C254F2B457F9368137929BE0DC2564F818A30605E15D8755BFC5	internal\minwin\priv_sdk\inc\ntpoapi_p.h
18	18362ClientEnt	x64	64E64A96488F854A266E1F4844C713ABC1983AA8E2D0BF7E57A070948CC7995E	internal\sdk\inc\minwin\ntosp.h
19	KB4512941V011	x64	2DB41A2E487A0D81794320165669F4D5E58F046AAF222491DF18CD277E380808	internal\sdk\inc\minwin\ntosp.h
20	18362ClientEnt	x64	842A433BA4EEE8E893E4FB338741216C76E185427F639A2982D467B4EBE820F5	internal\sdk\inc\minwin\zwapi.h
21	KB4512941V011	x64	479BBD7BB0DB3C421850209930BBC25FF2BDEFE3AF081E7FC9D42A8E82A746C7	internal\sdk\inc\minwin\zwapi.h
22	18362ClientEnt	x64	69513109A6015D6B3C19B431C5A81112A5C94567FE2CE09F2D65A3349A63BE5A	internal\sdk\inc\wil\staging.h
23	KB4512941V011	x64	FE0947C2DFA272F52D6AC0391AACC84C36A2B1D9943D226905D281C493BB1DE1	internal\sdk\inc\wil\staging.h
24	18362ClientEnt	x64	3A7867622180907F1F6CDD258B7B4071E049C03F264522AE59AB008B704E74D2	minkernel\busdrv\acpi\driver\inc\emguid.h
25	KB4512941V011	x64	BEE3074A1ECAF1018382AE5E3D922C81C7B37FBE681CFA3E2FAD7E1F0F05B447	minkernel\busdrv\acpi\driver\inc\emguid.h
26	18362ClientEnt	x64	81E793E0D14224555FED270026FBA41F209FAE3FD444E4F9C6EF8FA2597677E6	minkernel\busdrv\acpi\driver\intsup\init.c
27	KB4512941V011	x64	E5ABA551139745BDDE81CEA40564D25F75DF92E134C06E6AC7DCD62A428E6732	minkernel\busdrv\acpi\driver\intsup\init.c
28	18362ClientEnt	x64	45B65827814C0200208E3B93AC5E4E3021F0DB8073BE349E3401B1A53A266D76	minkernel\busdrv\acpi\driver\intsup\irqarb.c
29	KB4512941V011	x64	115E6452B2A9164A9A241B005B936A69EFF0888C66B6CF57C465C9F763A285E2	minkernel\busdrv\acpi\driver\intsup\irqarb.c
30	18362ClientEnt	x64	400FBB40BE02F75F6B2BAA9E0296AB02F864E4C517AF6C21BF1626513C078EE9	minkernel\busdrv\acpi\driver\intsup\irqarb.h
31	KB4512941V011	x64	A43CE9AFBFAAAE04092FD4445CDD2979D9AA4D87B3EC6AF69265915B93AFEBAB	minkernel\busdrv\acpi\driver\intsup\irqarb.h

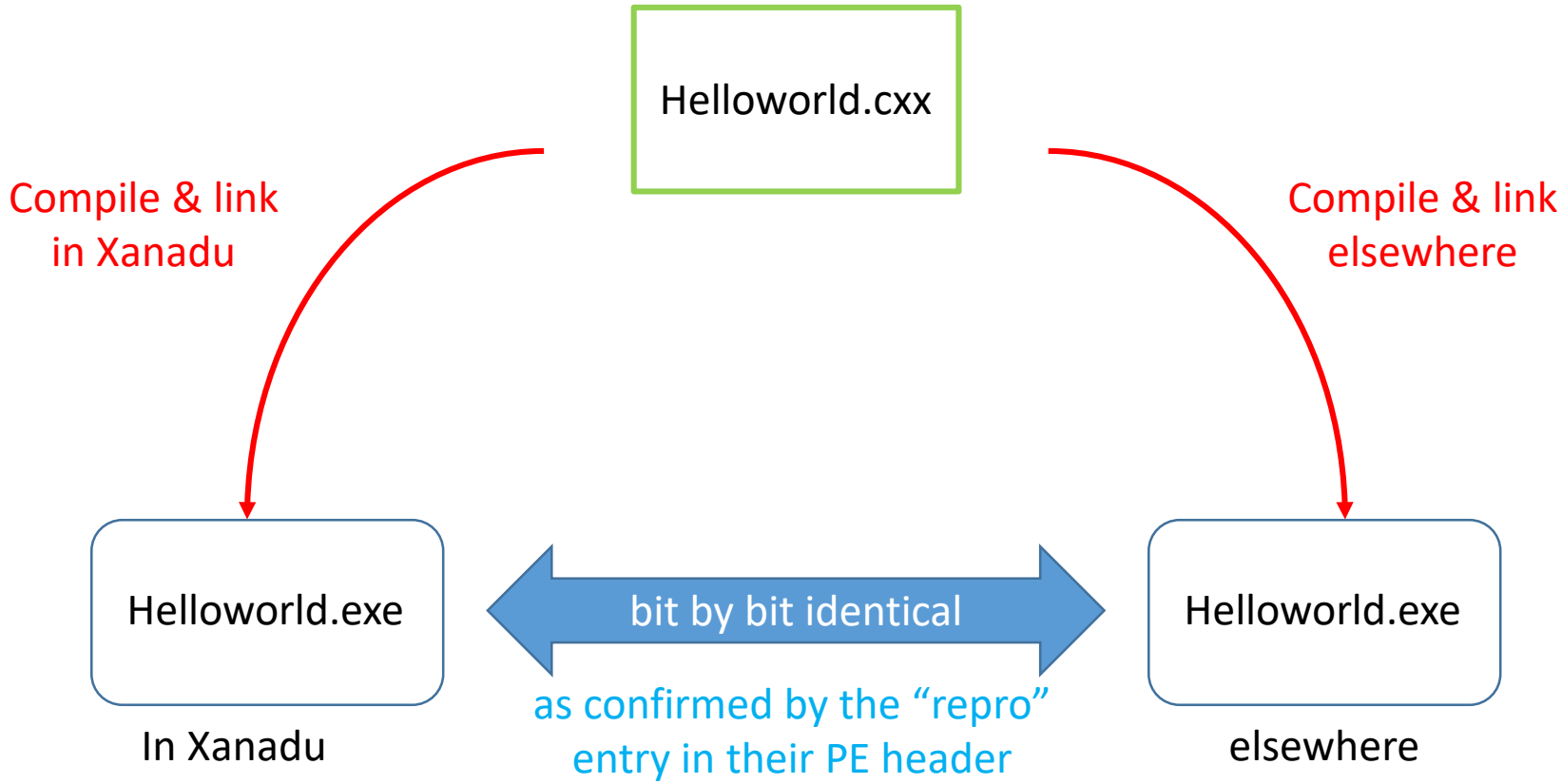
Implication to updates

- The more deterministic the executable generation is, the smaller the delta between the original and the update is

- The smaller the delta is, the tinier the bandwidth is needed to distribute the update

A vision for the software ecosystem future

Compilation Location Irrelevancy



Implies:

1) support of business recovery,
2) potential detection of unauthorized modification,
3) security audit / evaluation may be based on easier comparison or matching,
4) new business models / processes

Final Suggestion in 4 Steps

1. From the original, identify all the pieces
2. Clone a copy of the pieces
3. Reproduce from the cloned pieces
4. Compare the reproduced with the original

And in the case of OSS development

- An archive of the development environment that you originally used to build your project is critical

- A best practice described in https://www.freecodecamp.org/news/put-your-dev-env-in-github

Questions