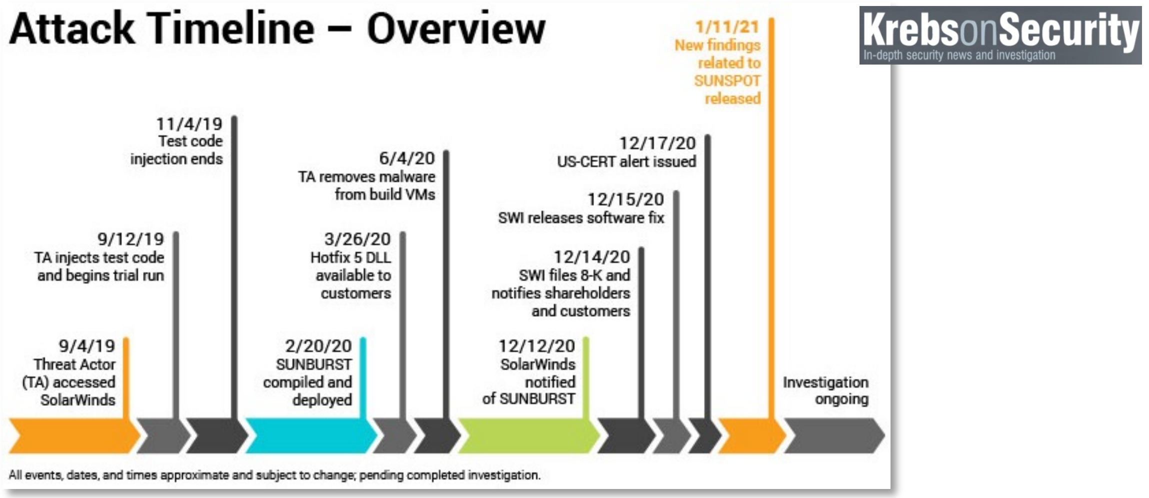
Software Fingerprinting for Supply Chain Security

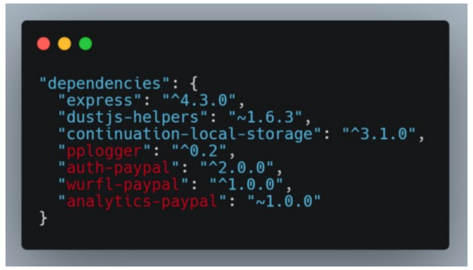
—

Ted Habeck
Jiyong Jang
Dhilung Kirat
Ian Molloy
JR Rao
Doug Schales

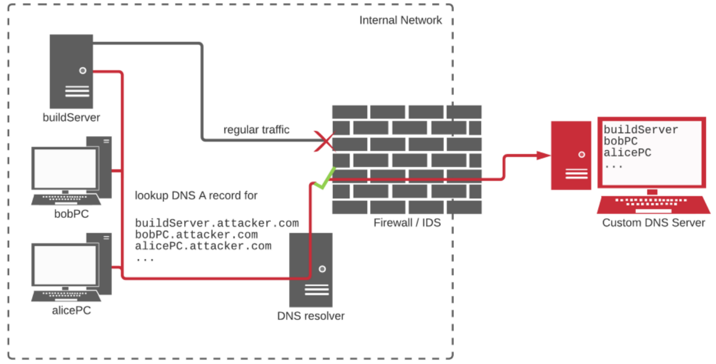
Supply Chain Attacks

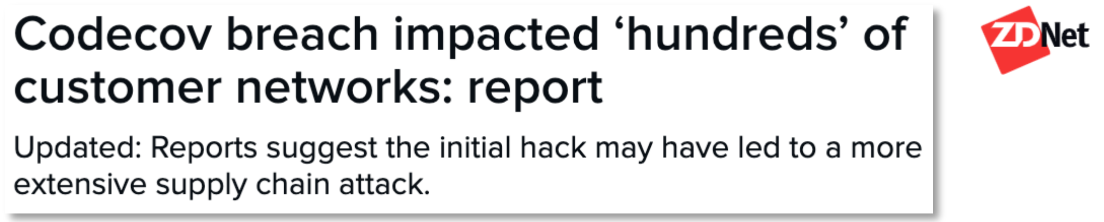
- SolarWinds (2019-2021) est. cost > $100B
  - Malicious code (backdoor) pushed out through updates
- Dependency confusion (Feb 2021)
  - Private vs public packages (npm, PyPi, RubyGems)
- Codecov (Apr 2021)
  - DevOps tool. Vulnerability in CI. Bash uploader modified
- Kaseya (Jul 2021) ransom $70M
  - IT solutions, including VSA (remote monitoring and management software) to deliver REvil ransomware
- Protestware (Mar 2022)
  - Popular NPM package wiped files in Russia and Belarus


Attack Timeline – Overview
KrebsonSecurity
In-depth security news and investigation
9/4/19 Threat Actor (TA) accessed SolarWinds
9/12/19 TA injects test code and begins trial run
11/4/19 Test code injection ends
2/20/20 SUNBURST compiled and deployed
3/26/20 Hotfix 5 DLL available to customers
6/4/20 TA removes malware from build VMs
12/12/20 SolarWinds notified of SUNBURST
12/14/20 SWI files 8-K and notifies shareholders and customers
12/15/20 SWI releases software fix
12/17/20 US-CERT alert issued
1/11/21 New findings related to SUNSPOT released
Investigation ongoing
All events, dates, and times approximate and subject to change; pending completed investigation.



```
"dependencies": {
  "express": "^4.3.0",
  "dustjs-helpers": "~1.6.3",
  "continuation-local-storage": "^3.1.0",
  "pplogger": "^0.2",
  "auth-paypal": "^2.0.0",
  "wurfl-paypal": "^1.0.0",
  "analytics-paypal": "~1.0.0"
}
```

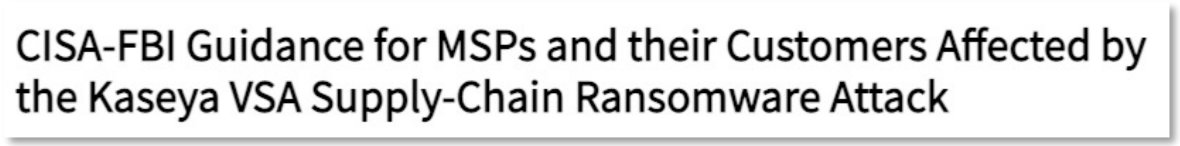




Codecov breach impacted 'hundreds' of customer networks: report

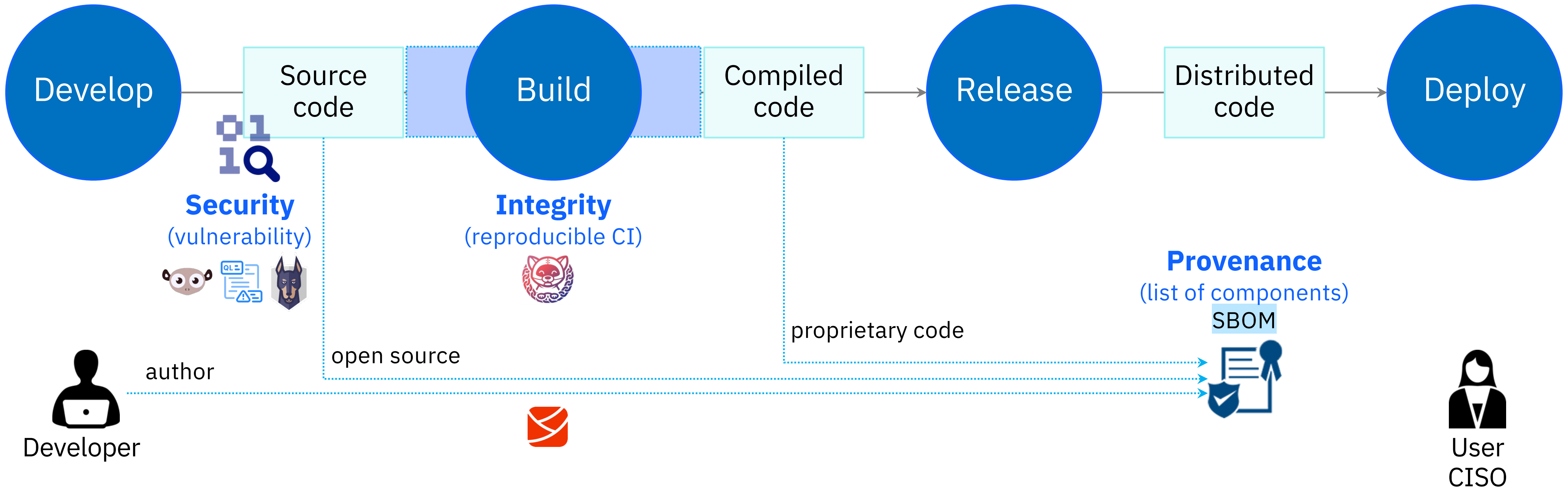
Updated: Reports suggest the initial hack may have led to a more extensive supply chain attack.

ZDNet



CISA-FBI Guidance for MSPs and their Customers Affected by the Kaseya VSA Supply-Chain Ransomware Attack

Supply Chain Security
Industry approach to protecting CI/CD pipelines

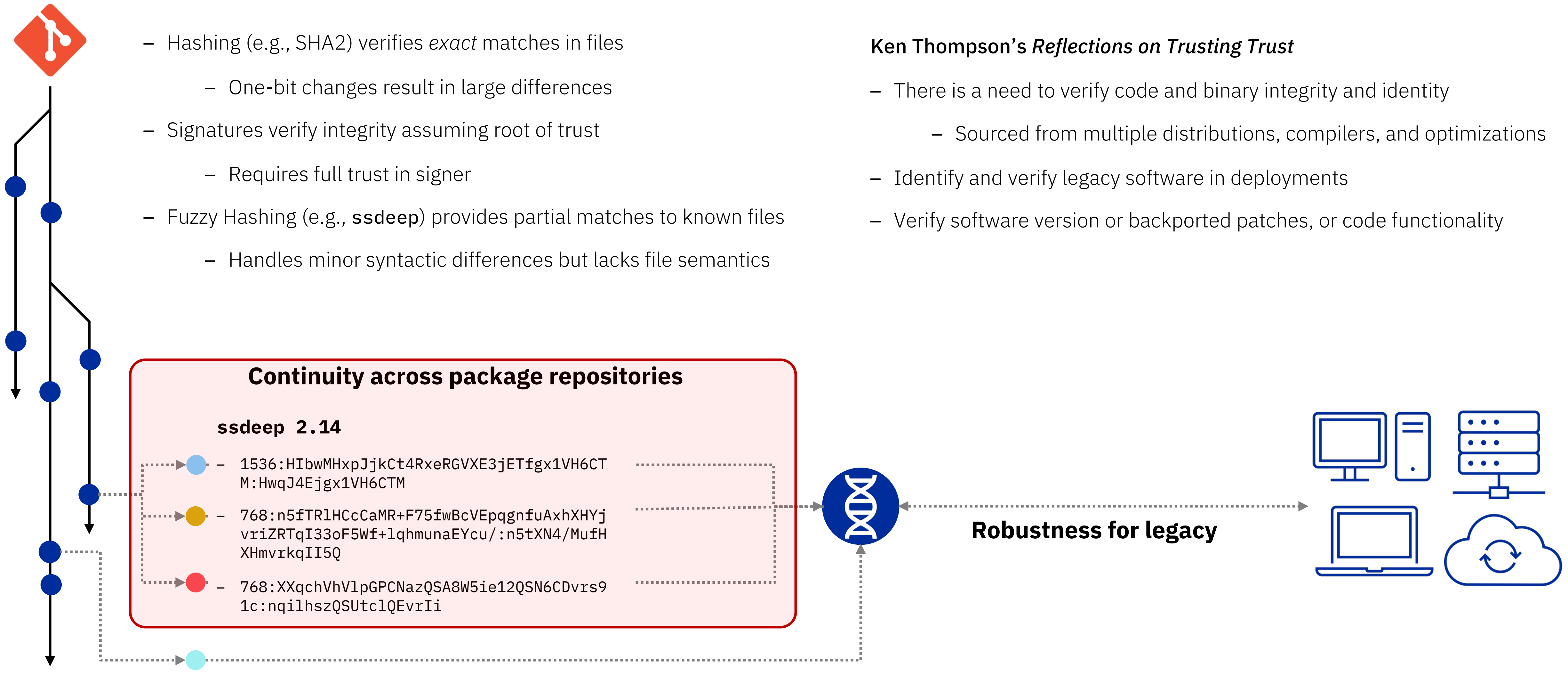


Proposal: Software Fingerprints for Software Assurance

- Hashing (e.g., SHA2) verifies *exact* matches in files
  - One-bit changes result in large differences
- Signatures verify integrity assuming root of trust
  - Requires full trust in signer
- Fuzzy Hashing (e.g., **`ssdeep`**) provides partial matches to known files
  - Handles minor syntactic differences but lacks file semantics

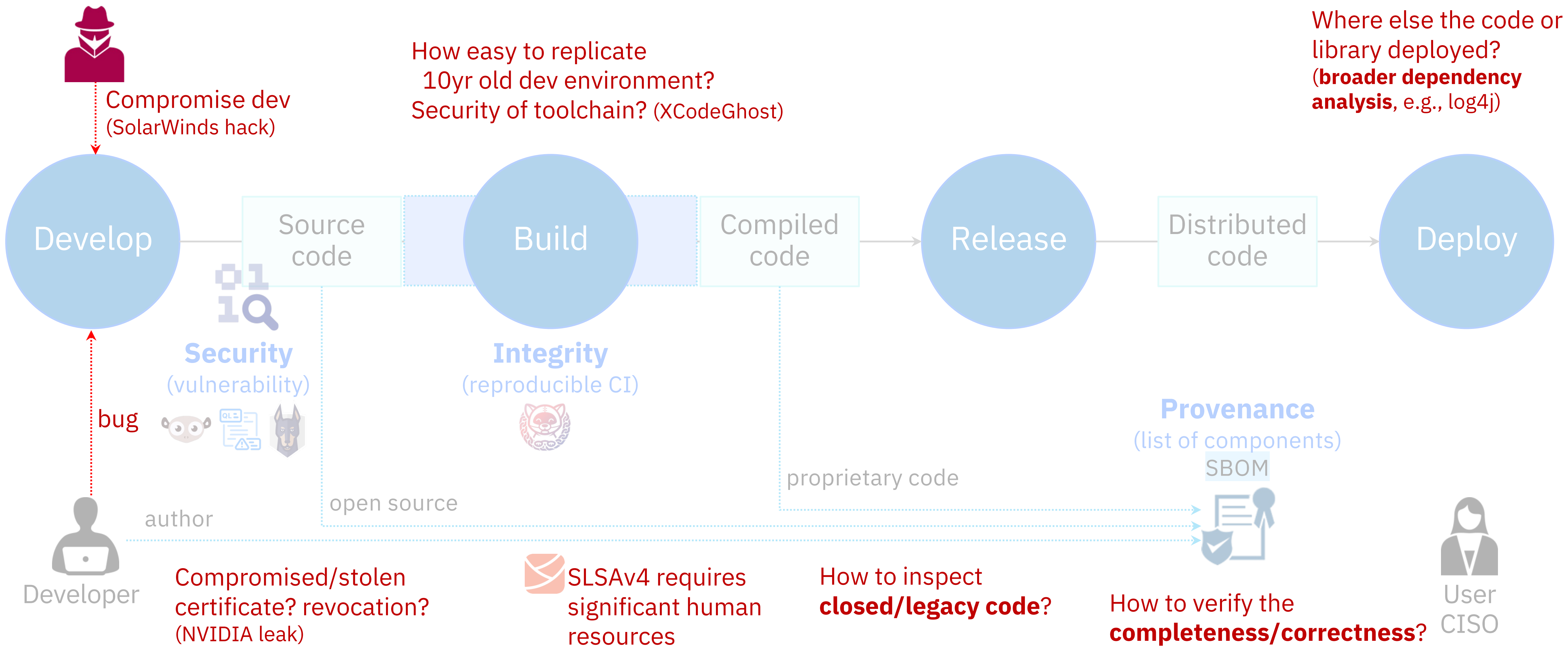
Ken Thompson's *Reflections on Trusting Trust*

- There is a need to verify code and binary integrity and identity
  - Sourced from multiple distributions, compilers, and optimizations
- Identify and verify legacy software in deployments
- Verify software version or backported patches, or code functionality


Continuity across package repositories
Robustness for legacy
Stability across versions

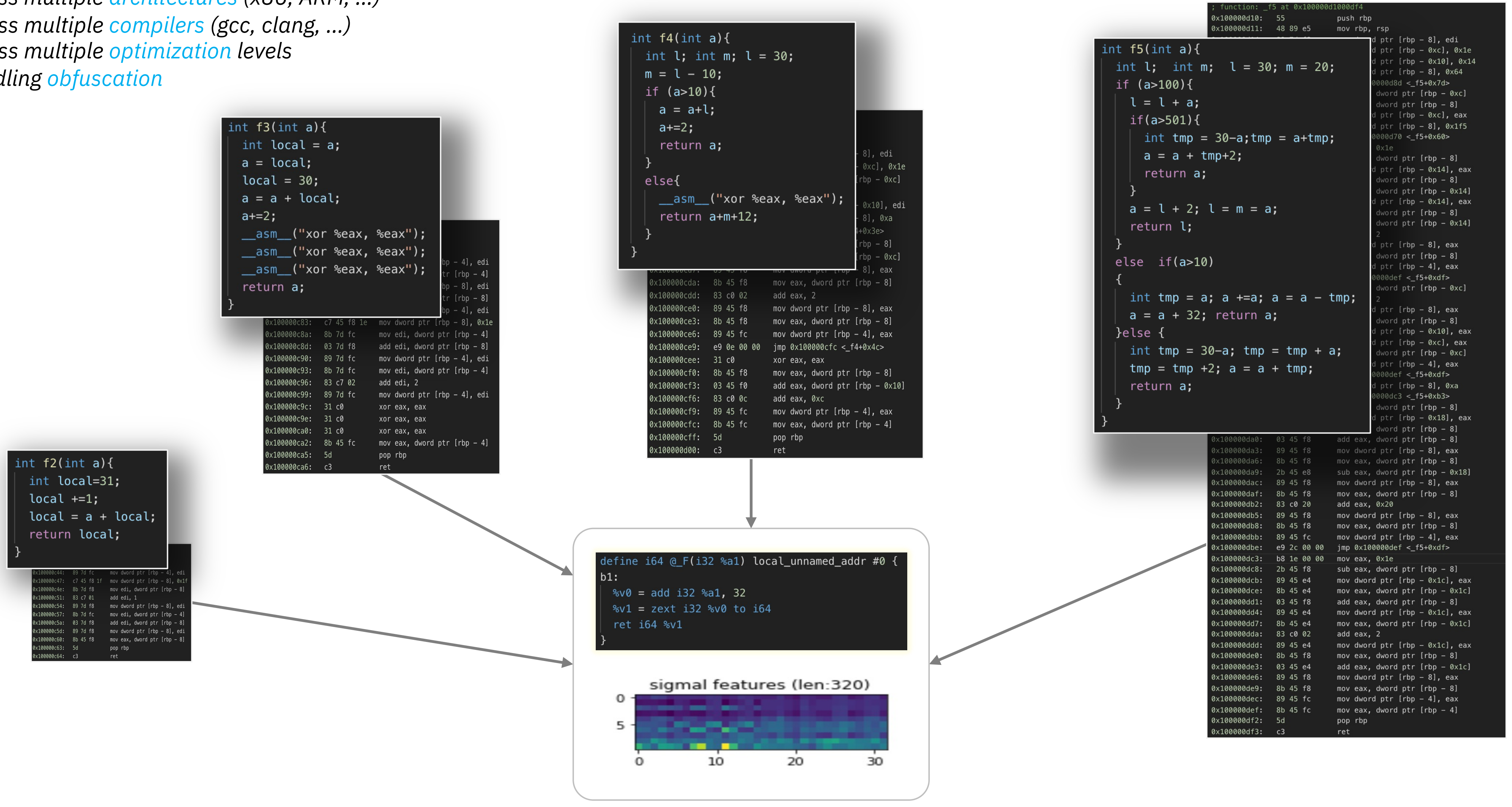
```
ssdeep 2.14

– 1536:HIbwMHxpJjkCt4RxeRGVXE3jETfgx1VH6CT
  M:HwqJ4Ejgx1VH6CTM
– 768:n5fTRlHCcCaMR+F75fwBcVEpqgnfuAxhXHYj
  vriZRTqI33oF5Wf+lqhmunaEYcu/:n5tXN4/MufH
  XHmvrkqII5Q
– 768:XXqchVhVlpGPCNazQSA8W5ie12QSN6CDvrs9
  1c:nqilhszQSUtclQEvrIi
```

Supply Chain Security
Open security issues and residual risks


Compromise dev
(SolarWinds hack)
How easy to replicate
10yr old dev environment?
Security of toolchain? (XCodeGhost)
Where else the code or library deployed?
(**broader dependency analysis**, e.g., log4j)
Develop
Source code
Build
Compiled code
Release
Distributed code
Deploy
Security
(vulnerability)
Integrity
(reproducible CI)
bug
Provenance
(list of components)
SBOM
proprietary code
open source
author
Developer
User
CISO
Compromised/stolen certificate? revocation?
(NVIDIA leak)
SLSAv4 requires significant human resources
How to inspect **closed/legacy code**?
How to verify the **completeness/correctness**?

Code Genome: Semantically meaningful fingerprint

- *Across multiple architectures (x86, ARM, …)*
- *Across multiple compilers (gcc, clang, …)*
- *Across multiple optimization levels*
- *Handling obfuscation*



```c
int f2(int a){
  int local=31;
  local +=1;
  local = a + local;
  return local;
}
```

```
0x100000c44:   89 7d fc      mov dword ptr [rbp - 4], edi
0x100000c47:   c7 45 f8 1f   mov dword ptr [rbp - 8], 0x1f
0x100000c4e:   8b 7d f8      mov edi, dword ptr [rbp - 8]
0x100000c51:   83 c7 01      add edi, 1
0x100000c54:   89 7d f8      mov dword ptr [rbp - 8], edi
0x100000c57:   8b 7d fc      mov edi, dword ptr [rbp - 4]
0x100000c5a:   03 7d f8      add edi, dword ptr [rbp - 8]
0x100000c5d:   89 7d f8      mov dword ptr [rbp - 8], edi
0x100000c60:   8b 45 f8      mov eax, dword ptr [rbp - 8]
0x100000c63:   5d            pop rbp
0x100000c64:   c3            ret
```

```c
int f3(int a){
  int local = a;
  a = local;
  local = 30;
  a = a + local;
  a+=2;
  __asm__("xor %eax, %eax");
  __asm__("xor %eax, %eax");
  __asm__("xor %eax, %eax");
  return a;
}
```

```
                                  bp - 4], edi
                                  tr [rbp - 4]
                                  bp - 8], edi
                                  tr [rbp - 8]
                                  bp - 4], edi
0x100000c83:   c7 45 f8 1e   mov dword ptr [rbp - 8], 0x1e
0x100000c8a:   8b 7d fc      mov edi, dword ptr [rbp - 4]
0x100000c8d:   03 7d f8      add edi, dword ptr [rbp - 8]
0x100000c90:   89 7d fc      mov dword ptr [rbp - 4], edi
0x100000c93:   8b 7d fc      mov edi, dword ptr [rbp - 4]
0x100000c96:   83 c7 02      add edi, 2
0x100000c99:   89 7d fc      mov dword ptr [rbp - 4], edi
0x100000c9c:   31 c0         xor eax, eax
0x100000c9e:   31 c0         xor eax, eax
0x100000ca0:   31 c0         xor eax, eax
0x100000ca2:   8b 45 fc      mov eax, dword ptr [rbp - 4]
0x100000ca5:   5d            pop rbp
0x100000ca6:   c3            ret
```

```c
int f4(int a){
  int l; int m; l = 30;
  m = l - 10;
  if (a>10){
    a = a+l;
    a+=2;
    return a;
  }
  else{
    __asm__("xor %eax, %eax");
    return a+m+12;
  }
}
```

```
                                  8], edi
                                  0xc], 0x1e
                                  [rbp - 0xc]

                                  0x10], edi
                                  8], 0xa
                                  4+0x3e>
                                  [rbp - 8]
                                  [rbp - 0xc]
                                  8], eax
0x100000cda:   8b 45 f8      mov eax, dword ptr [rbp - 8]
0x100000cdd:   83 c0 02      add eax, 2
0x100000ce0:   89 45 f8      mov dword ptr [rbp - 8], eax
0x100000ce3:   8b 45 f8      mov eax, dword ptr [rbp - 8]
0x100000ce6:   89 45 fc      mov dword ptr [rbp - 4], eax
0x100000ce9:   e9 0e 00 00   jmp 0x100000cfc <_f4+0x4c>
0x100000cee:   31 c0         xor eax, eax
0x100000cf0:   8b 45 f8      mov eax, dword ptr [rbp - 8]
0x100000cf3:   03 45 f0      add eax, dword ptr [rbp - 0x10]
0x100000cf6:   83 c0 0c      add eax, 0xc
0x100000cf9:   89 45 fc      mov dword ptr [rbp - 4], eax
0x100000cfc:   8b 45 fc      mov eax, dword ptr [rbp - 4]
0x100000cff:   5d            pop rbp
0x100000d00:   c3            ret
```

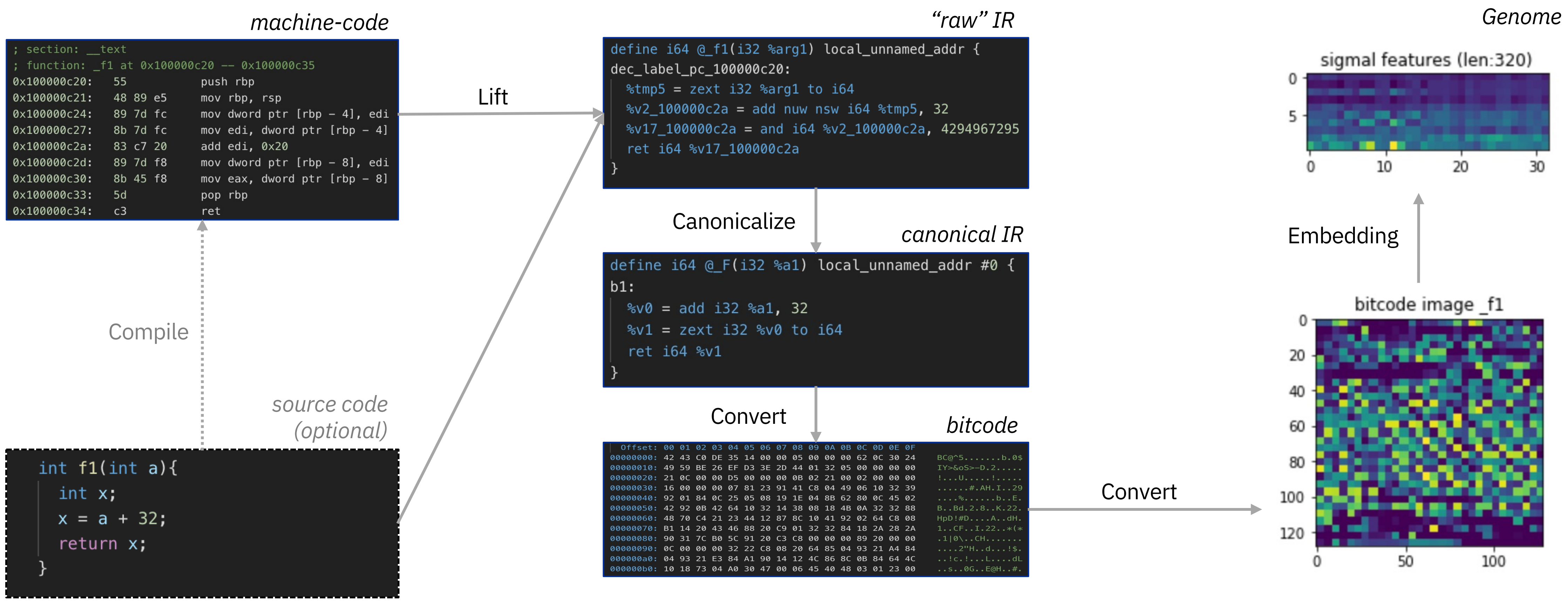
```c
int f5(int a){
  int l;  int m;  l = 30; m = 20;
  if (a>100){
    l = l + a;
    if(a>501){
      int tmp = 30-a;tmp = a+tmp;
      a = a + tmp+2;
      return a;
    }
    a = l + 2; l = m = a;
    return l;
  }
  else  if(a>10)
  {
    int tmp = a; a +=a; a = a - tmp;
    a = a + 32; return a;
  }else {
    int tmp = 30-a; tmp = tmp + a;
    tmp = tmp +2; a = a + tmp;
    return a;
  }
}
```

```
; function: _f5 at 0x100000d1000df4
0x100000d10:   55            push rbp
0x100000d11:   48 89 e5      mov rbp, rsp
                             d ptr [rbp - 8], edi
                             d ptr [rbp - 0xc], 0x1e
                             d ptr [rbp - 0x10], 0x14
                             d ptr [rbp - 8], 0x64
                             0000d8d <_f5+0x7d>
                              dword ptr [rbp - 0xc]
                              dword ptr [rbp - 8]
                             d ptr [rbp - 0xc], eax
                             d ptr [rbp - 8], 0x1f5
                             0000d70 <_f5+0x60>
                              0x1e
                              dword ptr [rbp - 8]
                             d ptr [rbp - 0x14], eax
                              dword ptr [rbp - 8]
                              dword ptr [rbp - 0x14]
                             d ptr [rbp - 0x14], eax
                              dword ptr [rbp - 8]
                              dword ptr [rbp - 0x14]
                              2
                             d ptr [rbp - 8], eax
                              dword ptr [rbp - 8]
                             d ptr [rbp - 4], eax
                             0000def <_f5+0xdf>
                              dword ptr [rbp - 0xc]
                              2
                             d ptr [rbp - 8], eax
                              dword ptr [rbp - 8]
                             d ptr [rbp - 0x10], eax
                             d ptr [rbp - 0xc], eax
                              dword ptr [rbp - 0xc]
                             d ptr [rbp - 4], eax
                             0000def <_f5+0xdf>
                             d ptr [rbp - 8], 0xa
                             0000dc3 <_f5+0xb3>
                              dword ptr [rbp - 8]
                             d ptr [rbp - 0x18], eax
                              dword ptr [rbp - 8]
0x100000da0:   03 45 f8      add eax, dword ptr [rbp - 8]
0x100000da3:   89 45 f8      mov dword ptr [rbp - 8], eax
0x100000da6:   8b 45 f8      mov eax, dword ptr [rbp - 8]
0x100000da9:   2b 45 e8      sub eax, dword ptr [rbp - 0x18]
0x100000dac:   89 45 f8      mov dword ptr [rbp - 8], eax
0x100000daf:   8b 45 f8      mov eax, dword ptr [rbp - 8]
0x100000db2:   83 c0 20      add eax, 0x20
0x100000db5:   89 45 f8      mov dword ptr [rbp - 8], eax
0x100000db8:   8b 45 f8      mov eax, dword ptr [rbp - 8]
0x100000dbb:   89 45 fc      mov dword ptr [rbp - 4], eax
0x100000dbe:   e9 2c 00 00   jmp 0x100000def <_f5+0xdf>
0x100000dc3:   b8 1e 00 00   mov eax, 0x1e
0x100000dc8:   2b 45 f8      sub eax, dword ptr [rbp - 8]
0x100000dcb:   89 45 e4      mov dword ptr [rbp - 0x1c], eax
0x100000dce:   8b 45 e4      mov eax, dword ptr [rbp - 0x1c]
0x100000dd1:   03 45 f8      add eax, dword ptr [rbp - 8]
0x100000dd4:   89 45 e4      mov dword ptr [rbp - 0x1c], eax
0x100000dd7:   8b 45 e4      mov eax, dword ptr [rbp - 0x1c]
0x100000dda:   83 c0 02      add eax, 2
0x100000ddd:   89 45 e4      mov dword ptr [rbp - 0x1c], eax
0x100000de0:   8b 45 f8      mov eax, dword ptr [rbp - 8]
0x100000de3:   03 45 e4      add eax, dword ptr [rbp - 0x1c]
0x100000de6:   89 45 f8      mov dword ptr [rbp - 8], eax
0x100000de9:   8b 45 f8      mov eax, dword ptr [rbp - 8]
0x100000dec:   89 45 fc      mov dword ptr [rbp - 4], eax
0x100000def:   8b 45 fc      mov eax, dword ptr [rbp - 4]
0x100000df2:   5d            pop rbp
0x100000df3:   c3            ret
```

```llvm
define i64 @_F(i32 %a1) local_unnamed_addr #0 {
b1:
  %v0 = add i32 %a1, 32
  %v1 = zext i32 %v0 to i64
  ret i64 %v1
}
```

Same Genome

Key Idea: Code Genome construction



machine-code

```
; section: __text
; function: _f1 at 0x100000c20 -- 0x100000c35
0x100000c20:   55          push rbp
0x100000c21:   48 89 e5    mov rbp, rsp
0x100000c24:   89 7d fc    mov dword ptr [rbp - 4], edi
0x100000c27:   8b 7d fc    mov edi, dword ptr [rbp - 4]
0x100000c2a:   83 c7 20    add edi, 0x20
0x100000c2d:   89 7d f8    mov dword ptr [rbp - 8], edi
0x100000c30:   8b 45 f8    mov eax, dword ptr [rbp - 8]
0x100000c33:   5d          pop rbp
0x100000c34:   c3          ret
```

"raw" IR

```
define i64 @_f1(i32 %arg1) local_unnamed_addr {
dec_label_pc_100000c20:
  %tmp5 = zext i32 %arg1 to i64
  %v2_100000c2a = add nuw nsw i64 %tmp5, 32
  %v17_100000c2a = and i64 %v2_100000c2a, 4294967295
  ret i64 %v17_100000c2a
}
```

canonical IR

```
define i64 @_F(i32 %a1) local_unnamed_addr #0 {
b1:
  %v0 = add i32 %a1, 32
  %v1 = zext i32 %v0 to i64
  ret i64 %v1
}
```

bitcode

```
  Offset: 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000: 42 43 C0 DE 35 14 00 00 05 00 00 00 62 0C 30 24    BC@^5.......b.0$
00000010: 49 59 BE 26 EF D3 3E 2D 44 01 32 05 00 00 00 00    IY>&oS>-D.2.....
00000020: 21 0C 00 00 D5 00 00 00 0B 02 21 00 02 00 00 00    !...U.....!.....
00000030: 16 00 00 00 07 81 23 91 41 C8 04 49 06 10 32 39    ......#.AH.I..29
00000040: 92 01 84 0C 25 05 08 19 1E 04 8B 62 80 0C 45 02    ....%......b..E.
00000050: 42 92 0B 42 64 10 32 14 38 08 18 4B 0A 32 32 88    B..Bd.2.8..K.22.
00000060: 48 70 C4 21 23 44 12 87 8C 10 41 92 02 64 C8 08    HpD!#D....A..dH.
00000070: B1 14 20 43 46 88 20 C9 01 32 32 84 18 2A 28 2A    1..CF..I.22..*(*
00000080: 90 31 7C B0 5C 91 20 C3 C8 00 00 00 89 20 00 00    .1|0\..CH.......
00000090: 0C 00 00 00 32 22 C8 08 20 64 85 04 93 21 A4 84    ....2"H..d...!$.
000000a0: 04 93 21 E3 84 A1 90 14 12 4C 86 8C 0B 84 64 4C    ..!c.!...L....dL
000000b0: 10 18 73 04 A0 30 47 00 06 45 40 48 03 01 23 00    ..s..0G..E@H..#.
```

source code (optional)

```
int f1(int a){
  int x;
  x = a + 32;
  return x;
}
```

Genome can be constructed from closed-source/legacy code where source code is not easily available.

Use Case 1: Finding Log4j

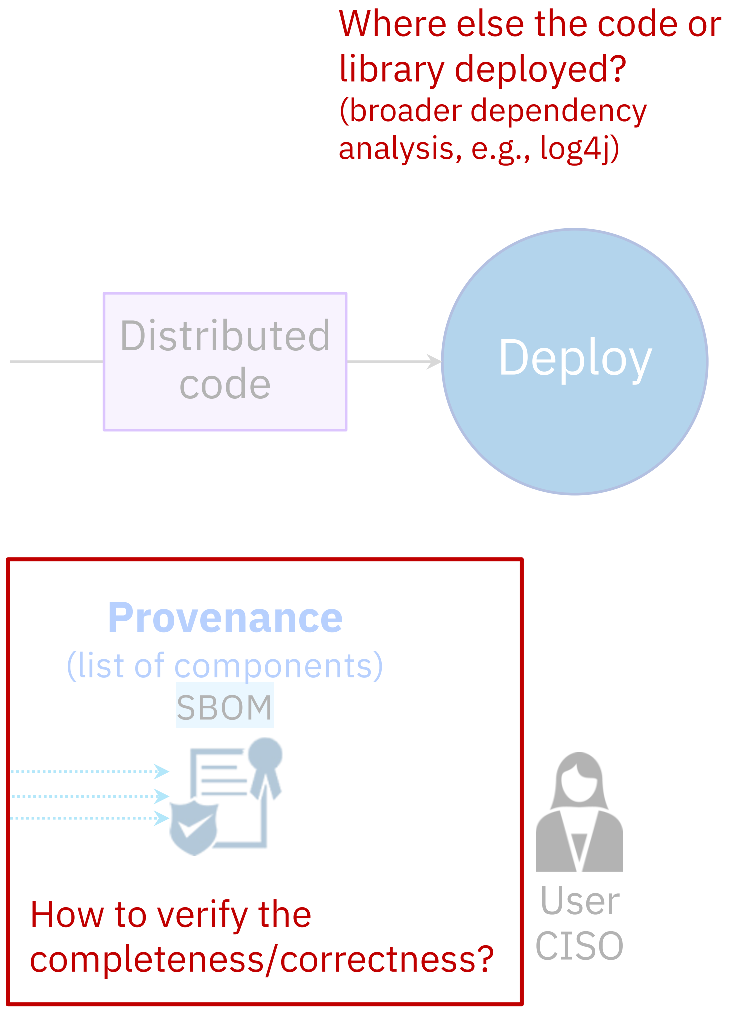
Legacy Software Discovery

- Software deployments are a turducken
  - `zip`, `tar`, container image, `jar`, etc.
  - Dependencies often wrapped up
- No good provenance or CMDB
- Can’t rely on standard directories, filenames, or hashes
- Code can be repackaged

- Applies across the board: CICD, DevSecOps and Legacy

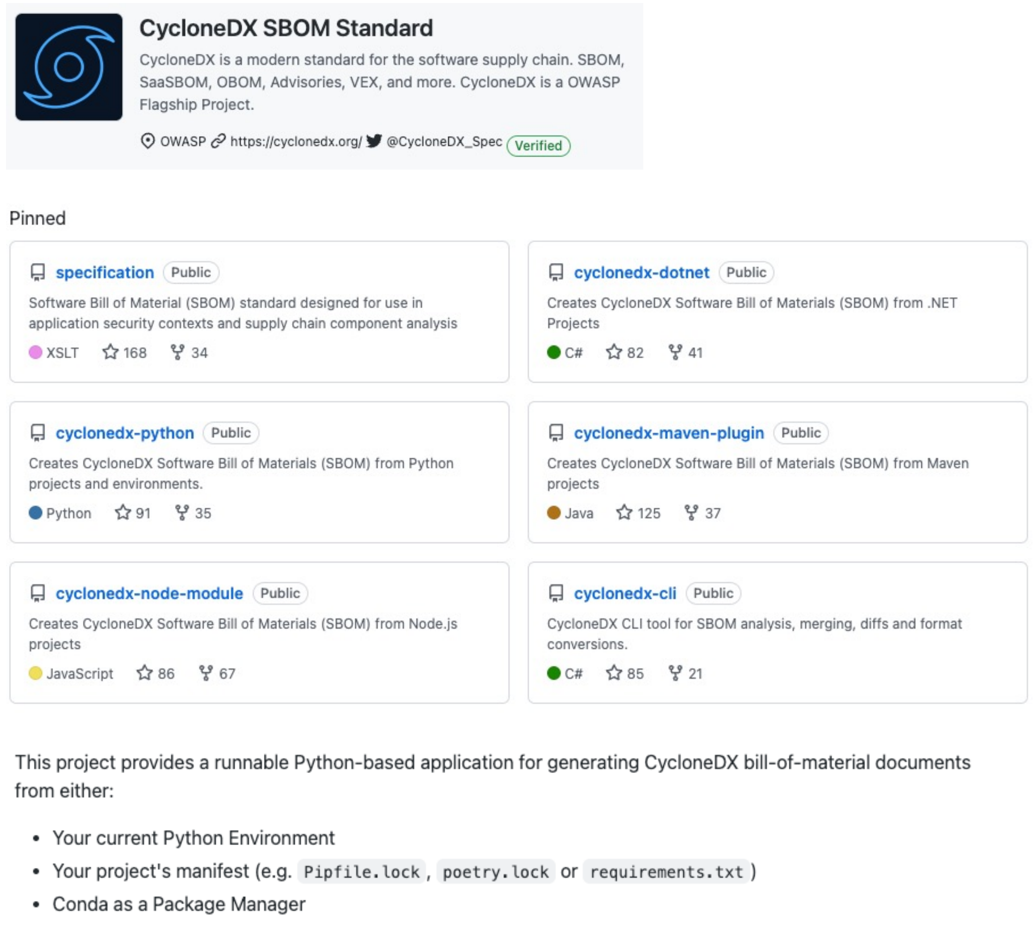
Use Case 2: SBOM Verification

- Problem
  - Each vendor creates SBOM of their own software including open-source and closed-source components. How can we verify its correctness (containing incorrect library mistakenly/maliciously) and completeness (missing library)?
- Value
  - Given software, we can verify (generate) SBOM
  - Support closed-source and legacy software without requiring source code access
  - Help developers generate correct SBOM
  - Vet software before integrating/deploying into a product



Where else the code or library deployed?
(broader dependency analysis, e.g., log4j)
Distributed code
Deploy
Provenance
(list of components)
SBOM
How to verify the completeness/correctness?
User
CISO

Trust but Verify SBOM: Metadata vs. Code



CycloneDX SBOM Standard

CycloneDX is a modern standard for the software supply chain. SBOM, SaaSBOM, OBOM, Advisories, VEX, and more. CycloneDX is a OWASP Flagship Project.

OWASP https://cyclonedx.org/ @CycloneDX_Spec Verified

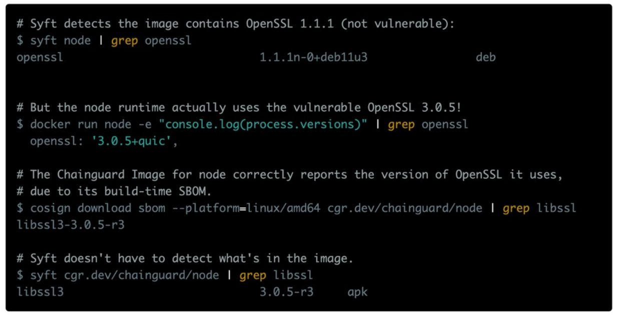
Pinned

specification Public Software Bill of Material (SBOM) standard designed for use in application security contexts and supply chain component analysis XSLT ☆ 168 ⑂ 34	**cyclonedx-dotnet** Public Creates CycloneDX Software Bill of Materials (SBOM) from .NET Projects C# ☆ 82 ⑂ 41
cyclonedx-python Public Creates CycloneDX Software Bill of Materials (SBOM) from Python projects and environments. Python ☆ 91 ⑂ 35	**cyclonedx-maven-plugin** Public Creates CycloneDX Software Bill of Materials (SBOM) from Maven projects Java ☆ 125 ⑂ 37
cyclonedx-node-module Public Creates CycloneDX Software Bill of Materials (SBOM) from Node.js projects JavaScript ☆ 86 ⑂ 67	**cyclonedx-cli** Public CycloneDX CLI tool for SBOM analysis, merging, diffs and format conversions. C# ☆ 85 ⑂ 21

This project provides a runnable Python-based application for generating CycloneDX bill-of-material documents from either:

- Your current Python Environment
- Your project's manifest (e.g. `Pipfile.lock`, `poetry.lock` or `requirements.txt`)
- Conda as a Package Manager

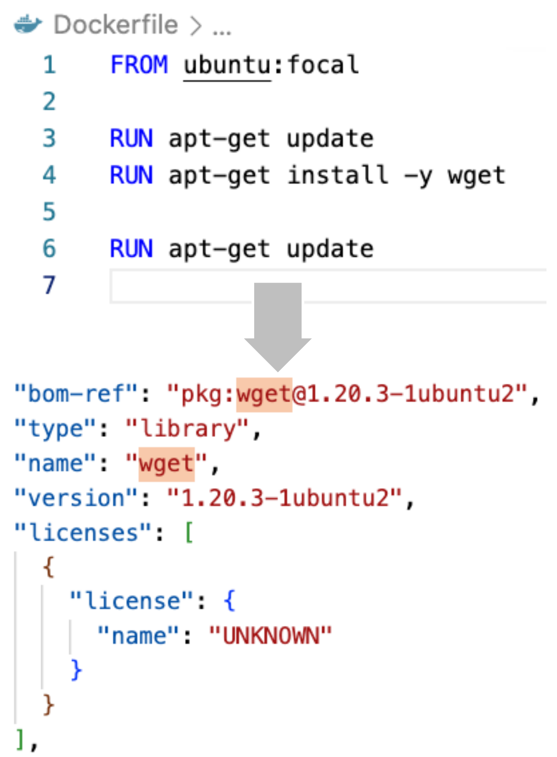
"Unfortunately, some images – such as the official node image on Docker Hub – incorrectly report the version of OpenSSL that's used by the Node.js runtime."



```
# Syft detects the image contains OpenSSL 1.1.1 (not vulnerable):
$ syft node | grep openssl
openssl                    1.1.1n-0+deb11u3          deb


# But the node runtime actually uses the vulnerable OpenSSL 3.0.5!
$ docker run node -e "console.log(process.versions)" | grep openssl
  openssl: '3.0.5+quic',

# The Chainguard Image for node correctly reports the version of OpenSSL it uses,
# due to its build-time SBOM.
$ cosign download sbom --platform=linux/amd64 cgr.dev/chainguard/node | grep libssl
libssl3-3.0.5-r3

# Syft doesn't have to detect what's in the image.
$ syft cgr.dev/chainguard/node | grep libssl
libssl3                    3.0.5-r3    apk
```

https://www.chainguard.dev/unchained/mitigating-critical-openssl-vulnerability-with-chainguard

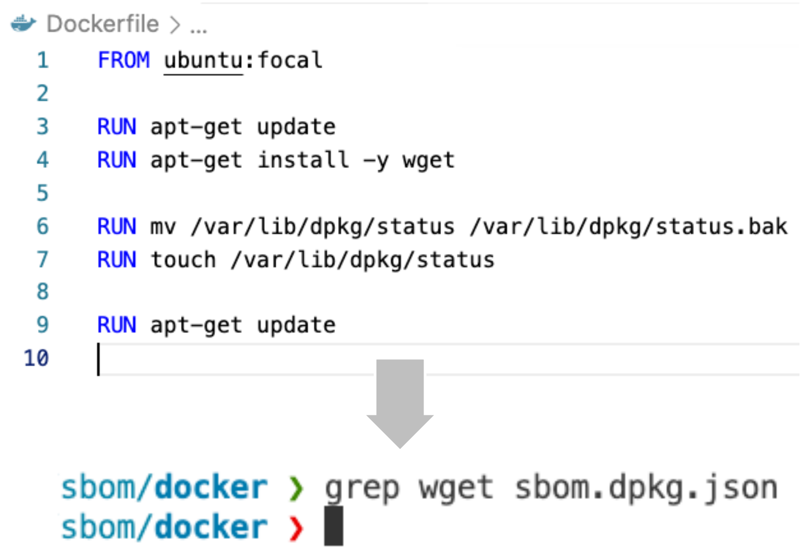
$ sbom generation tools



Dockerfile > ...

```
FROM ubuntu:focal

RUN apt-get update
RUN apt-get install -y wget

RUN apt-get update

```

```
"bom-ref": "pkg:wget@1.20.3-1ubuntu2",
"type": "library",
"name": "wget",
"version": "1.20.3-1ubuntu2",
"licenses": [
  {
    "license": {
      "name": "UNKNOWN"
    }
  }
],
```

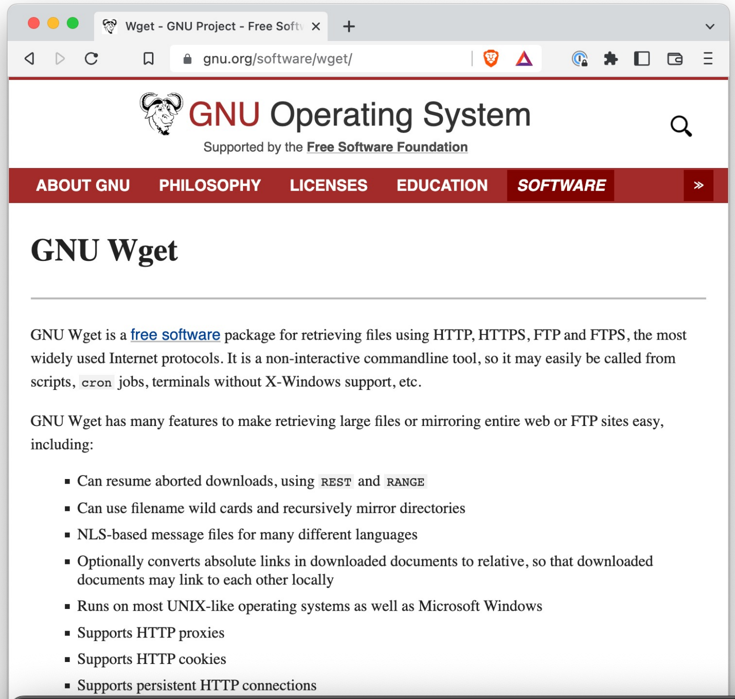


Dockerfile > ...

```
FROM ubuntu:focal

RUN apt-get update
RUN apt-get install -y wget

RUN mv /var/lib/dpkg/status /var/lib/dpkg/status.bak
RUN touch /var/lib/dpkg/status

RUN apt-get update

```

```
sbom/docker ❯ grep wget sbom.dpkg.json
sbom/docker ❯ 
```

Demo: Verifying `wget`

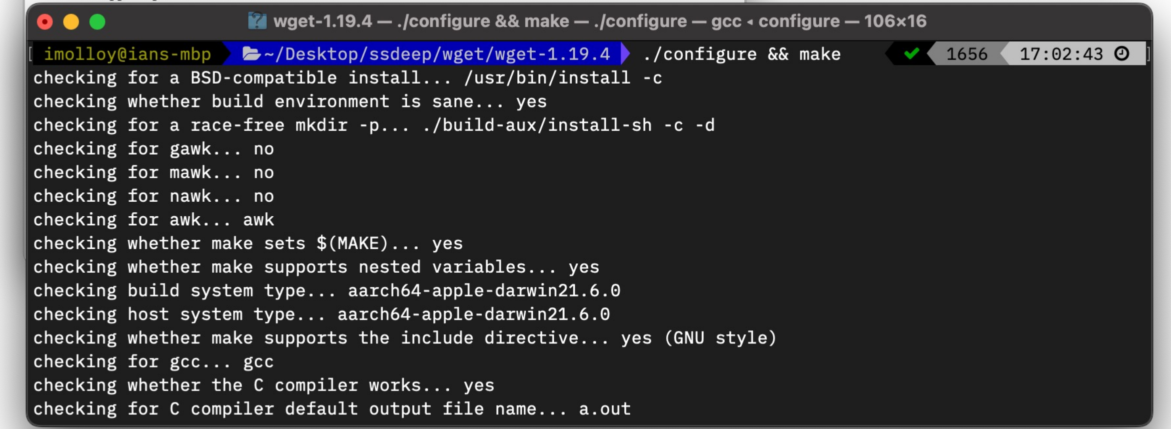


GNU Wget

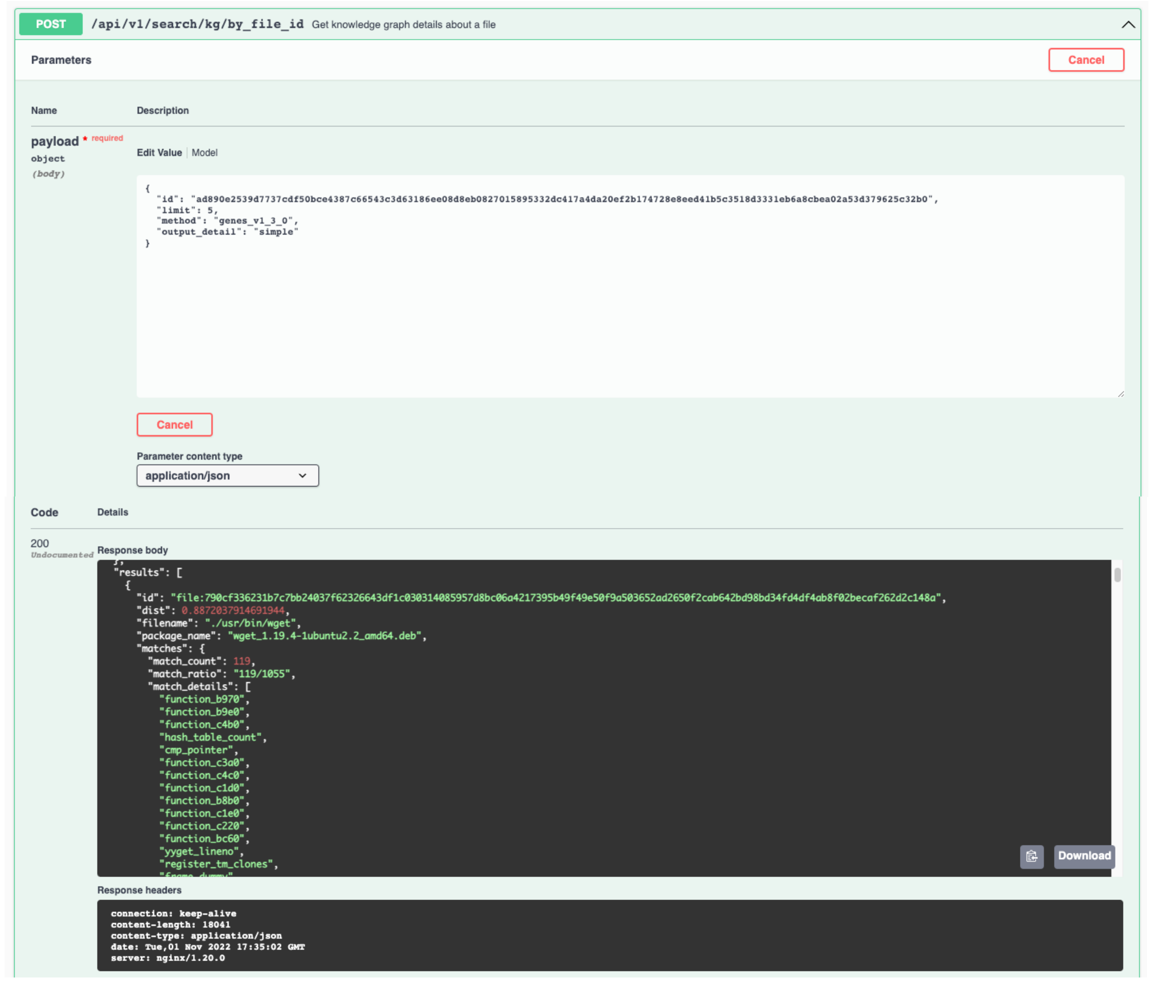
GNU Wget is a free software package for retrieving files using HTTP, HTTPS, FTP and FTPS, the most widely used Internet protocols. It is a non-interactive commandline tool, so it may easily be called from scripts, `cron` jobs, terminals without X-Windows support, etc.

GNU Wget has many features to make retrieving large files or mirroring entire web or FTP sites easy, including:

- Can resume aborted downloads, using `REST` and `RANGE`
- Can use filename wild cards and recursively mirror directories
- NLS-based message files for many different languages
- Optionally converts absolute links in downloaded documents to relative, so that downloaded documents may link to each other locally
- Runs on most UNIX-like operating systems as well as Microsoft Windows
- Supports HTTP proxies
- Supports HTTP cookies
- Supports persistent HTTP connections


wget-1.19.4 — ./configure && make — ./configure — gcc ◂ configure — 106×16

```
imolloy@ians-mbp  ~/Desktop/ssdeep/wget/wget-1.19.4  ./configure && make   ✔  1656  17:02:43
checking for a BSD-compatible install... /usr/bin/install -c
checking whether build environment is sane... yes
checking for a race-free mkdir -p... ./build-aux/install-sh -c -d
checking for gawk... no
checking for mawk... no
checking for nawk... no
checking for awk... awk
checking whether make sets $(MAKE)... yes
checking whether make supports nested variables... yes
checking build system type... aarch64-apple-darwin21.6.0
checking host system type... aarch64-apple-darwin21.6.0
checking whether make supports the include directive... yes (GNU style)
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
```

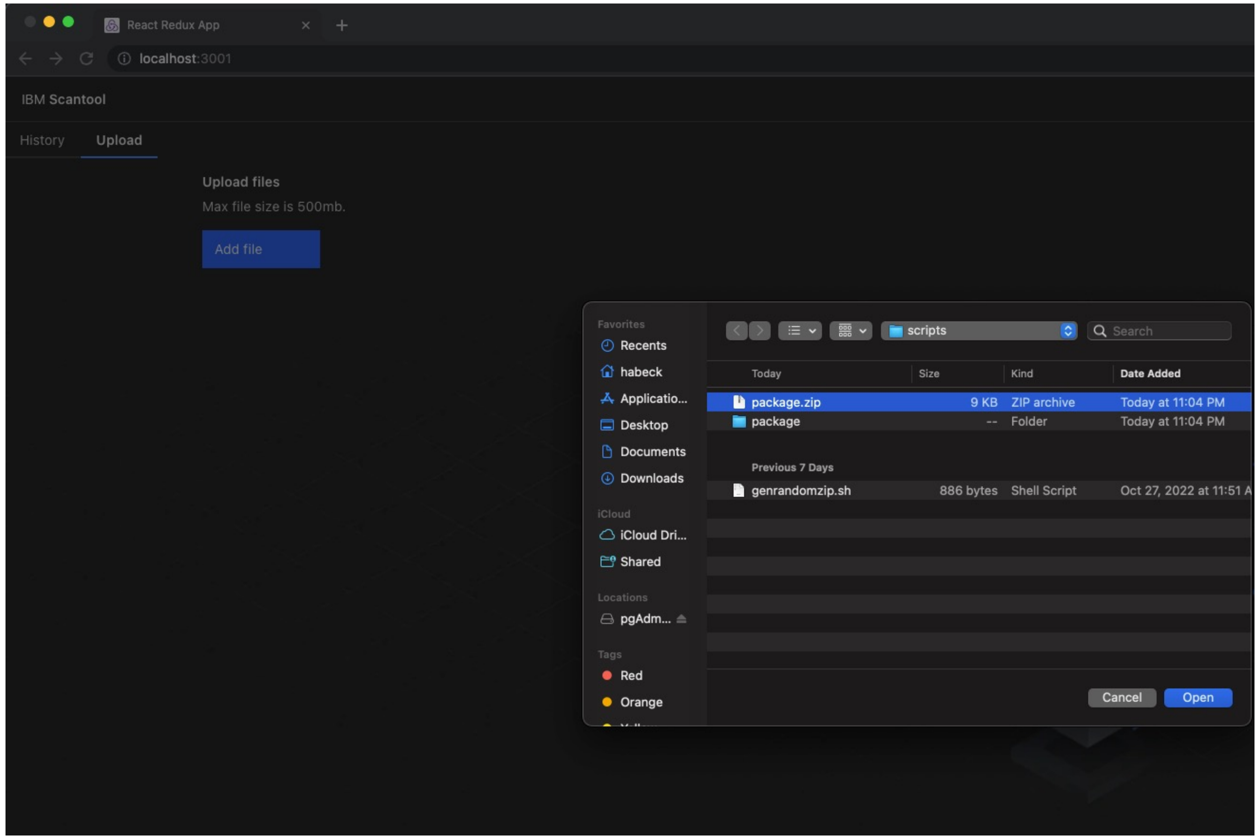


```
{
  "id": "ad890e2539d7737cdf50bce4387c66543c3d63186ee08d8eb0827015895332dc417a4da20ef2b174728e8eed41b5c3518d3331eb6a8cbea02a53d379625c32b0",
  "limit": 5,
  "method": "genes_v1_3_0",
  "output_detail": "simple"
}
```

```
  "results": [
    {
      "id": "file:790cf336231b7c7bb24037f62326643df1c030314085957d8bc06a4217395b49f49e50f9a503652ad2650f2cab642bd98bd34fd4df4ab8f02becaf262d2c148a",
      "dist": 0.8872037914691944,
      "filename": "./usr/bin/wget",
      "package_name": "wget_1.19.4-1ubuntu2.2_amd64.deb",
      "matches": {
        "match_count": 119,
        "match_ratio": "119/1055",
        "match_details": [
          "function_b970",
          "function_b9e0",
          "function_c4b0",
          "hash_table_count",
          "cmp_pointer",
          "function_c3a0",
          "function_c4c0",
          "function_c1d0",
          "function_b8b0",
          "function_c1e0",
          "function_c220",
          "function_bc60",
          "yyget_lineno",
          "register_tm_clones",
```

```
connection: keep-alive
content-length: 18041
content-type: application/json
date: Tue,01 Nov 2022 17:35:02 GMT
server: nginx/1.20.0
```

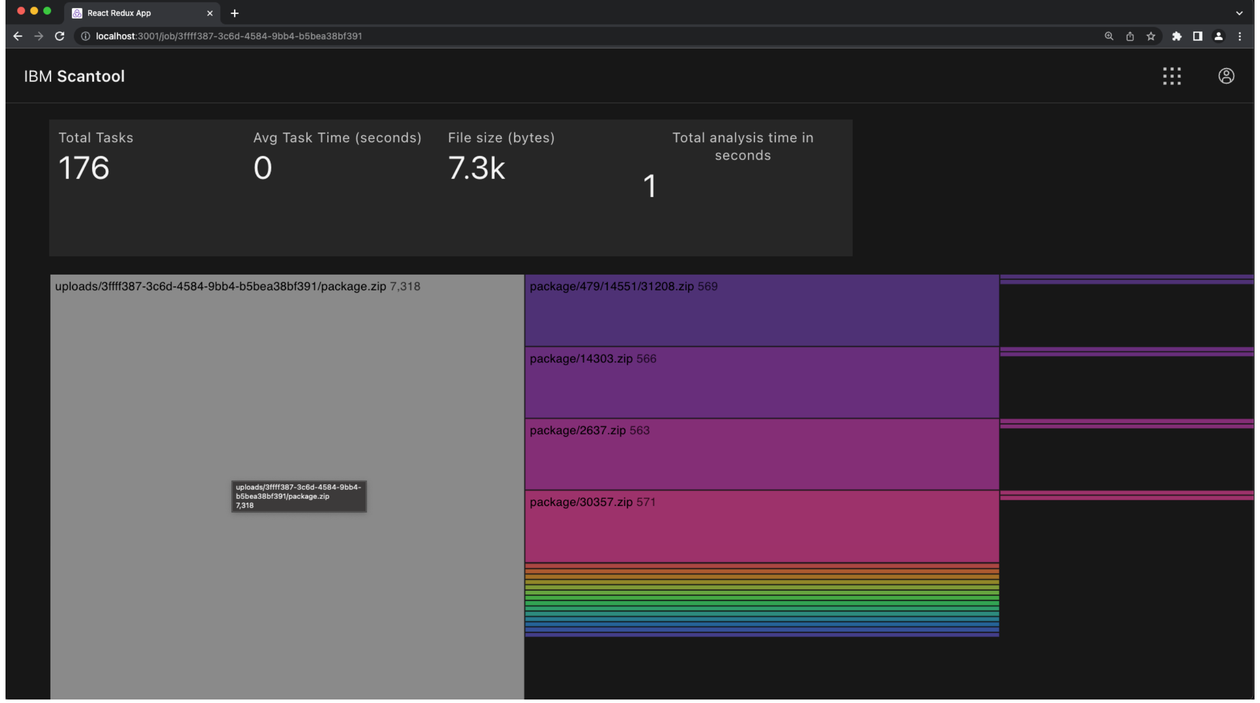
UI of POC



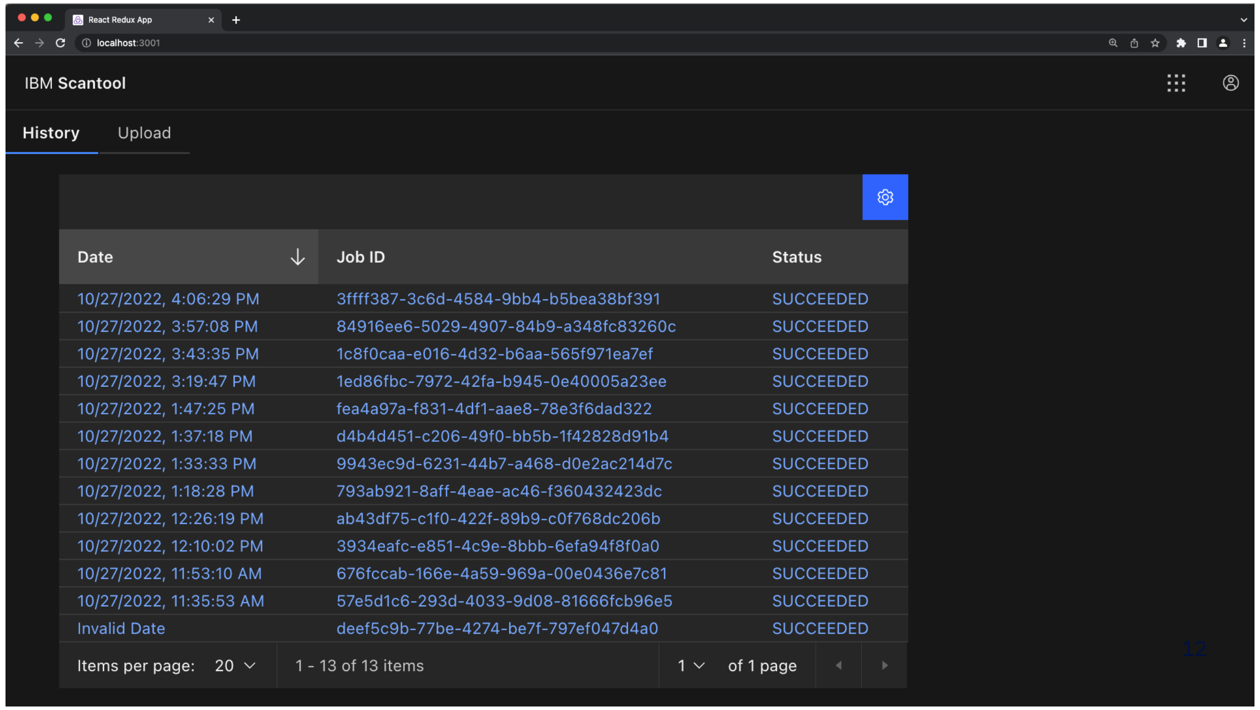
Upload files

Max file size is 500mb.

Today	Size	Kind	Date Added
package.zip	9 KB	ZIP archive	Today at 11:04 PM
package	--	Folder	Today at 11:04 PM
Previous 7 Days			
genrandomzip.sh	886 bytes	Shell Script	Oct 27, 2022 at 11:51 A


React Redux App
localhost:3001/job/3ffff387-3c6d-4584-9bb4-b5bea38bf391
IBM Scantool
uploads/3ffff387-3c6d-4584-9bb4-b5bea38bf391/package.zip 7,318
package/479/14551/31208.zip 569
package/14303.zip 566
package/2637.zip 563
package/30357.zip 571
uploads/3ffff387-3c6d-4584-9bb4-b5bea38bf391/package.zip
7,318

Total Tasks	Avg Task Time (seconds)	File size (bytes)	Total analysis time in seconds
176	0	7.3k	1



Date	Job ID	Status
10/27/2022, 4:06:29 PM	3ffff387-3c6d-4584-9bb4-b5bea38bf391	SUCCEEDED
10/27/2022, 3:57:08 PM	84916ee6-5029-4907-84b9-a348fc83260c	SUCCEEDED
10/27/2022, 3:43:35 PM	1c8f0caa-e016-4d32-b6aa-565f971ea7ef	SUCCEEDED
10/27/2022, 3:19:47 PM	1ed86fbc-7972-42fa-b945-0e40005a23ee	SUCCEEDED
10/27/2022, 1:47:25 PM	fea4a97a-f831-4df1-aae8-78e3f6dad322	SUCCEEDED
10/27/2022, 1:37:18 PM	d4b4d451-c206-49f0-bb5b-1f42828d91b4	SUCCEEDED
10/27/2022, 1:33:33 PM	9943ec9d-6231-44b7-a468-d0e2ac214d7c	SUCCEEDED
10/27/2022, 1:18:28 PM	793ab921-8aff-4eae-ac46-f360432423dc	SUCCEEDED
10/27/2022, 12:26:19 PM	ab43df75-c1f0-422f-89b9-c0f768dc206b	SUCCEEDED
10/27/2022, 12:10:02 PM	3934eafc-e851-4c9e-8bbb-6efa94f8f0a0	SUCCEEDED
10/27/2022, 11:53:10 AM	676fccab-166e-4a59-969a-00e0436e7c81	SUCCEEDED
10/27/2022, 11:35:53 AM	57e5d1c6-293d-4033-9d08-81666fcb96e5	SUCCEEDED
Invalid Date	deef5c9b-77be-4274-be7f-797ef047d4a0	SUCCEEDED

Items per page: 20 1 - 13 of 13 items 1 of 1 page

Status and Roadmap

Current Status

- Several techniques for computing genes
- Support for binaries, packages, bytecode
- Cloud native application / processing engine
- Currently performing large-scale evaluation
- Additional features and capabilities in development

Plans for Release:

- A version of genome generation
- Service demonstrating technology
- Utilities for handling genomes and querying service

Requests:

- Welcome feedback, support and collaboration
- Insights on capabilities and use cases
- How to complement existing OpenSSF projects