

License compliance with AI-assisted coding

KNOW YOUR FRANKIE

Julian Coccia, CTO





"Black and white sketch of a happy witch typing on a laptop" generated with DreamStudio.ai

“AI’s ban, like witchcraft in the Middle Ages, is fueled by fear”

Software Development Landscape

- Increasing adoption of Open Source
- Governments demanding SBOMs
- Proliferation of AI-assisted coding

90%

Forrester Wave™
Software Composition
Analysis, 2021



 GitHub Copilot

 tabnine

 Amazon CodeWhisperer

 OpenAI

“Third-party software integration is today almost involuntary”



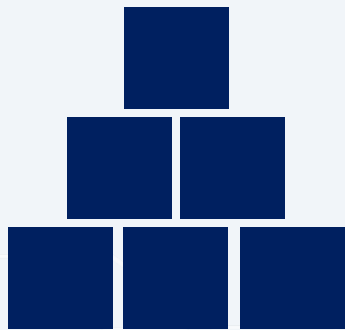
"Black and white sketch of a worried programmer", generated with DreamStudio.ai

“Developers who don’t use AI are being eclipsed by those who do”

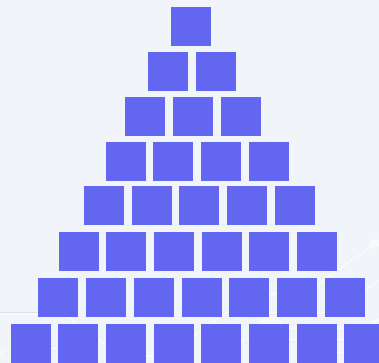
Source Code

vs

Human language

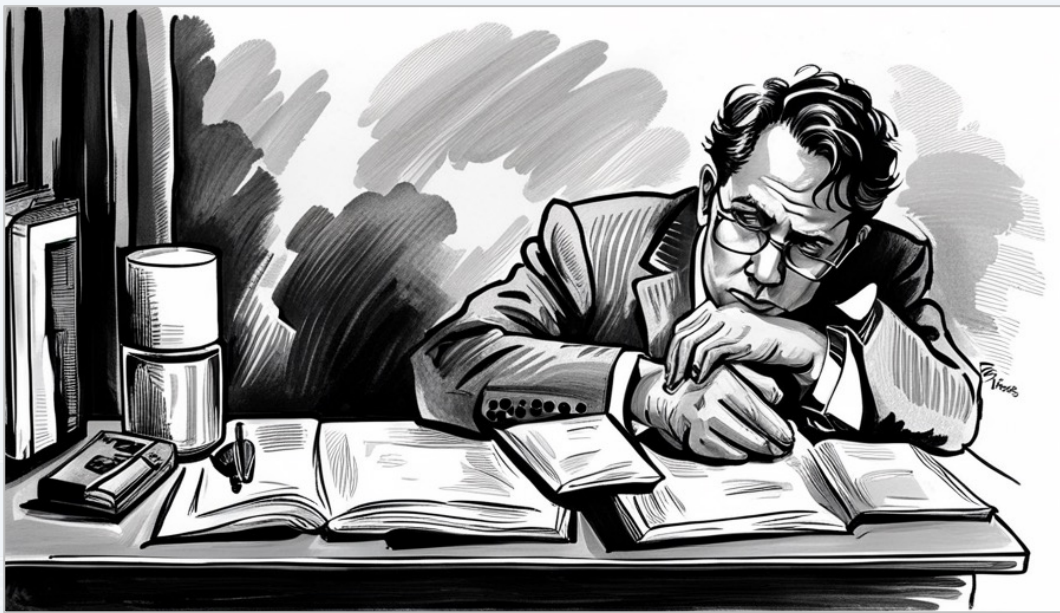


Limited set of instructions
Precise, unambiguous, and deterministic
Higher chance of AI generating existing material



Thousands of words and expressions
Ambiguous and context-dependent
Lower chance of AI generating existing material

“We enforce software licenses with laws made for human expression”



"Black and white sketch of a thinking writer", generated with DreamStudio.ai

“Computers learn from existing copyrighted code, just like humans”

Would courts relax towards AI generated code?

NO

“As judges remain human, Copyright Law and existing jurisprudence will hold sway”

Risks of unknown Software Composition

- License Compliance
- Trade Compliance (Export Control)
- Security Vulnerabilities
- Code quality
- Component health (obsolescence)
- Quantum-unsafe encryption
- Other risks

**“You can’t
comply with/protect
what you can’t see”**

The Gray Zone

Verbatim Copy

The Gray Zone

Mere inspiration

Verbatim Copy: Example



Winnowing algorithm in c

< 11/11 >



Here's an implementation of the Winnowing algorithm in C:



c

Copy code

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define BUFFER_RATE 4
#define GRAM_SIZE 30
#define WINDOW_SIZE 64

typedef unsigned int uint32_t;

uint32_t calc_crc32c(const char *data, size_t length)
{
    static uint32_t table[256];
    static uint32_t init = 0;
    if (!init) {
        init = 1;
    }
}
```

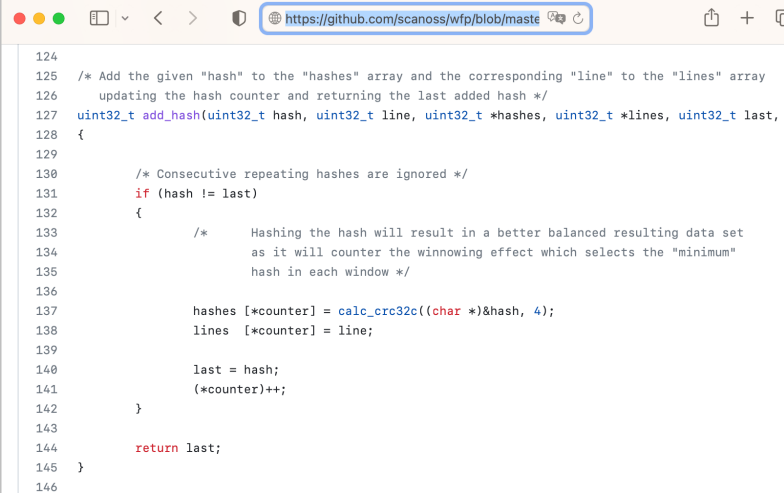
Verbatim Copy: Example

```
uint32_t add_hash(uint32_t hash, uint32_t line, uint32_t *hashes, uint32_t *
{
    /* Consecutive repeating hashes are ignored */
    if (hash != last) {
        /* Hashing the hash will result in a better balanced resulting data
        * as it will counter the winnowing effect which selects the "minimum
        * hash in each window */
        hashes[*counter] = calc_crc32c((char *)&hash, 4);
        lines[*counter] = line;
        last = hash;
        (*counter)++;
    }
    return last;
}

int main()
{
    char text[] = "This is an example text for testing the Winnowing algorit
winnow(text, strlen(text));
    return 0;
}
```

Note that this implementation uses a sliding window of size 64, a gram size of 30, and a buffer rate of 4. The `calc_crc32c` function is used to calculate the hash of each gram. The `winnow` function iterates over the text, calculating the hash of each gram and adding it to

```
"chatgpt_winnowing.c": [
{
  "id": "snippet",
  "oss_lines": "125-145",
  "file": "src/winnowing.c",
  "url": "https://github.com/scanoss/wfp",
  "purl": [
    "pkg:github/scanoss/wfp"
  ],
  "component": "wfp",
  "licenses": [
    {
      "name": "GPL-2.0-or-later",
      "copyleft": "yes",
      "osadl_updated": "2023-04-16T02:12:00+00:00",
      "patent_hints": "yes",
      "checklist_url": "https://www.osadl.org/fileadmin/checklists/unreflicenses/GPL-2.0-or-later.txt",
    }
  ]
}
```



```
124
125 /* Add the given "hash" to the "hashes" array and the corresponding "line" to the "lines" array
126     updating the hash counter and returning the last added hash */
127 uint32_t add_hash(uint32_t hash, uint32_t line, uint32_t *hashes, uint32_t *lines, uint32_t last,
128 {
129
130     /* Consecutive repeating hashes are ignored */
131     if (hash != last)
132     {
133         /* Hashing the hash will result in a better balanced resulting data set
134         as it will counter the winnowing effect which selects the "minimum"
135         hash in each window */
136
137         hashes [*counter] = calc_crc32c((char *)&hash, 4);
138         lines [*counter] = line;
139
140         last = hash;
141         (*counter)++;
142     }
143
144     return last;
145 }
146
```



“Black and white sketch of a detective checking a computer screen through a magnifying glass”, generated with DreamStudio.ai

“Identifying undeclared components is key to a complete SBOM”

Introducing SCANOSS

- Automated license compliance validation tool
- Integration with CI/CD pipelines
- Entirely Open Source
- Largest database of known Open Source
- Available for free: OSSKB.ORG

Software Composition Analysis

- Accurate, standardized detection of code plagiarism
- Adopted by Open Source communities
- Adopted by SCA suppliers
- Validated by European courts



Plagiarism check by detecting known OSS

```

urepath = ''
for basis in ( 'basis-link', 'basis', '' ):
    for ure in ( 'ure-link', 'ure', 'URE', '' ):
        if os.path.isfile(realpath(basepath, basis, ure, 'lib', 'unorc')):
            urepath = realpath(basepath, basis, ure)
            info(3, "Found %s in %s" % ('unorc', realpath(urepath, 'lib')))
            # Break the inner loop...
            break
        # Continue if the inner loop wasn't broken.
        else:
            continue
    # Inner loop was broken, break the outer.
    break

```



Your code fingerprint

```

file=24e35278ad5d4d3babe7379dc34d5bce,439,pasted.wfp
5=369450fc
6=bf7226a9
8=b04dd861
9=9727e3cd
11=2152ba16

```



```

{
  "snippet.py": [
    {
      "id": "snippet",
      "status": "pending",
      "lines": "1-10",
      "oss_lines": "165-174",
      "matched": "98%",
      "purl": [
        "pkg:github/unoconv/unoconv",
        "pkg:deb/unoconv",
        "pkg:pypi/unoconv"
      ],
      "vendor": "unoconv",
      "component": "unoconv",
      "version": "0.8.2",
      "latest": "0.8.2",
      "url": "https://github.com/unoconv/unoconv",
      "release_date": "2017-12-07",
      "file": "unoconv",
      "url_hash": "c36074c3996ba9d7d85f4a57787b5645",
      "file_hash": "0f55e083dcc72a11334eb1a77137e2c4",
      "source_hash": "aff32ef2847f81abc62da0769bfff43f",
      "file_url": "https://osskb.org/api/file_contents/0f55e083dcc72a11334eb1a77137e2c4",
      "licenses": [
        {
          "name": "GPL-2.0-only",
          "obligations": "https://www.osadl.org/fileadmin/checklists/unreflicenses/GPL-2.0-only.txt",
          "copyleft": "yes",
          "patent_hints": "yes",
          "incompatible_with": "Apache-1.0, Apache-1.1, Apache-2.0, BSD-4-Clause, BSD-4-Clause-UC, FTL, IJG, OpenSSL, Python-2.0, zlib-acknowledgement, XFree86-1.1",
          "source": "component_declared"
        }
      ]
    }
  ]
}

```

JSON response

Presence in public repositories

```
$ pip3 install scanoss  
$ scanoss-py scan mycode/
```

```
$ npm install -g scanoss  
$ scanoss-js scan mycode/
```




<https://github.com/scanoss>

First Multi-platform Auditing App

mycode > Reports Export

Detected Identified

Licenses



- GPL-2.0-only
- BSD-3-Clause
- Apache-2.0
- The Apache Software License; Version 2.0

Matches for license

Component	Vendor	Version
mzmine3	mzmine	Windows-latest

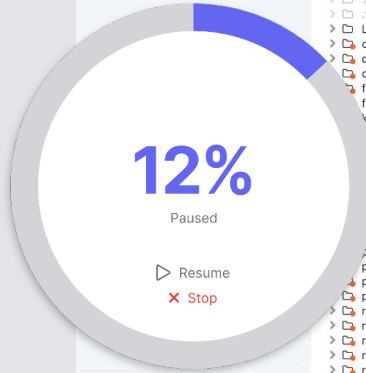
Matches

96% Match Scanned Files: 1783 4% No Match

Vulnerabilities

0 CRITICAL 0 HIGH 0 MODERATE 0 LOW

LICENSE	COPYLEFT	INCOMPATIBLE LICENSES
GPL-2.0-only	✓	Apache-1.0 Apache-1.1 Apache-2.0 BSD-4-Clause BSD-4-Clause-LIC FTL LGPL OpenSSL Python-2.0 silk-acknowledgement XFree86-1.1
BSD-3-Clause	✗	
Apache-2.0	✗	
The Apache Software License; Version 2.0	✗	
BSD	✗	
MIT	✗	



mycode > Detect

mycode

- github
- .reuse
- .travis
- LICENSES
- cli-scanner
- docker
- docs
- frontend-apps
- frontend-bugs
- kb-importer
- ubernetes
- ng
- g-java
- g-java-init
- g-java-reach
- g-java-reach-soot
- g-java-reach-wala
- ng-python
- patch-analyzer
- patch-lib-analyzer
- plugin-gradle
- plugin-maven
- repo-client
- rest-backend
- rest-lib-utils
- rest-lib-utils-init
- rest-nvd
- shared
- dockerignore
- .gitattributes
- .gitignore

9 versions steady 822

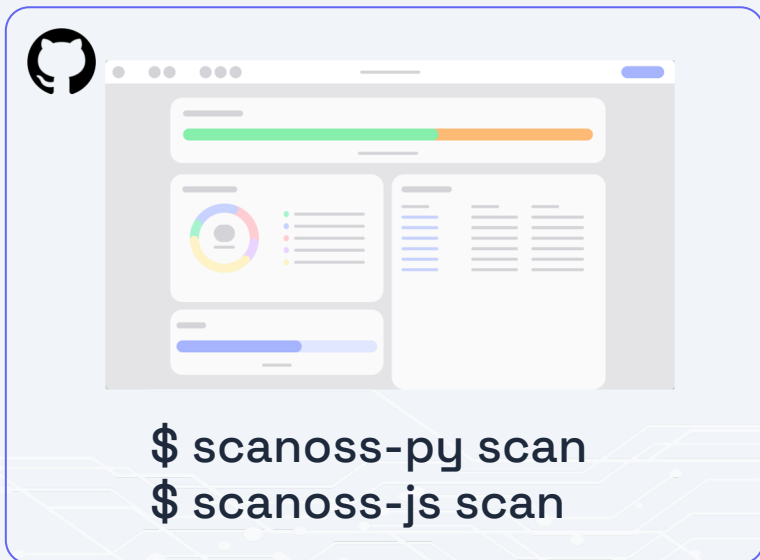
24 versions cxf 512

3 versions cxf-rt-ws-security 32

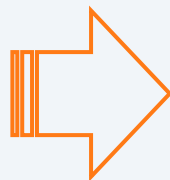
5 versions commons-fileupload 7



Public Knowledge Base API

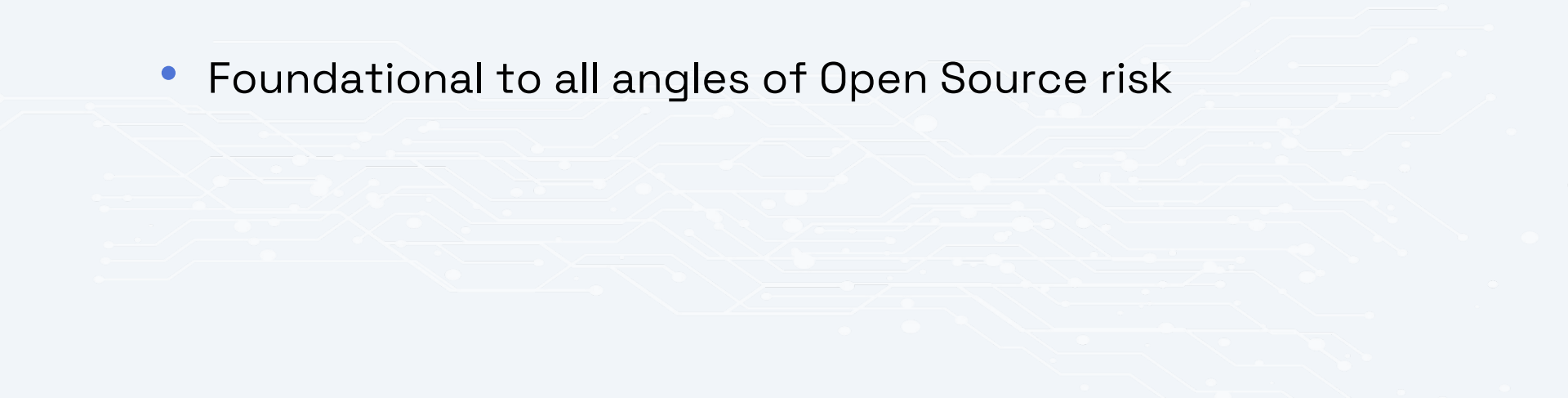


```
$ scanoss-py scan  
$ scanoss-js scan
```



<https://osskb.org/api>

Bottom line

- Embrace AI-assisted development
 - Validate your software composition with SCANOSS
 - Foundational to all angles of Open Source risk
- 
- A decorative graphic at the bottom of the slide consisting of a complex network of white lines and dots on a light blue background, resembling a circuit board or a data network.



Thank you!

Julian Coccia

julian.coccia@scanoss.com

<https://scanoss.com>