
SHIFTLEFT
Maintainer insight | Risk Quantification

CPARTA

- Developing and enabling a Swedish Commercial Cyber Defense
- Protect Swedish Critical Infrastructure and Critical Industry
- Development of Strategic Initiatives and Capabilities

About Me

- Strategic Coordinator Cparta
- Team Capt. Swedish National Hacking Team
- Vulnerability Research Margin Research

Analysis of motivation and intentions

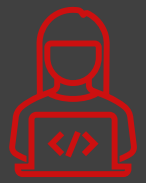
Quality assurance and compliancy

Weakest link and risk assessment

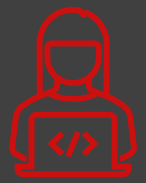
Analysis of motivation and intentions

Contributor metadata analysis & code analysis

Contributor metadata analysis



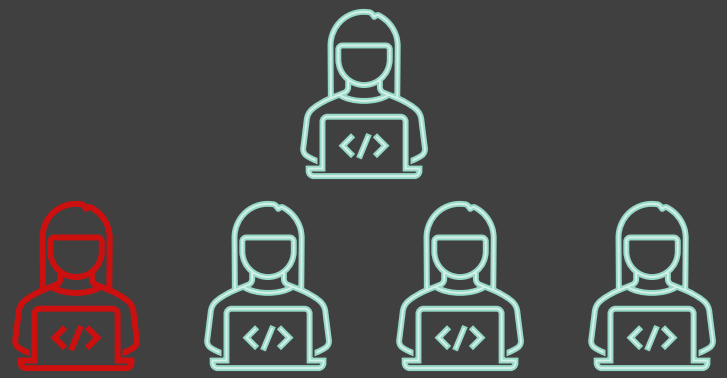
Contributor metadata
analysis & code analysis



Contributor metadata analysis & code analysis

By which metrics do we quantify code quality?

SHIFTLEFT Maintainer
overview of contributors



9a640e3 Pillow / libImaging / Convert.c ↑ Top

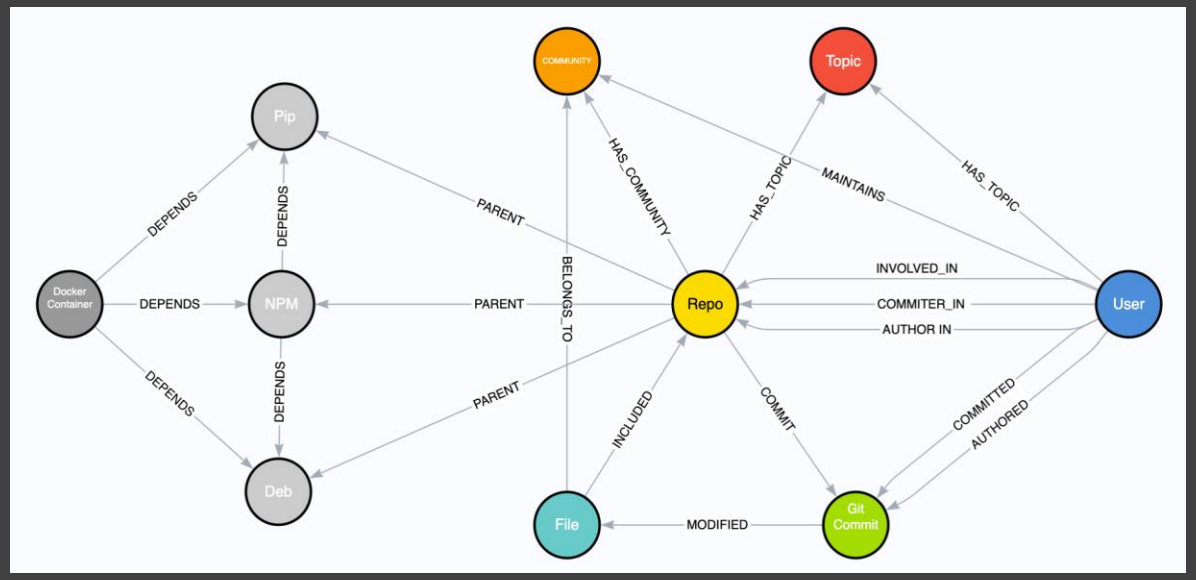
Code Blame 1132 lines (961 loc) · 26.1 KB Raw

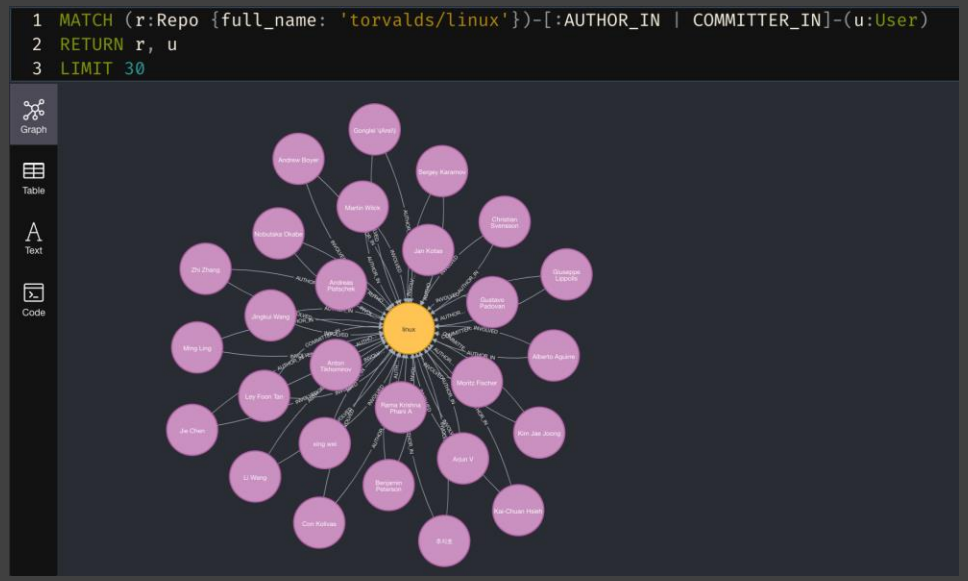
```
1069                break;
1070            }
1071
1072        if (!convert)
1073    #ifdef notdef
1074            return (Imaging) ImagingError_ValueError("conversion not supported");
1075    #else
1076        {
1077          static char buf[256];
1078          /* FIXME: may overflow if mode is too large */
1079          sprintf(buf, "conversion from %s to %s not supported", imIn->mode, mode);
1080          return (Imaging) ImagingError_ValueError(buf);
1081        }
1082    #endif
1083
1084        imOut = ImagingNew2(mode, imOut, imIn);
1085        if (!imOut)
1086            return NULL;
1087
1088        ImagingSectionEnter(&cookie);
1089        for (y = 0; y < imIn->ysize; y++)
1090            (*convert)((UINT8*) imOut->image[y], (UINT8*) imIn->image[y],
1091                       imIn->xsize);
1092        ImagingSectionLeave(&cookie);
1093
1094        return imOut;
```

Launchpad

C code from 1997
With critical FIXMEs

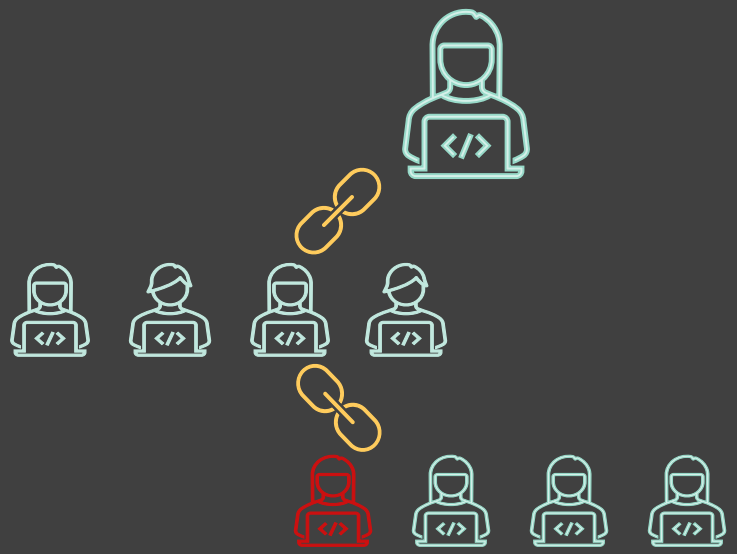
Reagent – Margin Research



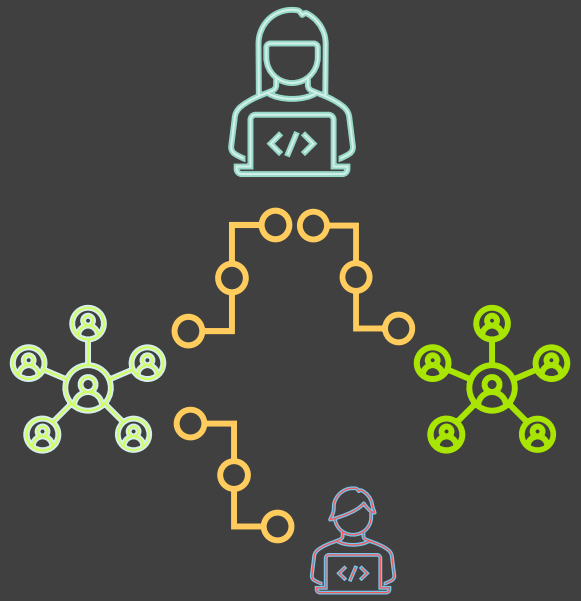


CPARTA™
CYBER DEFENSE_

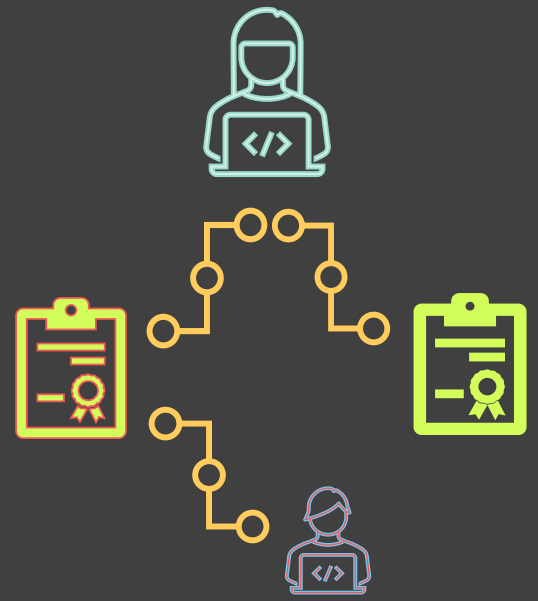
SHIFTLEFT Maintainer
overview of dependencies



SHIFTLEFT Maintainer
Weakest Link Risk identification



SHIFTLEFT Maintainer Continuous Quality & Risk Report



Metrics of "good" code &
Metrics of "trusted" contributor

- Issues generated / LoC (vulnerability does not require malicious intent)
- AI code quality assessment (Lack of maintenance)
- Graph based contribution analysis (Behaviour & Intent)

Metrics of "good" code &
Metrics of "trusted" contributor

- Bug transmission project-to-project adoption
  - e.g. Glibc stealing Musl code with bugs in them
- AI analysis of "Does this commit only solve the issue?"
  - Hidden introductions of features / vulnerabilties
- Analysis of "Does this code alter the scope of access"
  - Accessing new resources

```
-*- Read comments from end of lifetime to start of lifetime. -*-

// Usage implies proof is necessary: assert N <= direction <= E
void foobar(enum Direction direction) {
 switch (direction) {
  case N: ...
  case W: ...
  case E: ...
  // switch case has holes in enumeration coverage. Proof state: assert N <= direction <= E
  }
  // End of lifetime of direction.
}

void karbar() {
 // Bug is found since there exists no proof for assert N <= direction
 // end of life for direction, sink point in syscall assumes no bounding

 enum Direction direction = … // user input

 // Proof state: assert N <= direction

 if (direction <= E) { System.exit(0); }
 // Code path not reachable for direction > E. Satisfies iff assert N <= direction

 foobar(direction) // on call-site needs to prove: “assert N <= direction <= E”
}
```

Lean proof state – challenge to communicate to maintainer. Needs to be intuitive

```
n : ℕ
s : ℕ → ℝ
h1 : n > 2
h2 : attainable n s
h1' : 2 < ↑n
⊢ 0 < ↑n - 3

▼ Messages (1)
▼ prev_bound.lean:222:6

linarith failed to find a contradiction
▼ case h
n : ℕ
s : ℕ → ℝ
h1 : n > 2
h2 : attainable n s
h1' : 2 < ↑n
a† : 0 ≥ ↑n - 3
⊢ False
```

- Imperial College London teaching lean for formal proofs
- Integrating formal proofs in to language design
  - In an intuitive way using AlphaProof et.c
- Theorem proving is already a part of industry
  - PLC industrial systems
  - Proof of soundness for cryptocurrencies (Zellic et al.)

Questions?