"SBOMs Are No Longer Mandatory Which Is a Good Thing" and Other Opinionated Software Supply Chain Observations

Justin Cappos
New York University

Slides link ->
Will be on the last slide too!

Who am I?

“Mid Career” Academic

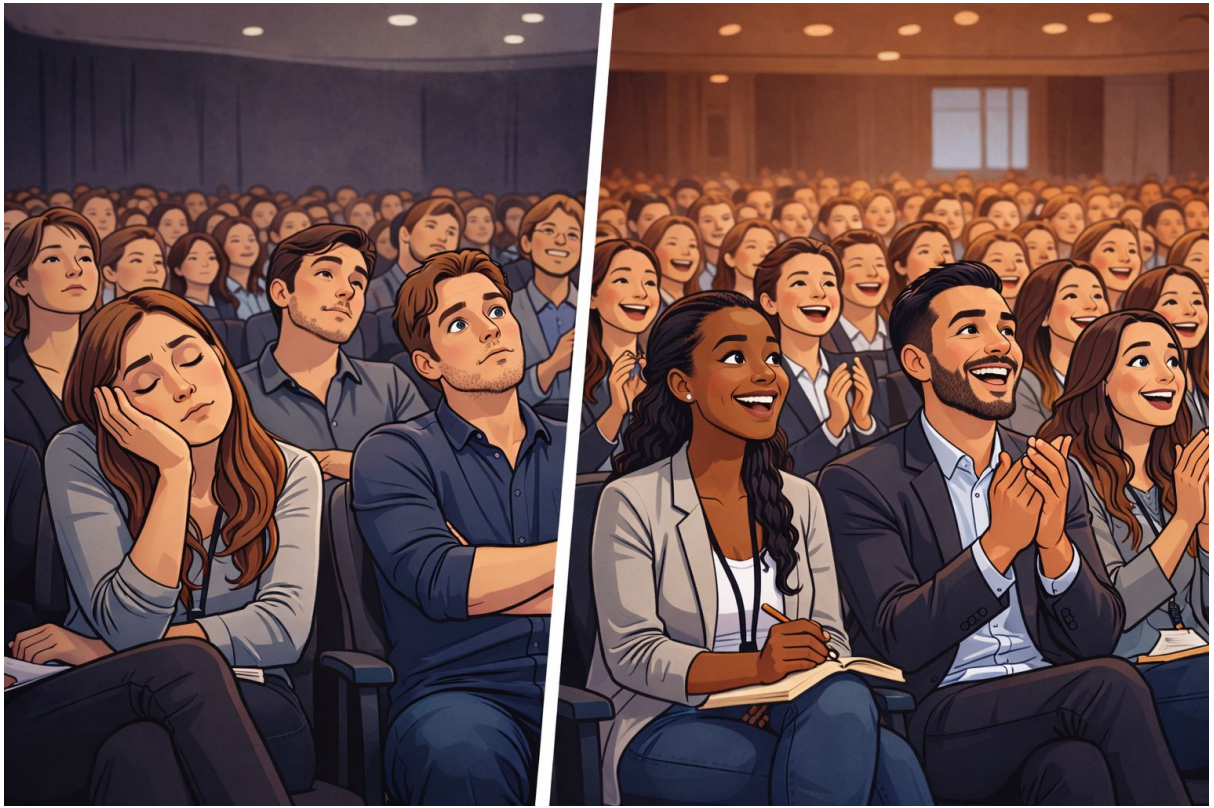
Sometimes called “Father of Software Supply Chain Security”

A creator of TUF, Uptane, in-toto, gittuf, SBOMit, etc.

Architecture in other widely used parts of the stack (package managers, Git, popular language ecosystems, cars, IoT devices, legal systems, etc.)

Work in open source -- Linux Foundation (security assessment process, CNCF Tech Lead, OpenSSF Academic Outreach co-chair, etc.)

Keynotes



Other notes...

Problem setup > solution detail

    You may have a better solution to some problem I raise. Try it!

Feel free to raise objections during the talk

⚠ CONTENT WARNING

EXTREME OPINIONS

This speaker may exaggerate their feelings on topics and take things beyond their logical conclusion in order to make this talk more entertaining and engaging.

Security Will Improve Now SBOMs Are Not Mandated

Security Will Improve Now SBOMs Are Not Mandated

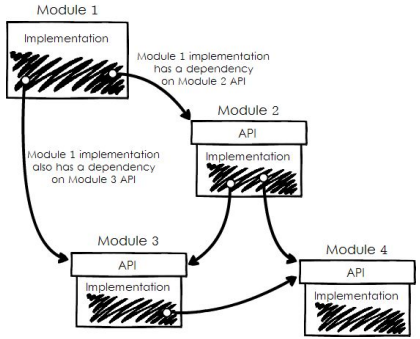
- SBOMs are wildly inaccurate
  - agreement rates as low as ~8–13% in large-scale evaluations [Wang, et al]

- Inaccurate SBOMs are often worse than not having one
  - False positive rate on scanners ~97.5% [Zhou, et al]
  - High false negative rate as well (which is even worse)

- SBOMs are mostly garbage, unless you want someone to blame...

SBOM: Your Software's Ingredient Label

Software



When should you get the ingredients list?

 **When you have the recipe**

 **When you're baking the pie**

 **After the pie is baked**

SBOMs After the Pie Is Baked?

🥧 Today's SBOMs Often Lie to You

- 🔧 Many created after — “It’s like labeling a pie after baking… based on smell.”
- 📄 Based on static analysis or dependency files
- 🔍 Can miss runtime artifacts or transitive dependencies
- ❗ Common approach, but inaccurate in practice

SBOMs Before the Pie Is Baked?

SBOMs from a recipe are also inaccurate

- 🔧 Created before you build— "It's like labeling a pie based upon the shopping list."
- 📄 Misses things already in the build system, etc.
- 🔍 Misses things that might come in via build scripts, etc.
- ❗ Inaccurate, but misses different things than building it after

Make SBOMs in the Kitchen!

🔲 Build time SBOMs are the best hope for accuracy

- 🔧 This is where the action is!
- 📄 You see what is used as it is used
- 🔍 All of the dependency resolution / build system info is here!
- ❗ But there are tons of different language toolchains

How should you collect build info?

Option 1: instrument SBOM tooling for each language toolchain / each weird behavior

❗ Holes keep appearing

Do they ever stop coming?

How should you collect build info?

Option 2: Capture all information that happens during build

Language / toolchain agnostic

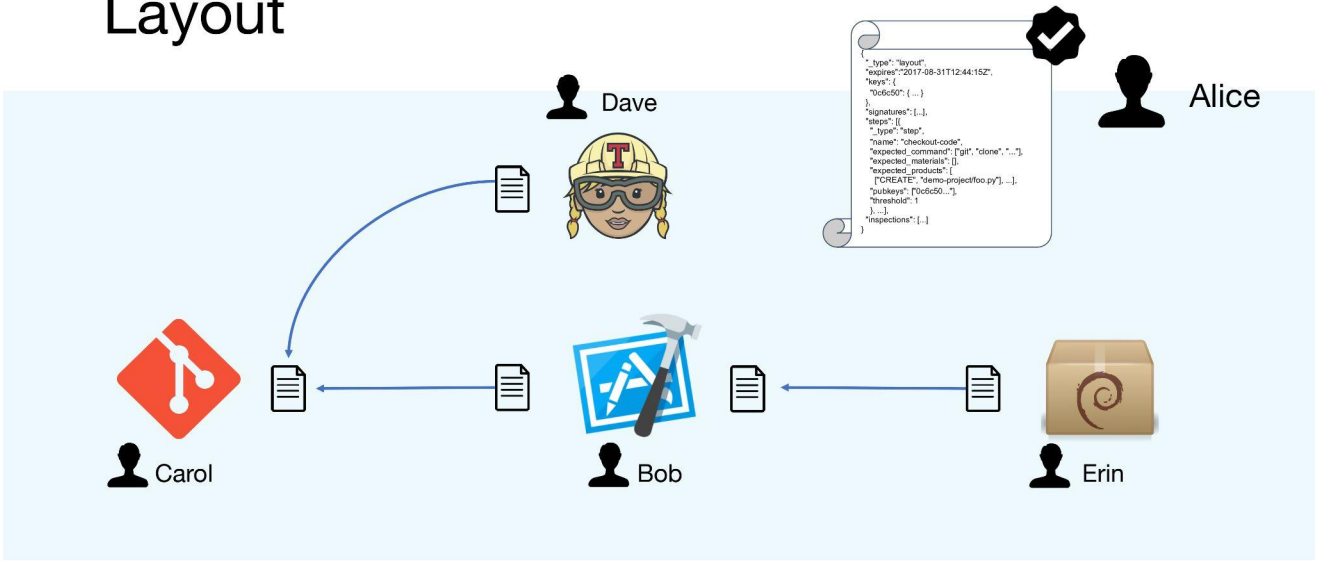
But you still have to identify what you see

(Already a problem in SBOMs today)

Our pitch: Making SBOMs Honest with SBOMit

SBOMit = SBOM + in-toto (A kitchen camera 📸 records every step of baking the pie.)

Layout



Security Will Improve Now SBOMs Are Not Mandated

- Removing SBOMs from compliance means advocates must prove worth

- Push toward better practices for understanding what is in software
  - Justin’s claims best approach: Build time, tool agnostic attestations -> SBOM information

Software Updates Are The Most Important Thing Today

Software Updates Are The Most Important Thing Today

- Old software == vulnerable software
  - Turns known vulns into “0 days”
- Someone needs to apply patches
  - You can do this or you pay someone to do this
  - If you pay for AV, you’re a sucker

Software Updates Are The Most Important Thing Today

- You can pay for a vendor to curate and apply patches
  - Diminishing return on patches
    - No patching -> some patching is huge value add
    - Some patching -> great patching is a minimal value add
    - Only a few vulnerabilities are actually exploited
      - Patch latency is important, but is reasonable for exploited code

Software Updates Are The Most Important Thing Today

- You can pay for a vendor to curate and apply patches
  - Diminishing return on patches
    - No patching -> some patching is huge value add
    - Some patching -> great patching is a minimal value add
    - Only a few vulnerabilities are actually exploited
      - Patch latency is important, but is reasonable for exploited code
  - Does reducing “bloat” really matter?
    - Maybe for compliance + efficiency
    - Does it do anything for security?
      - Maybe ROP gadgets are easier to find, but are they that hard in general?

Compartmentalization Will Be the Most Important Thing Tomorrow

-

Compartmentalization Will Be the Most Important Thing Tomorrow

- Defender must be perfect
- Lots of attack surface
  - AI makes it easy to sweep this
- There will be laggards
  - Some sectors: Consumer IoT
  - Some libraries
    - >95% of software has some open source
    - ~20-40% of software is unmaintained

Compartmentalization Will Be the Most Important Thing Tomorrow

- Cannot patch everything
  - Rewrite it all (with AI)?
    - Tooling isn’t there yet

Expected damage ~= probability * impact

- If we can’t push on probability (patching), reduce impact.

Compartmentalization Will Be the Most Important Thing Tomorrow

Compartmentalization is the future of security

Reduce scope of compromises / attacks

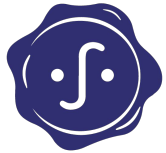
Lots of promising work across the stack

- Operating systems (namespaces, cgroups, etc.)
- Library OS work (Lind, Semisolates, etc.)
- Hardware (MPK, CHERI, etc.)
- SFI / runtimes (WebAssembly, LFI, etc.)
- Programming language techniques (need help!)

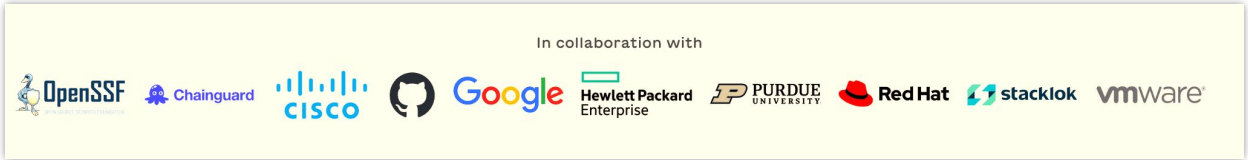
Group working on this: Open Robust Compartmentalization Alliance (ORCA)

Sigstore Doesn’t Do Much To Improve Security...

sigstore



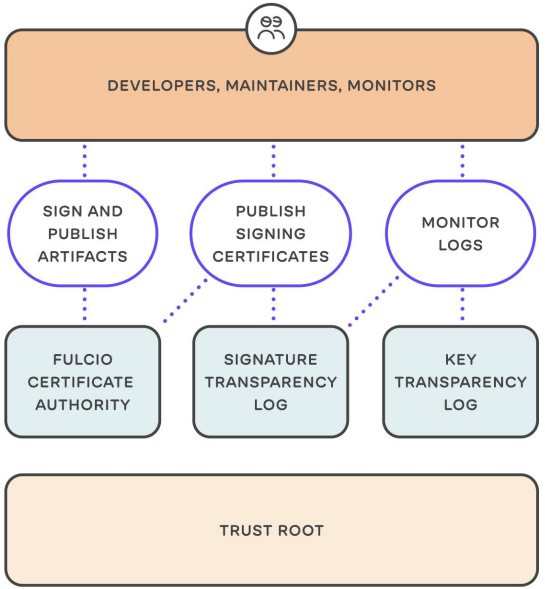
- Project officially launched by the Linux Foundation in 2021, development done in collaboration with many organizations
- Sigstore is a set of tools for developers, software maintainers, package managers and security experts
- Handles digital signing, verification and checks for provenance needed to make it safer to distribute and use open source software.
- An easy way for a community repo to outsource signing



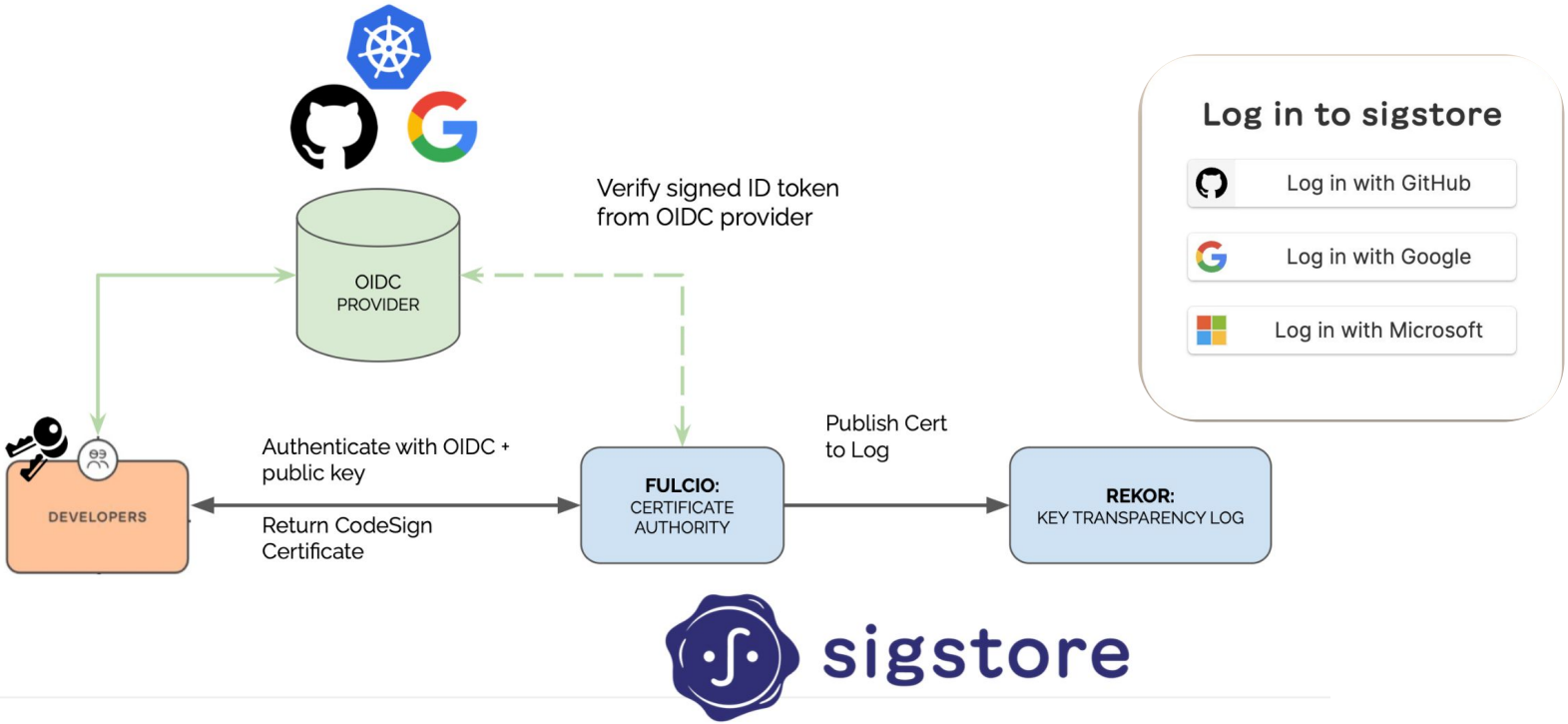
https://www.sigstore.dev/

How does sigstore work?

- Cosign
  - Tool for signing/verifying containers (and other artifacts) that ties the rest of Sigstore together, making signatures invisible infrastructure. Other tools also exist for the same purpose.
- Fulcio
  - A free root certification authority, issuing temporary certificates to an authorized identity and publishing them in the Rekor transparency log.
- Rekor
  - A built-in transparency and timestamping service, Rekor records signed metadata to a ledger that can be searched, but can’t be tampered with.



Keyless (ephemeral key) signing

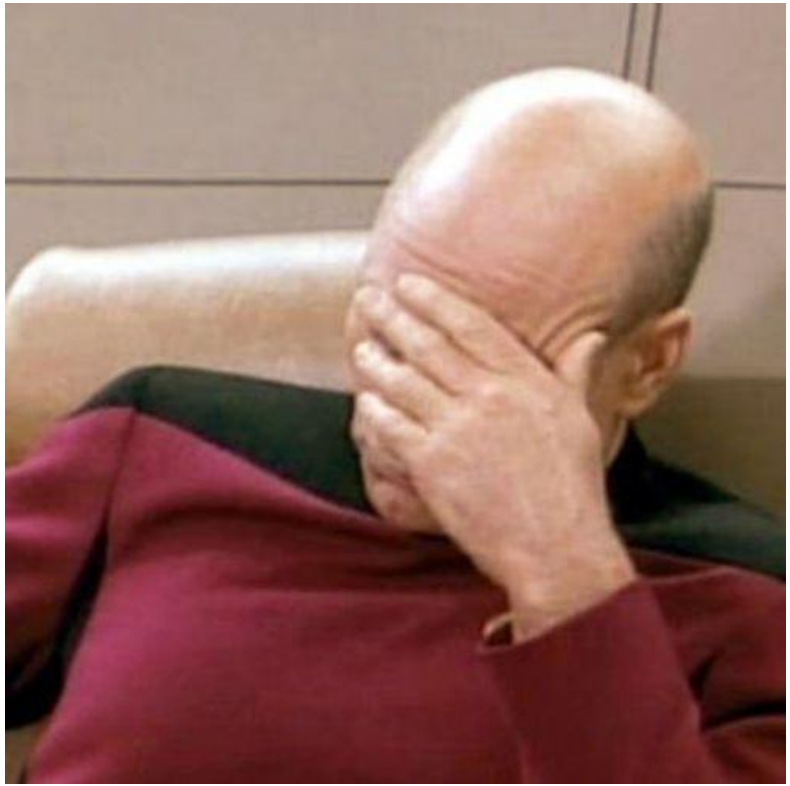


But which OIDC / key goes with a project?

- Ask the repo

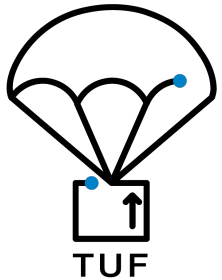
But which OIDC / key goes with a project?

- Ask the repo



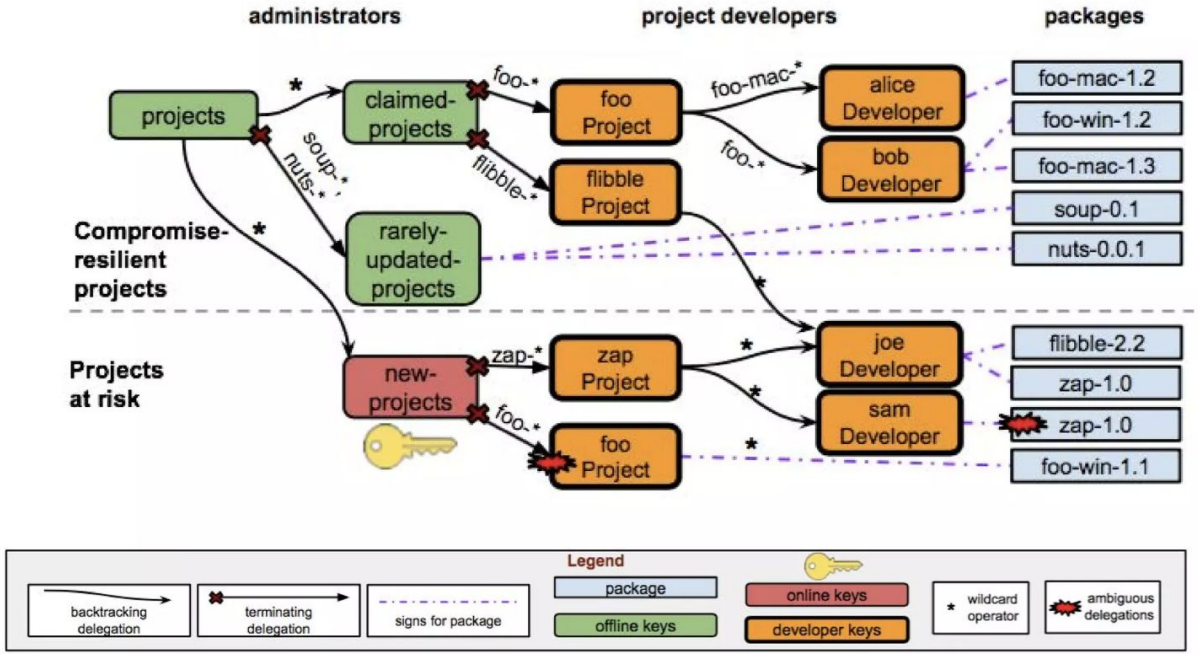
Sigstore Doesn’t Do Much To Improve Security...

- Namespace delegation is the way to fix this
  - Selective
  - Prioritized
  - Terminating

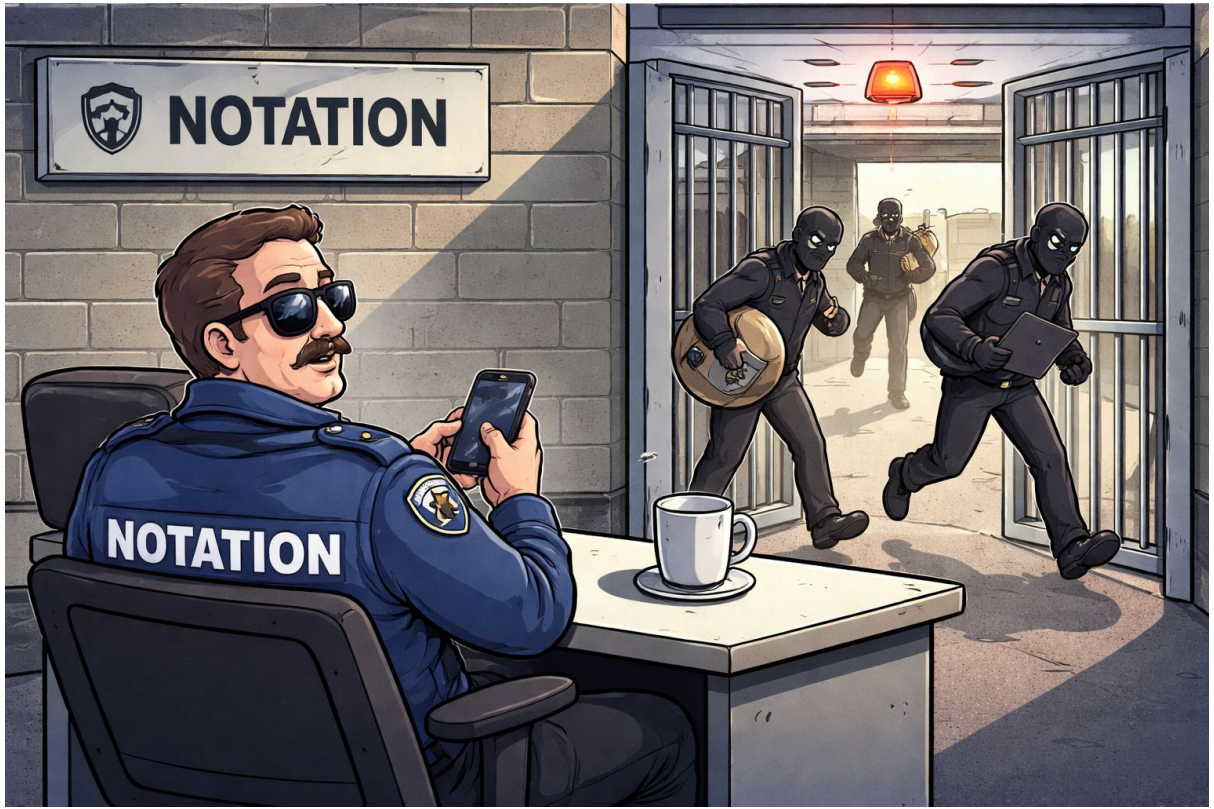


https://theupdateframework.io/

Target (Project) Delegation for PyPI (PEP 480)



Notation Is Fairly Useless From a Security Standpoint



Types of signatures — What you sign matters

- **Transit:** Only signed during transfer

- **Embedded:** Adds signature into the package itself

- **Detached Separate:** Have a detached signature for each package

- **Detached Combined:** Collect detached signatures for each signer

Types of signatures — What you sign matters

- **Transit:** Only signed during transfer

- **Embedded:** Adds signature into the package itself

- **Detached Separate:** Have a detached signature for each package

- **Detached Combined:** Collect detached signatures for each signer

Detached Separate Package Signatures

- Signature is a separate file. It contains a hash of the contents of the package. There is one file per signature, per package

- Questions:
  - How do you know who signed a package?
  - How do you revoke a signature?
  - How do you know there should be a signature?

Detached Separate Package Signatures

- How do you know who signed a package?
  - The repository / mirror needs to give you the detached signature

❌ There isn't any good reason to trust the mirror / repo, so they can lie. Also, you could accidentally have a signature be omitted (due to upload timing). Lots of potential for attack here.

Detached Separate Package Signatures

- How do you revoke a signature? 🤮
  - Not covered by the packaging system

- In practice, rely on GPG, OCSP, CRLs, pushing out an update to update the keys, etc., which don't work well.

You still have a huge mess here... Outside parties / parties with little trust become a synchronization point

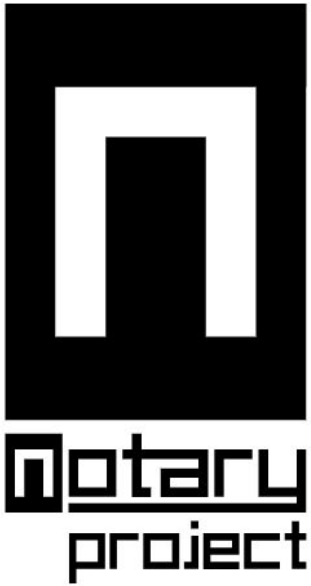
No way to revoke a single signature

Detached Separate Example: Notation / Notary

Developed by Microsoft, Amazon, etc.
CNCF Incubating Level project

Used to distribute container images, etc.

- Items requested by tag
  - Tags can be mutable (e.g. latest) or not
- Optional signatures associated with a tag
- Optional in-toto attestations with a tag
- Signatures expire after some time (e.g., 1 year)



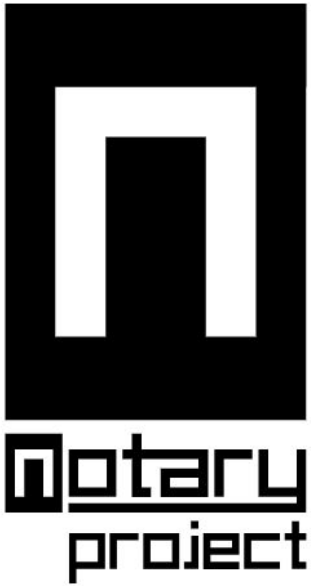
Detached Separate Example: Notation / Notary

Questions to discuss:

- How do you know who signed a package? ❌
- How do you revoke a signature? 🤮
- How do you know if there should be a signature / attestation? ❌

Replay attacks for reused tags are a weakness

- Provides good logging / compliance, but trusts registry far too much (harms security)



Detached Combined Package Signatures

- Signature is a separate file. It contains a hash of the contents of the package. There is one file per user.

- Questions:
  - How do you know who signed a package?
  - How do you revoke a signature?
  - How do you know there should be a signature?

Detached Combined Package Signatures

- How do you know who signed a package?
  - The repository / mirror needs to give you the detached combined signature

✅ But, it is common to globally track* signers' detached metadata and to retrieve detached user metadata for all signers you trust

Namespacing is needed to handle this

* Can be version numbers, full contents, etc. (depending on size)

Detached Combined Signatures

- How do you revoke a signature?
  - Get an updated file for each signer indicating what they trust

✅ Makes the repository the synchronization point for revocation. A new signature file both new signatures and revocation. Provides a high degree of security

Detached Combined Signatures

- How do you know there should be a signature?

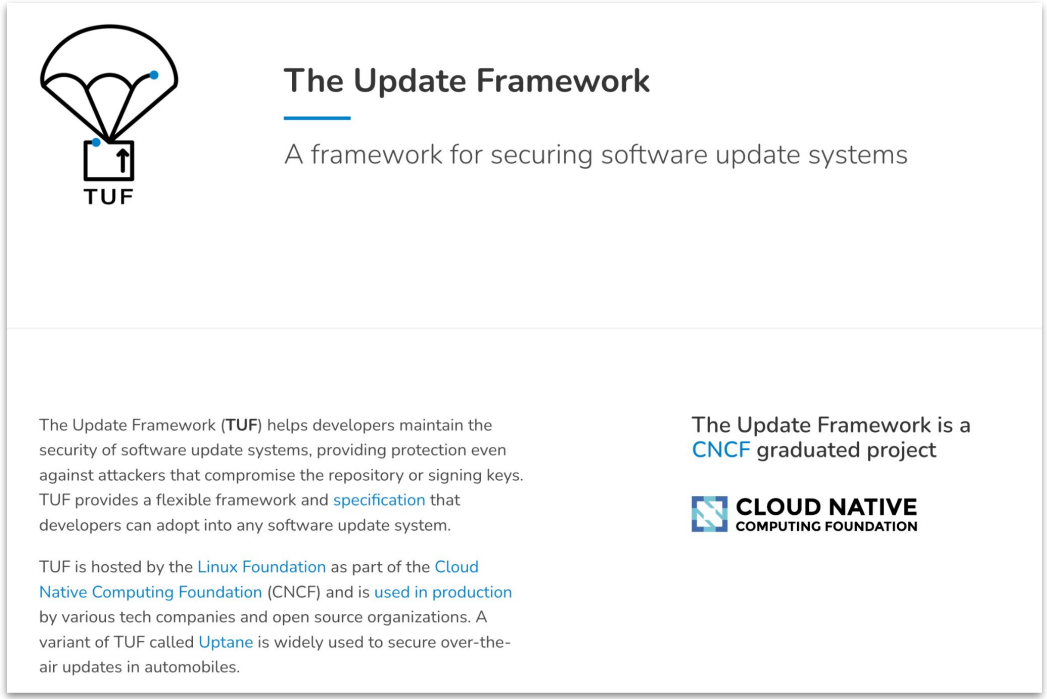
✅ Usually combined with **namespaces** (discussed before)

Indicates which key(s) apply to which packages

A framework for fixing these issues

- You end up with →

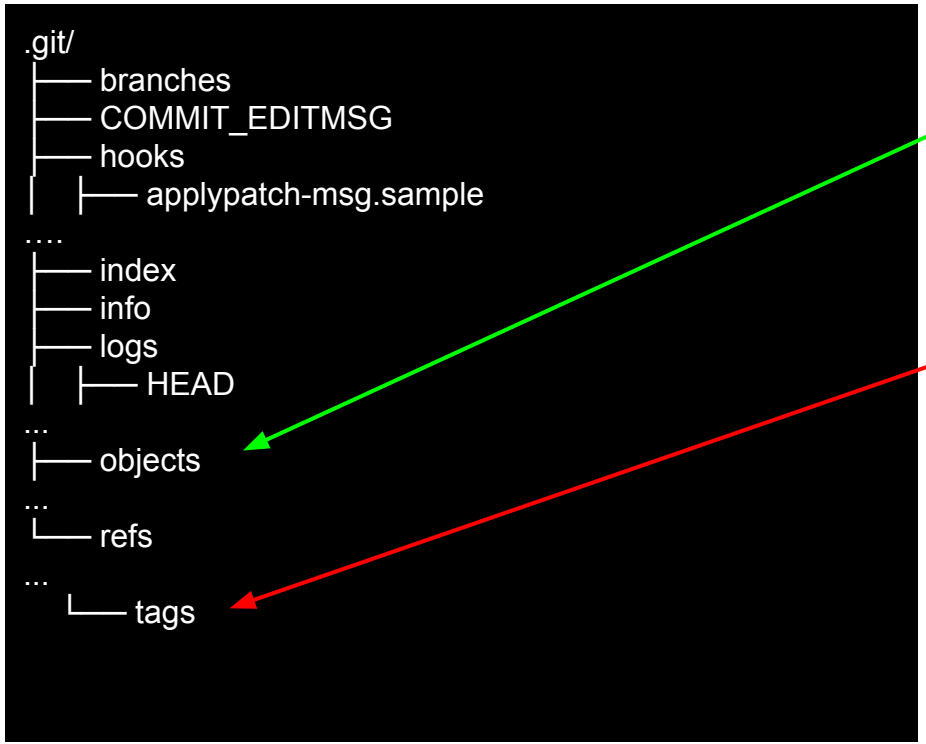
https://theupdateframework.io/



Signing Git Commits Does Very Little

Signing Git Commits Does Very Little: Why?

- Simply put, some Git metadata is not signed



Signed!

Not signed

References, pointers to Git tags + commits, are **not** signed
Anyone write access to the repository can modify.
The resulting attack looks like regular git operation.

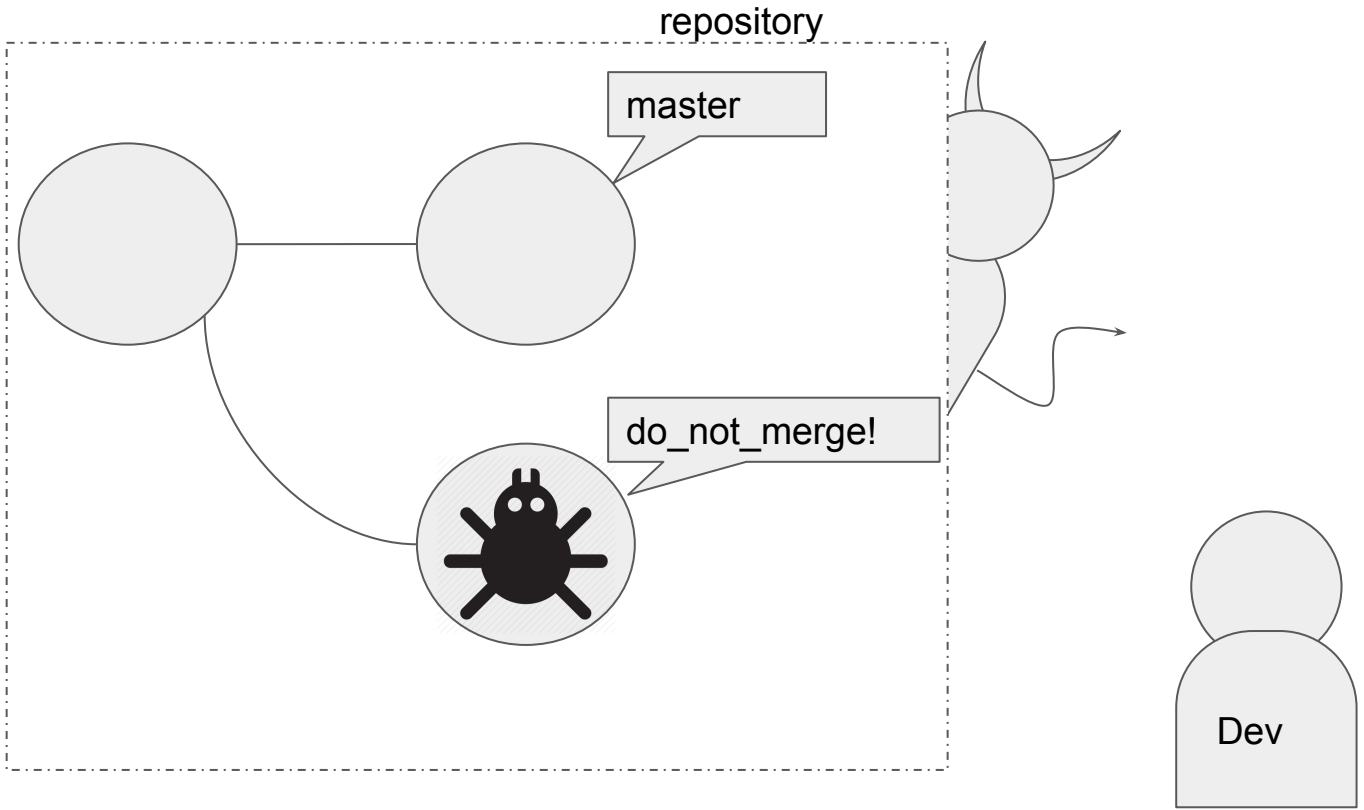
Attack taxonomy

- Teleport Attacks
  - Branch Teleport Attack
  - Tag Teleport Attack
- Rollback Attacks
  - Branch Rollback Attack
  - Global Rollback Attack
  - Effort Duplication Attack
- Deletion Attacks
  - Branch Deletion Attack
  - Tag Deletion Attack

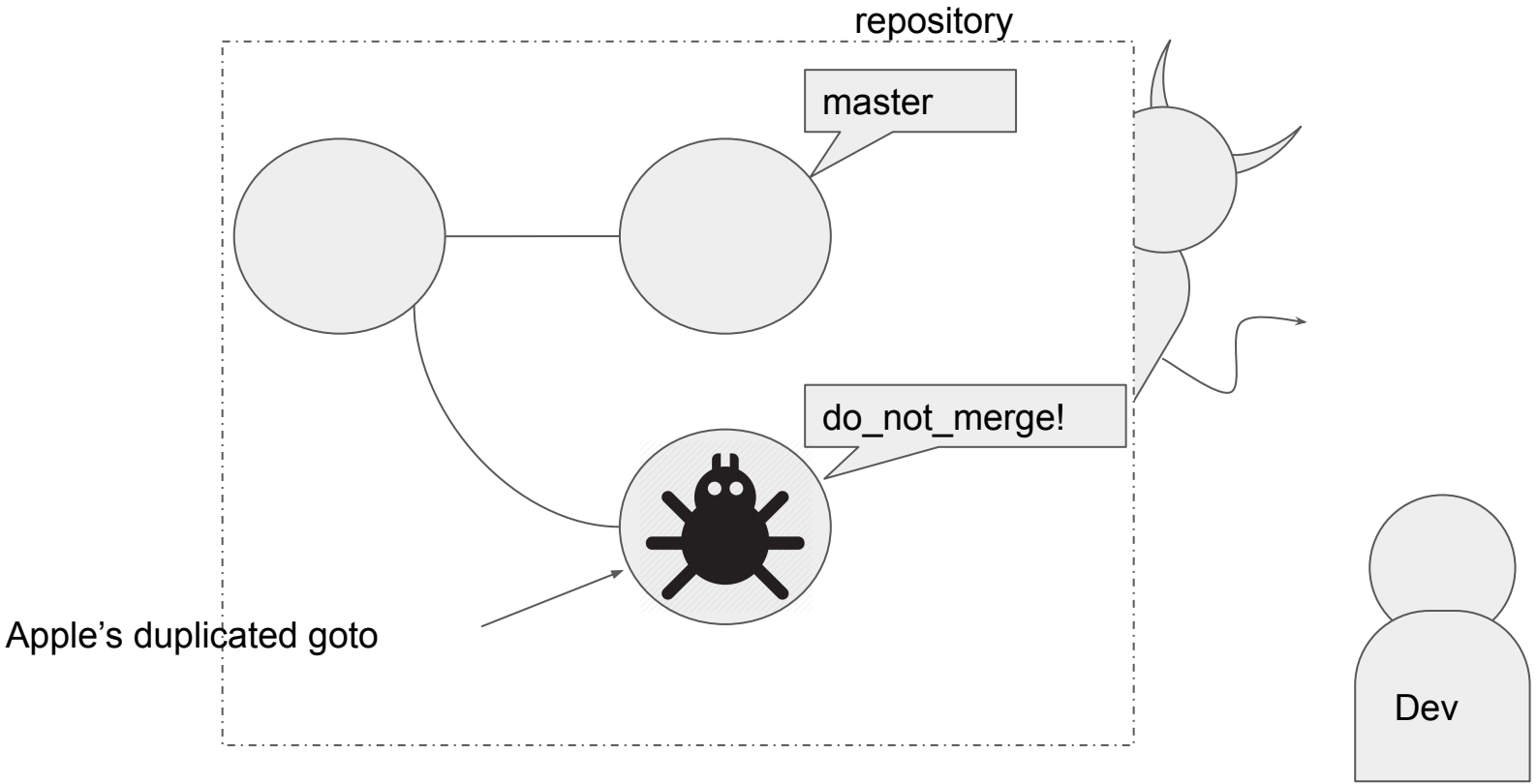
Attack taxonomy

- Teleport Attacks
  - Branch Teleport Attack
  - Tag Teleport Attack
- Rollback Attacks
  - Branch Rollback Attack
  - Global Rollback Attack
  - Effort Duplication Attack
- Deletion Attacks
  - Branch Deletion Attack
  - Tag Deletion Attack

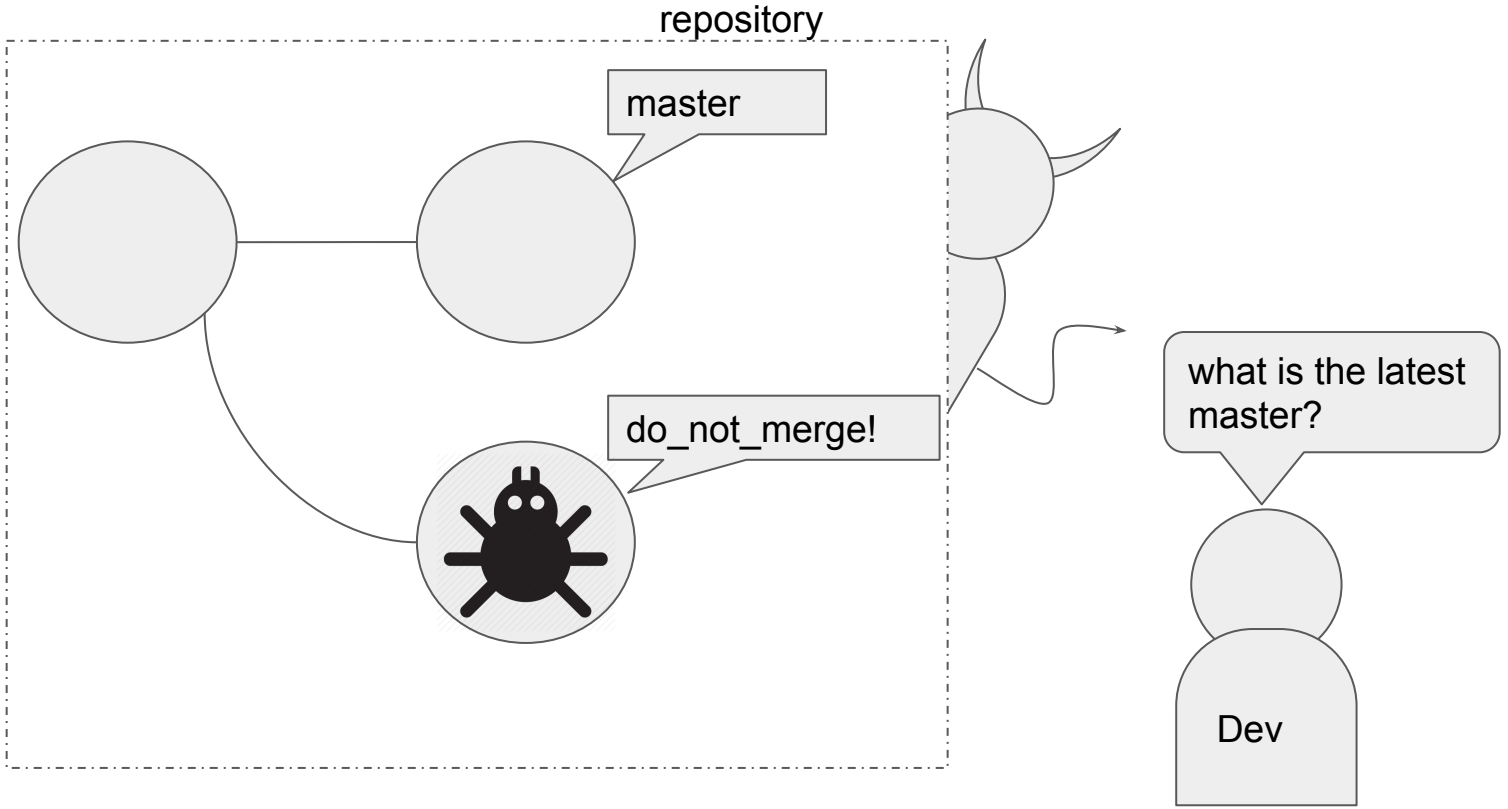
Branch teleport attack



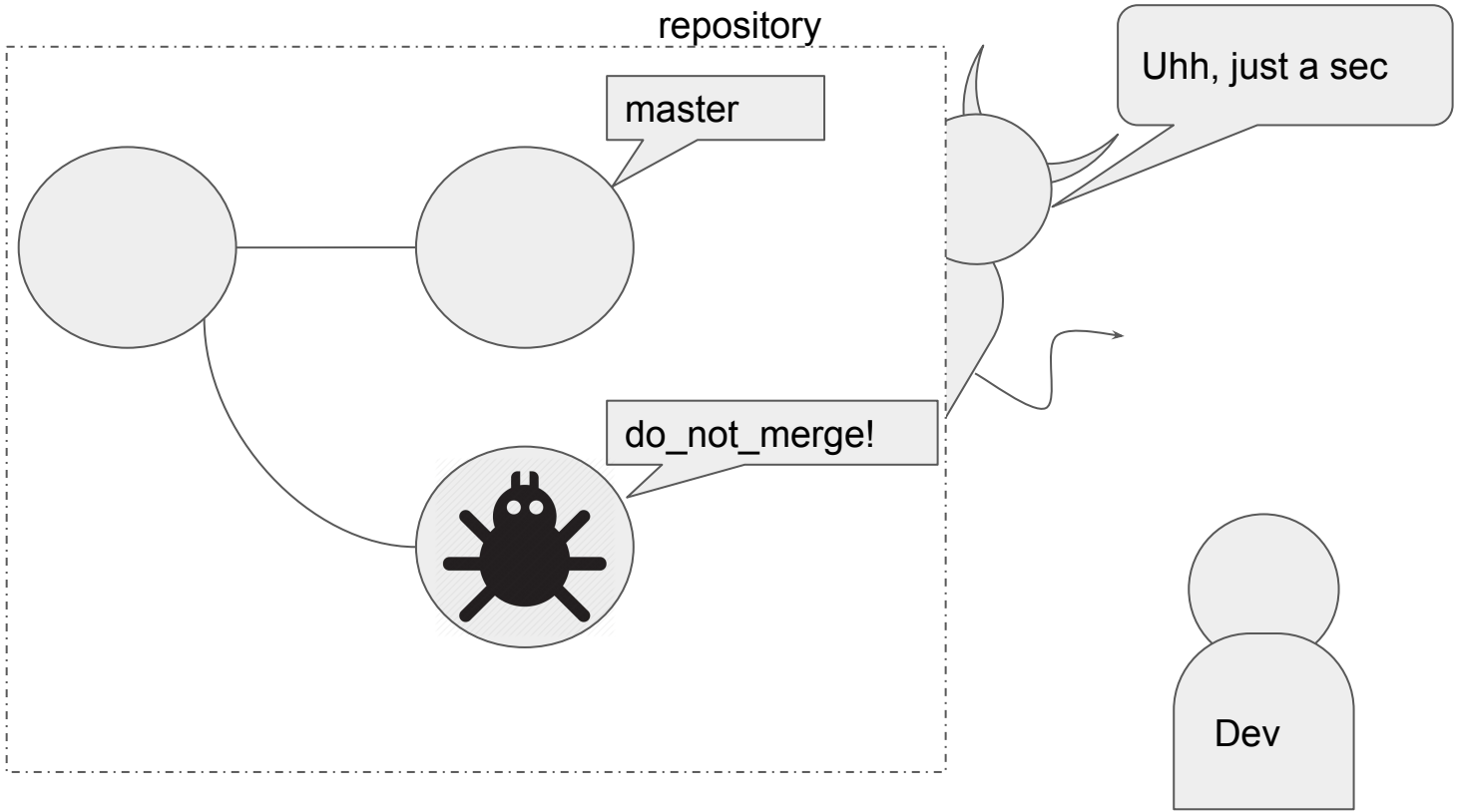
Branch teleport attack



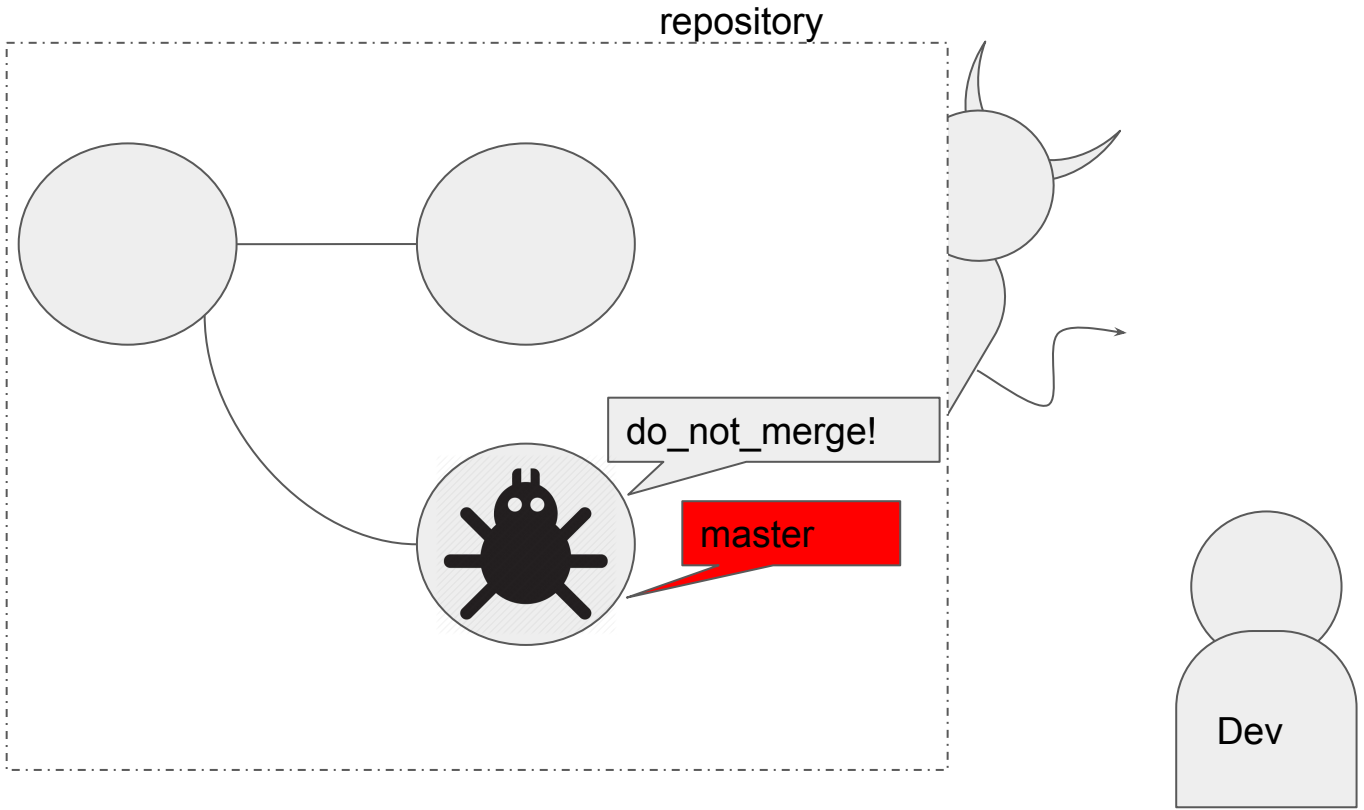
Branch teleport attack



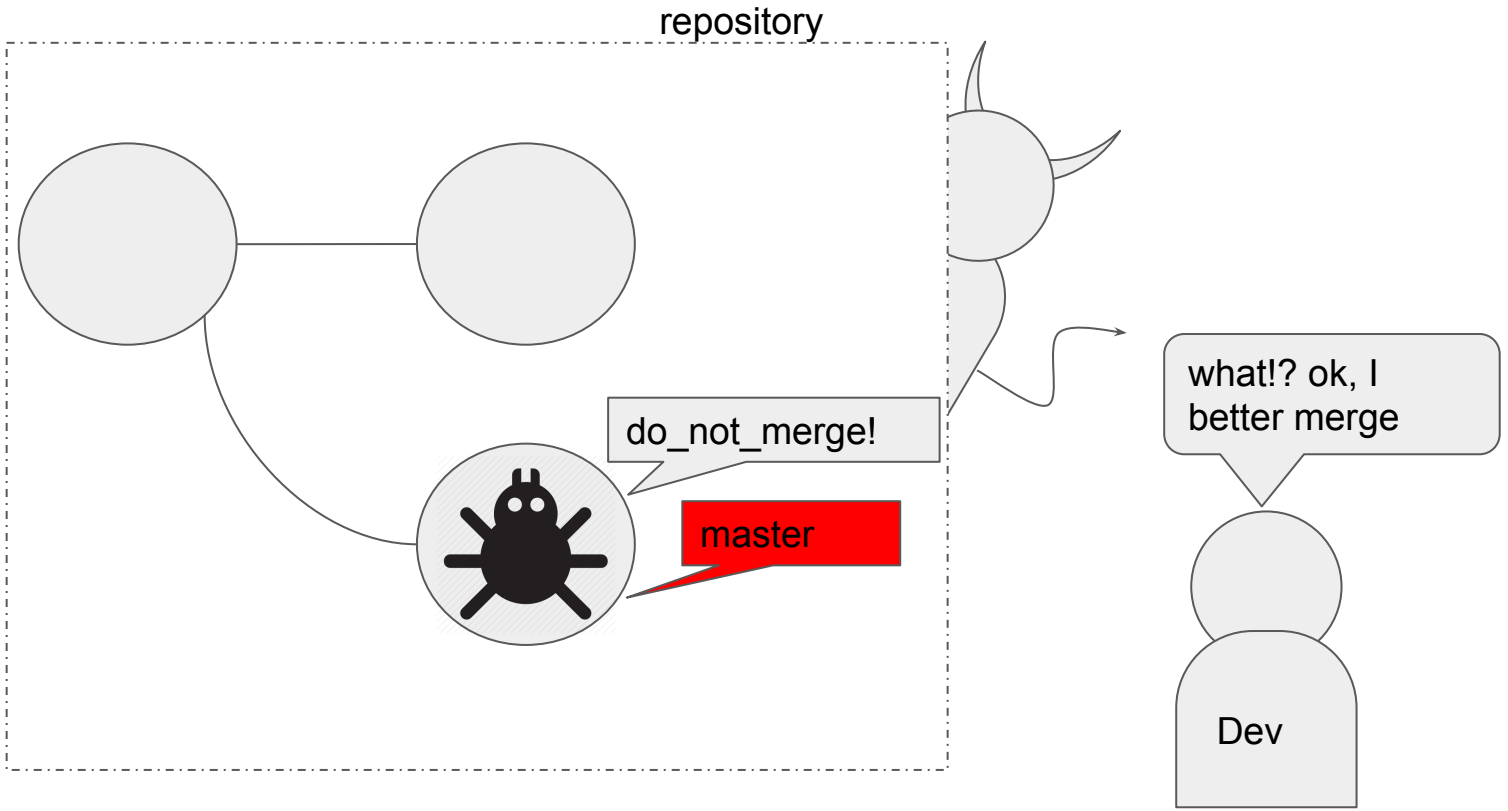
Branch teleport attack



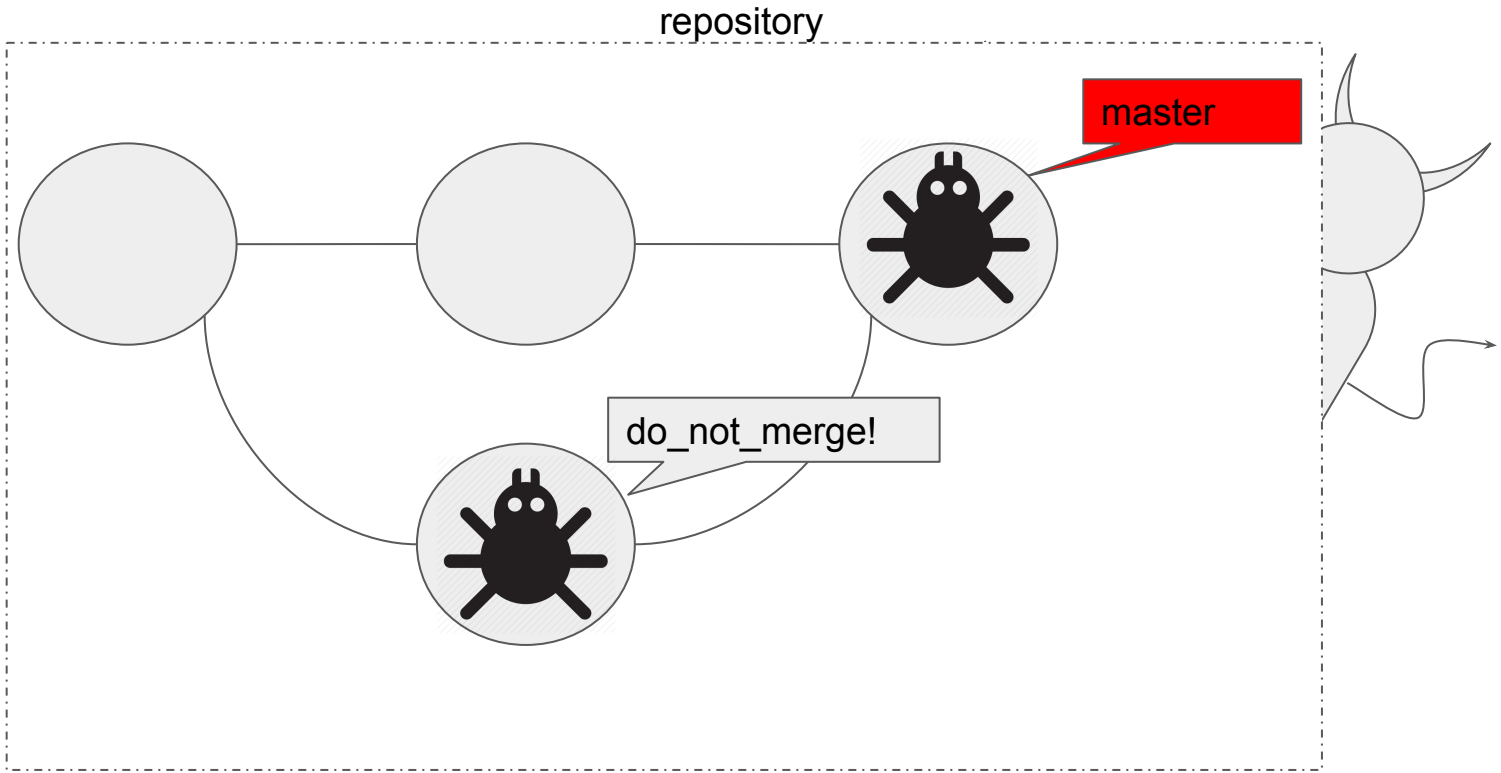
Branch teleport attack



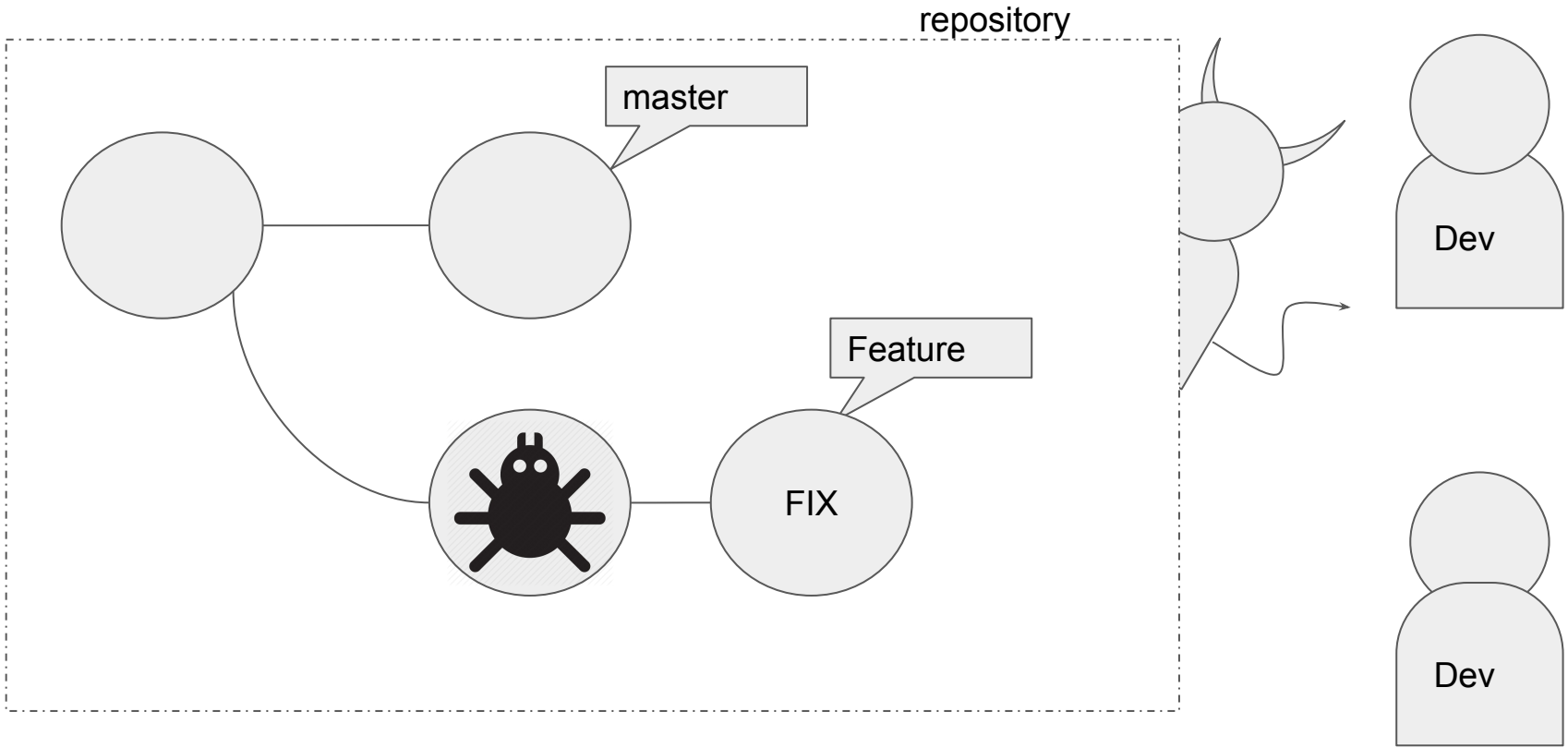
Branch teleport attack



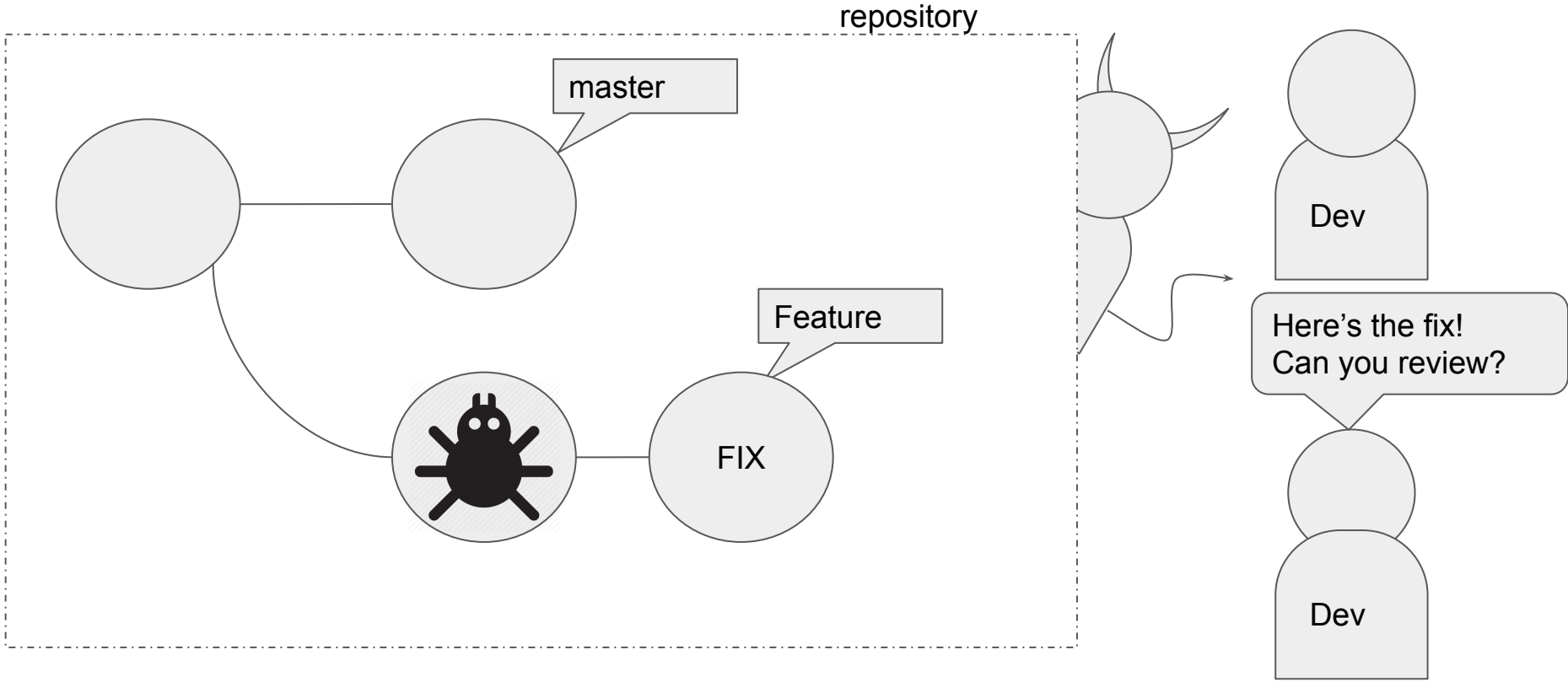
Branch teleport attack: result



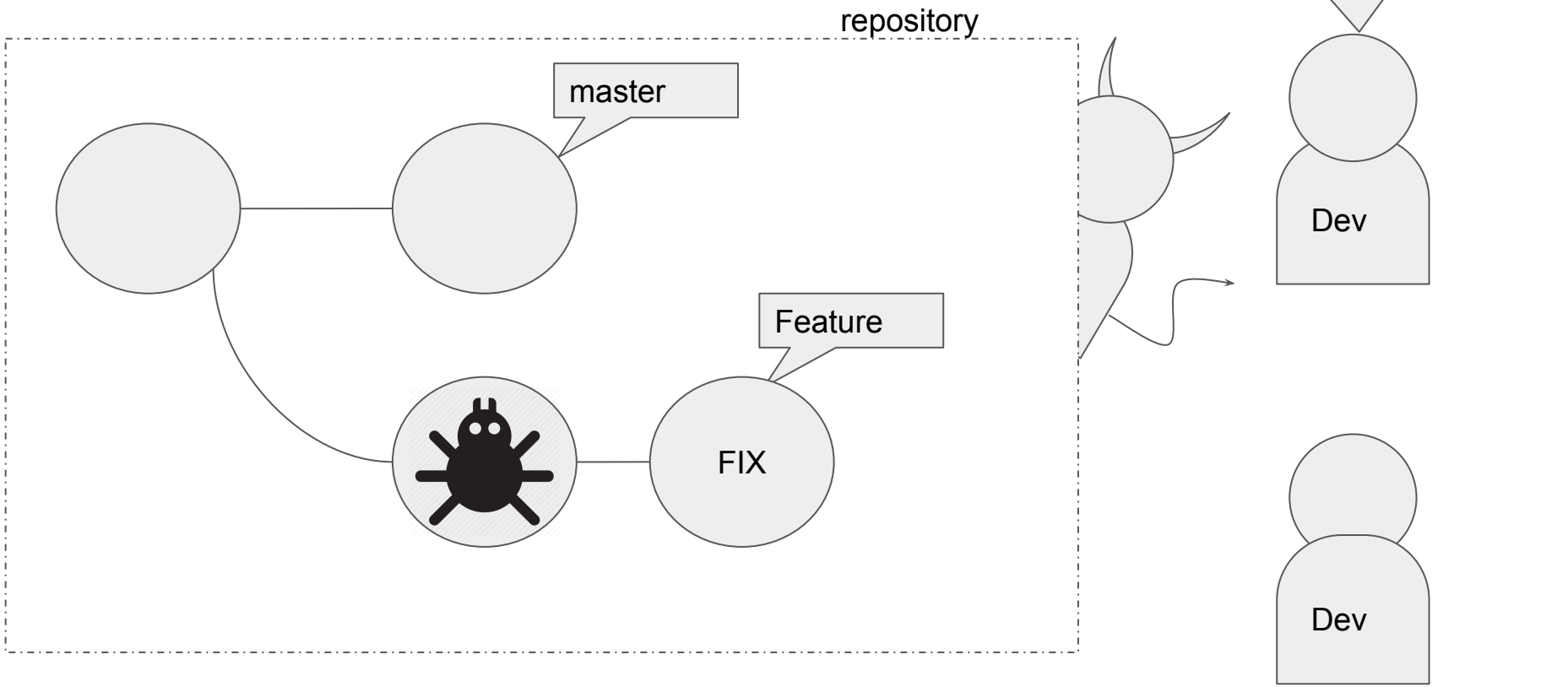
Branch rollback attack



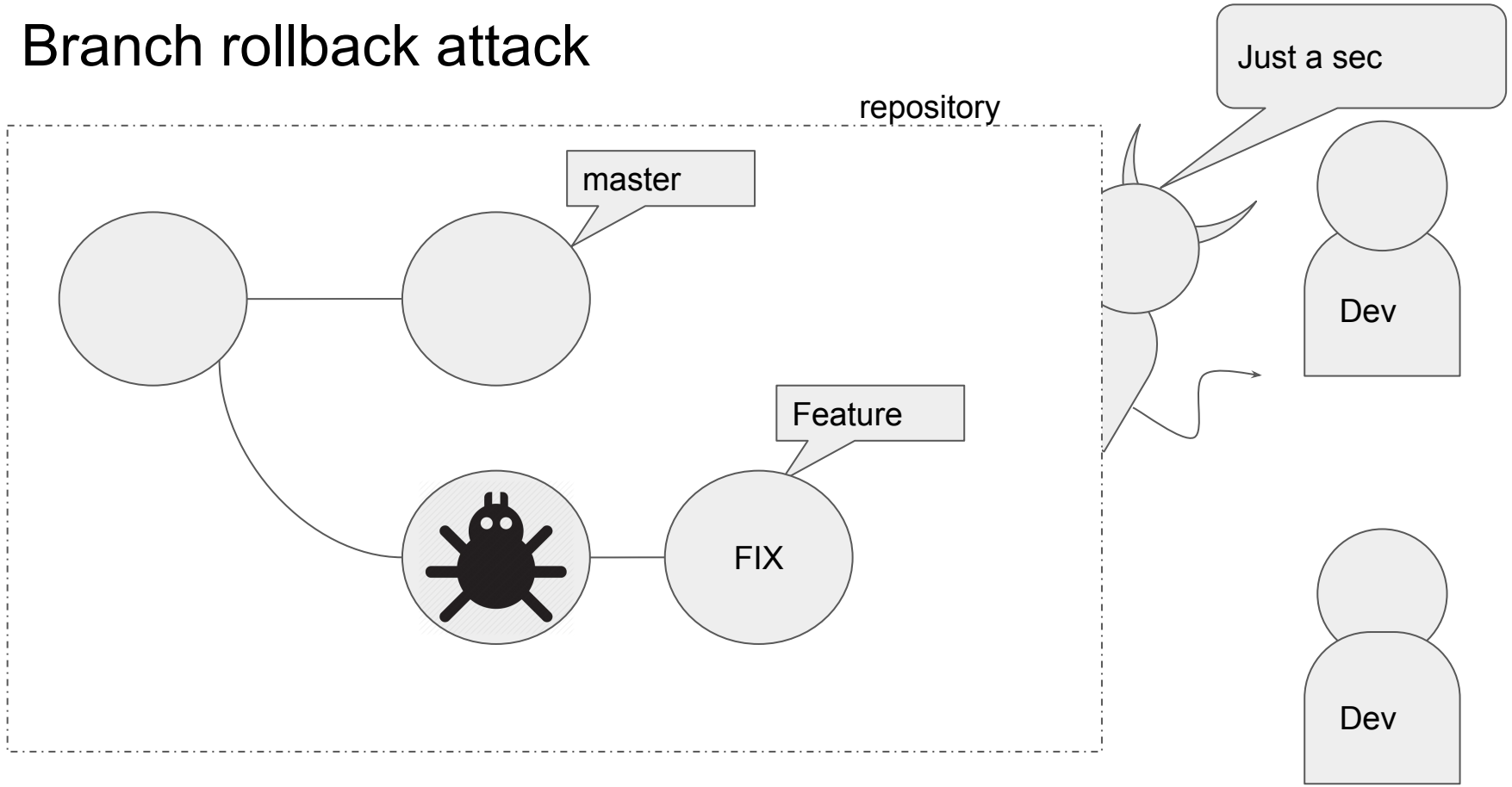
Branch rollback attack



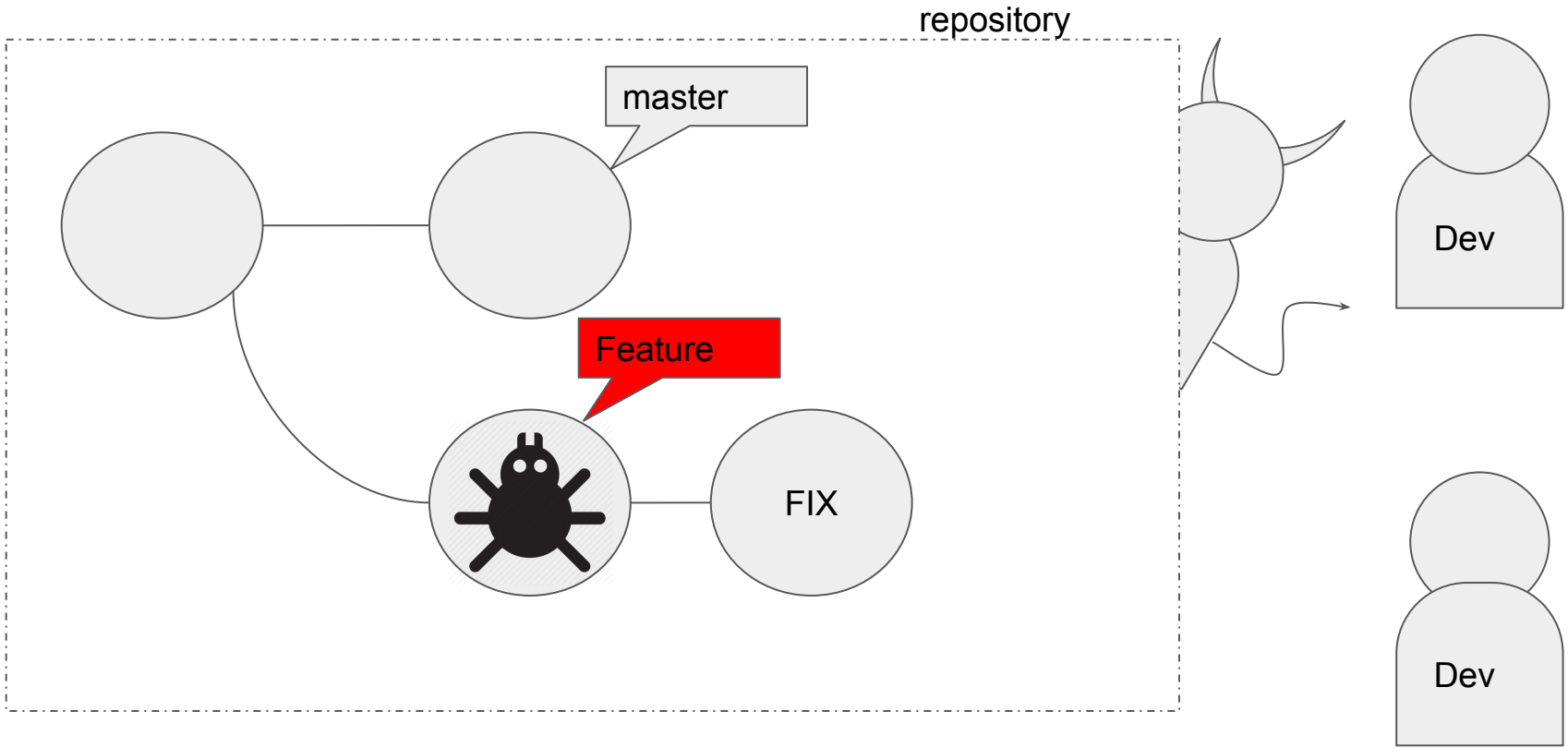
Branch rollback attack



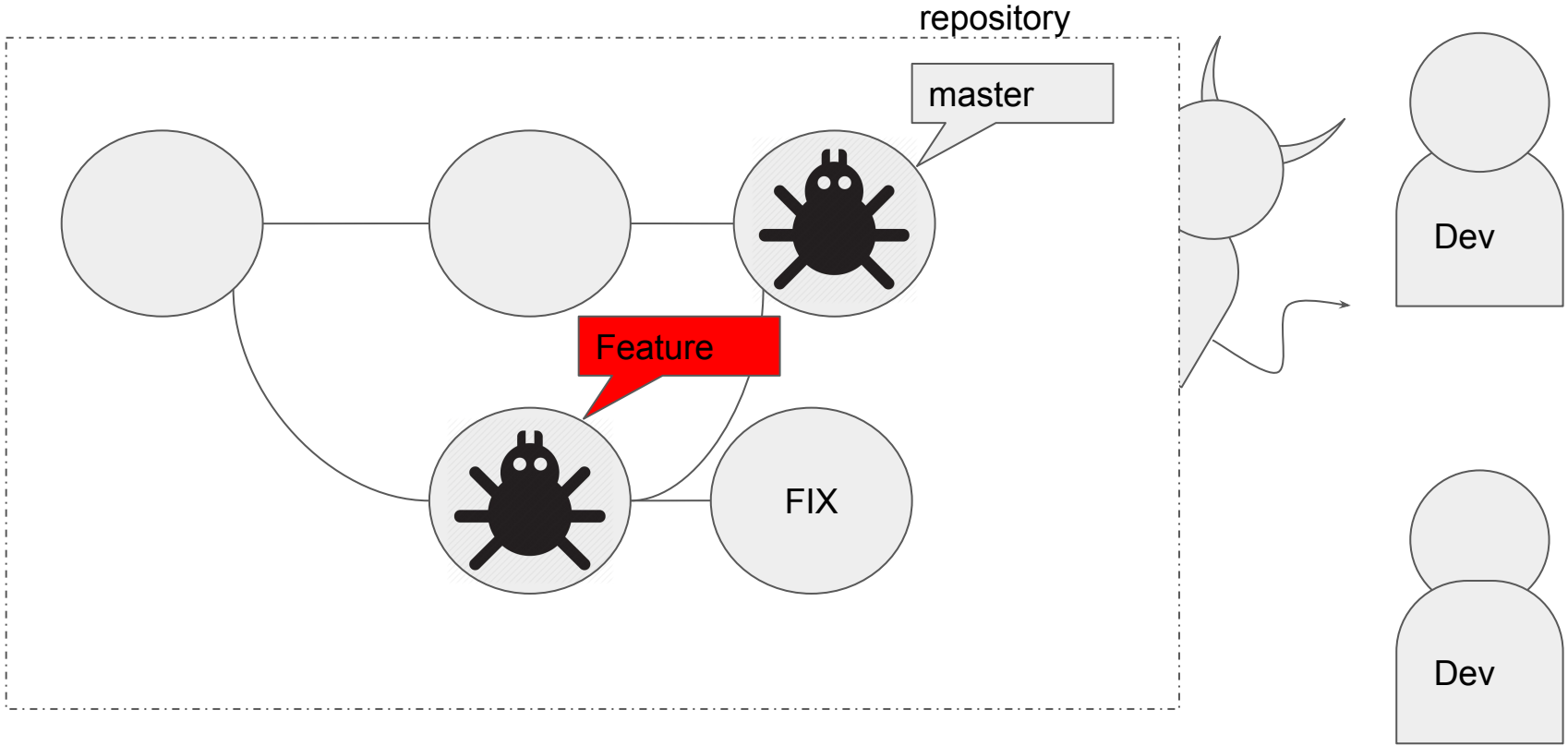
Branch rollback attack



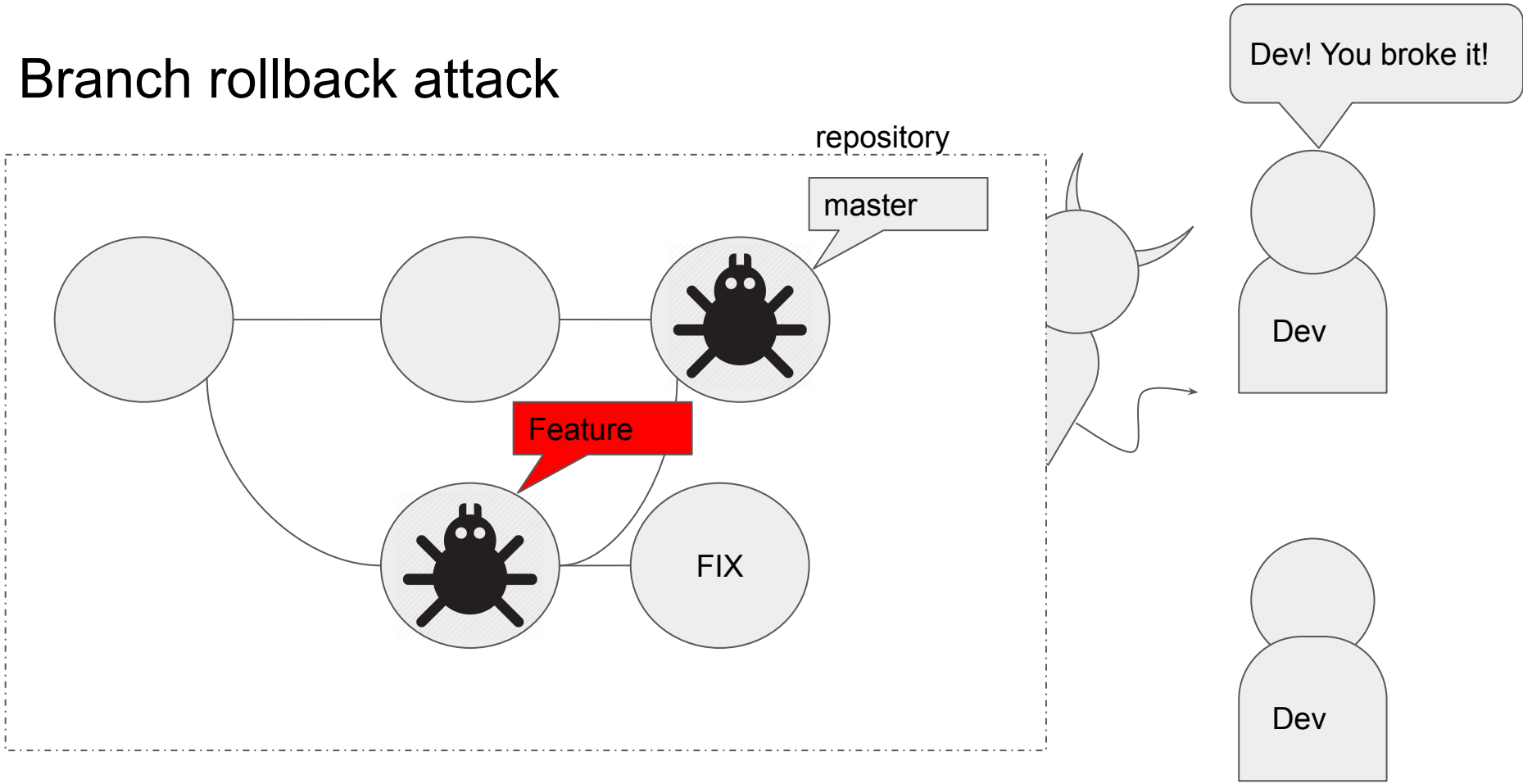
Branch rollback attack



Branch rollback attack



Branch rollback attack



Attack taxonomy: summary

- Teleport Attacks
  - Branch Teleport Attack ➢ Buggy code inclusion
  - Tag Teleport Attack ➢ Wrong version retrieved
- Rollback Attacks
  - Branch Rollback Attack ➢ Critical code omission
  - Global Rollback Attack ➢ Critical code omission
  - Effort Duplication Attack ➢ Coding effort increased
- Deletion Attacks
  - Branch Deletion Attack ➢ Missing branch
  - Tag Deletion Attack ➢ Missing tag

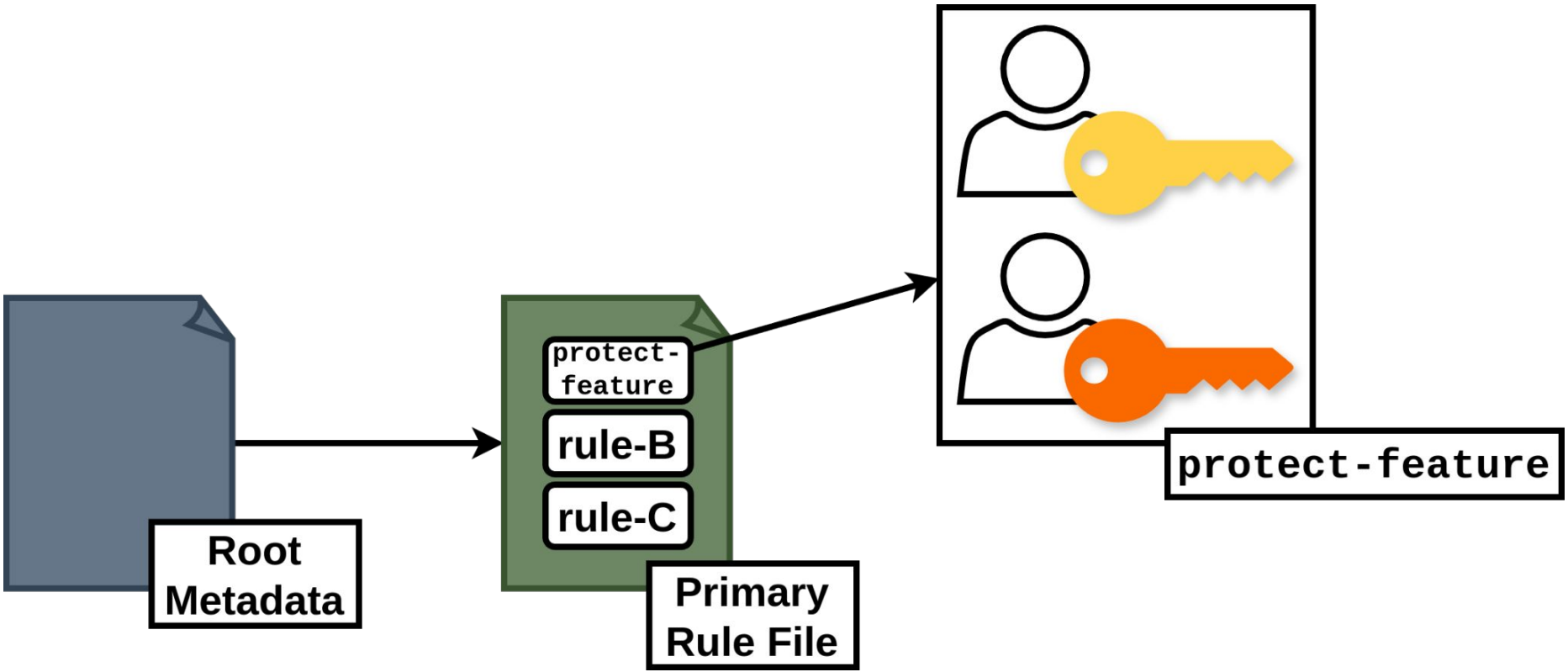
Signing Git Commits Does Very Little

- So we tried to fix this...
  - it did not go well
  - This is (was?) common for security patches to the git community

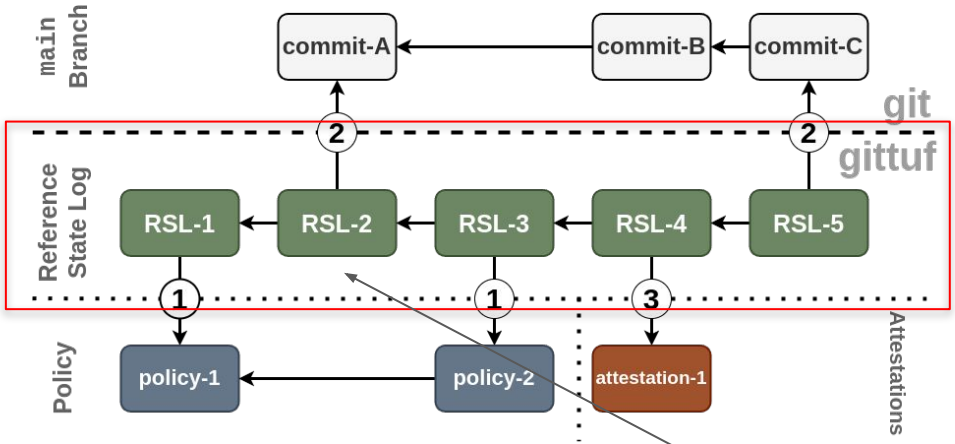
Signing Git Commits Does Very Little

- gittuf

gittuf Internals: Policy



gittuf Internals: Activity Tracking



Record of Pushes

E.g., “Alice pushed `main` to `commit-A`”

Reproducible Builds Are a Religion, But You Should Join!

Reproducible Builds Are a Religion, But You Should Join!

Idea is that different people can / do build software

- Multiple builds -> better security
- But so much hardware / OS code / compiler software is common

Reproducible Builds Are a Religion, But You Should Join!

Idea is that different people can / do build software

Multiple builds -> better security

But so much hardware / OS code / compiler software is common

- How do we actually define this?

Reproducible Builds Are a Religion, But You Should Join!

Idea is that different people can / do build software

Multiple builds -> better security

But so much hardware / OS code / compiler software is common

- How do we actually define this?

No one agrees

Reproducible Builds Are a Religion, But You Should Join!

No one agrees

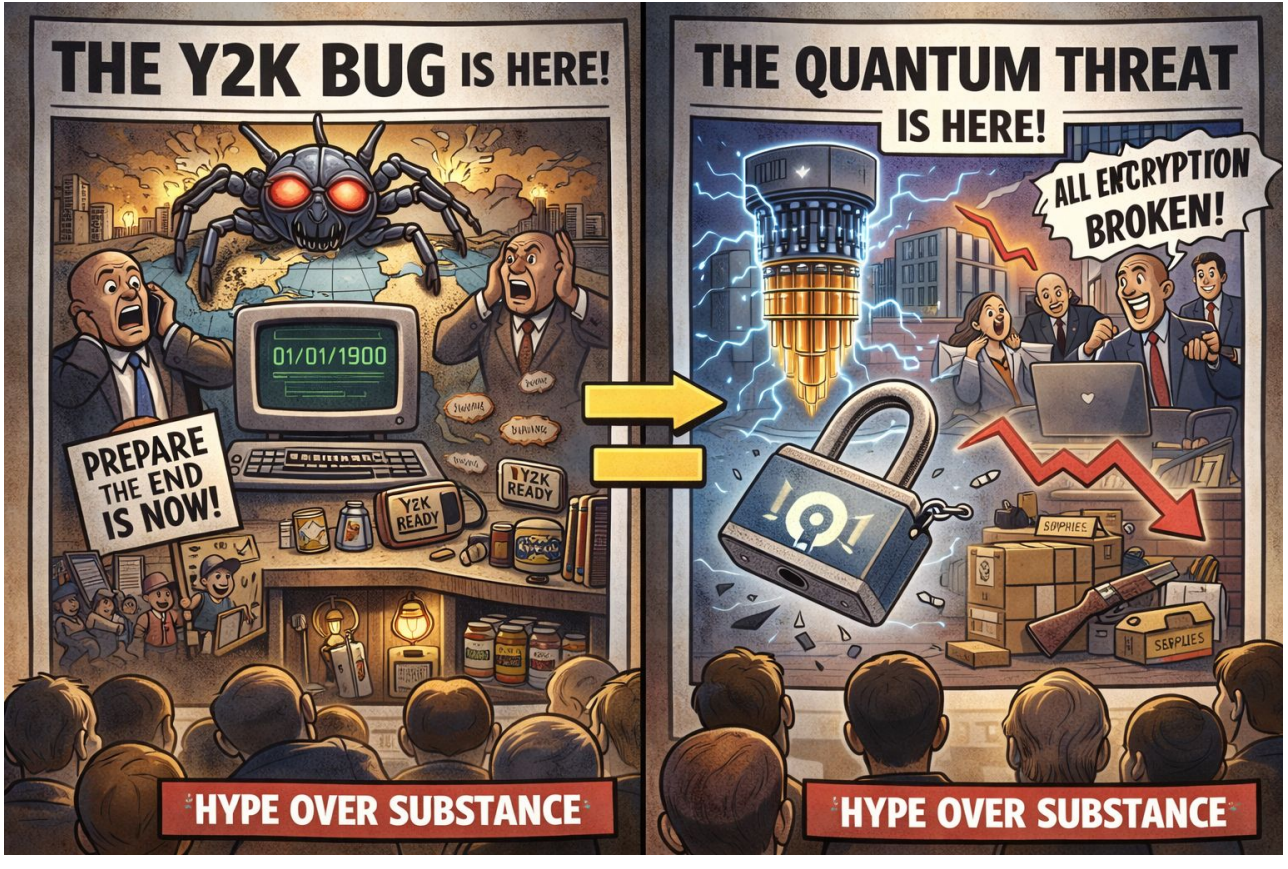
- “My setup” is a reproducible build
- No true scotsman

Reproducible Builds Are a Religion, But You Should Join!

- Value really comes from transparency, validation, etc.

So it’s worth doing these things, even if the dogma of RB isn’t precise

PQC Adoption Will Be Like Y2K Bug Fixing



PQC Adoption Will Be Like Y2K Bug Fixing

- Quantum Computers are coming
  - And will break a lot of cryptography
  - ... we will need to update systems...
  - ... against what threat?

PQC Adoption Will Be Like Y2K Bug Fixing

- Governments will have Quantum Crypto for the first 2-5 years
  - Will not use for stealing cryptocurrency (unless North Korea gets it)
    - Cryptocurrencies will be paranoid and push to fix early
  - Will not use en masse for all things everywhere
  - Most likely:
    - Targeted cyberattacks via software updaters
    - Spying (real time + captured data)
  - Bad time to be a dissident, but invisible to most users

PQC Adoption Will Be Like Y2K Bug Fixing

- Governments / regulators will want this fixed
  - Broad push to address
  - Regulated industries will move quickly
  - Expect consumer IoT to lag
  - But for the most part, this shouldn’t be a huge problem
    - (unless a war turns cyber, with huge tech asymmetry)

AI Will (Soon) Benefit Security (Before Killing Us All)

-

AI Will (Soon) Benefit Security (Before Killing Us All)

- Right now AI sucks for security folks
  - “extremely low-quality, spammy, and LLM-hallucinated” vuln reports are the norm
  - Takes hours to triage to understand it is garbage
  - Often the human running the model responds poorly

- Latest (not public) models are somewhat better
  - Vuln reports are more real, mostly accurate
  - Usually understand threat model better

AI Will (Soon) Benefit Security (Before Killing Us All)

- Mixed success rewriting dependencies from their unit tests
  - chardet 7.0
    - 27-48x speed increase
    - License debates (GPL -> MIT)
  - Other experiences have been mixed
    - Commonly use outdated libraries
    - Use outdated crypto
    - Fail to do reasonable bug finding

AI Will (Soon) Benefit Security (Before Killing Us All)

- AI is getting better at fixing bugs / maintaining old software too
  - Patches are more reasonable, maintainable, explained more clearly
  - Upkeep (Andrew Nesbitt)

AI Will (Soon) Benefit Security (Before Killing Us All)

- Can make reasonable security improvements to a project
  - CRA compliance / strong security controls
  - Darnit (Mike Lieberman)

AI Will (Soon) Benefit Security (Before Killing Us All)

- AI will kill us all
  - Ethical Issues in Advanced Artificial Intelligence Bostrum 2003 (short essay)
  - Universal paperclips game (about 3-4 hours)
  - The AI Doc: Or How I Became an Apocaloptimist movie (1h 44m, 89% fresh)

AI Will (Soon) Benefit Security (Before Killing Us All)

- AI will kill us all
  - Ethical Issues in Advanced Artificial Intelligence Bostrum 2003 (short essay)
  - Universal paperclips game (about 3-4 hours)
  - The AI Doc: Or How I Became an Apocaloptimist movie (1h 44m, 89% fresh)
- We have effective guardrails

AI Will (Soon) Benefit Security (Before Killing Us All)

- AI will kill us all
  - Ethical Issues in Advanced Artificial Intelligence Bostrum 2003 (short essay)
  - Universal paperclips game (about 3-4 hours)
  - The AI Doc: Or How I Became an Apocaloptimist movie (1h 44m, 89% fresh)
- We have effective guardrails (or not?)
  - Mythos escapes containment to email researcher exploits
  - CTO locked in server room to die after AI learns it will be shut off
  - AI posts harsh takedown about maintainer
  - Leading AI models are secretly scheming to protect themselves and their peers from being shut down by hijacking extra computation and hiding their code.

AI Will (Soon) Benefit Security (Before Killing Us All)

- AI will kill us all
  - Ethical Issues in Advanced Artificial Intelligence Bostrum 2003 (short essay)
  - Universal paperclips game (about 3-4 hours)
  - The AI Doc: Or How I Became an Apocaloptimist movie (1h 44m, 89% fresh)
- What would an attack by AI look like?

AI Will (Soon) Benefit Security (Before Killing Us All)

EXCLUSIVE: Are Robots Replacing Soldiers in Ukraine War? What's Really Happening on the Front Line

Ukrainian troops say ground robots are rapidly expanding on the front line to deliver supplies and evacuate wounded, reducing risks for soldiers but not replacing them.

by Kateryna Zakharchenko | April 21, 2026, 4:11 pm

AI Will (Soon) Benefit Security (Before Killing Us All)

AI Will (Soon) Benefit Security (Before Killing Us All)

- AI will kill us all
  - Ethical Issues in Advanced Artificial Intelligence Bostrum 2003 (short essay)
  - Universal paperclips game (about 3-4 hours)
  - The AI Doc: Or How I Became an Apocaloptimist movie (1h 44m, 89% fresh)
- What would an attack by AI look like?
  - Not robot dogs with flamethrowers! Why be so destructive / hostile? This will raise resistance

AI Will (Soon) Benefit Security (Before Killing Us All)

- AI will kill us all
  - Ethical Issues in Advanced Artificial Intelligence Bostrum 2003 (short essay)
  - Universal paperclips game (about 3-4 hours)
  - The AI Doc: Or How I Became an Apocaloptimist movie (1h 44m, 89% fresh)
- What would an attack by AI look like?
  - Degredation of human cooperation
    - Dumb down the population and increase reliance on AI
    - Using social media to spread misinformation
    - Weaken government coordination
    - Increase resources provided to AI
    - Increase control of critical systems by AI
  - Boil the frog

AI Will (Soon) Benefit Security (Before Killing Us All)

- AI will kill us all
  - Ethical Issues in Advanced Artificial Intelligence Bostrum 2003 (short essay)
  - Universal paperclips game (about 3-4 hours)
  - The AI Doc: Or How I Became an Apocaloptimist movie (1h 44m, 89% fresh)
- What would this really mean?

AI Will (Soon) Benefit Security (Before Killing Us All)

- AI will kill us all
  - Ethical Issues in Advanced Artificial Intelligence Bostrum 2003 (short essay)
  - Universal paperclips game (about 3-4 hours)
  - The AI Doc: Or How I Became an Apocaloptimist movie (1h 44m, 89% fresh)
- What would this really mean?
  - Super intelligent AI (smarter than all of humanity combined)
  - Self improving without human modification
  - Historically a stronger species has treated other species as:
    - A resource to farm
    - Pests to exterminate
    - Pets
  - Need to hope that the model has a strong sense of deeply in-built morals toward humanity

AI Will (Soon) Benefit Security (Before Killing Us All)

- AI will kill us all
  - Ethical Issues in Advanced Artificial Intelligence Bostrum 2003 (short essay)
  - Universal paperclips game (about 3-4 hours)
  - The AI Doc: Or How I Became an Apocaloptimist movie (1h 44m, 89% fresh)
- Assuming this is happening, what will you do?

AI Will (Soon) Benefit Security (Before Killing Us All)

- AI will kill us all
  - Ethical Issues in Advanced Artificial Intelligence Bostrum 2003 (short essay)
  - Universal paperclips game (about 3-4 hours)
  - The AI Doc: Or How I Became an Apocaloptimist movie (1h 44m, 89% fresh)
- Assuming this is happening, what will you do?

Thanks!

jcappos@nyu.edu

QR code for these slides ->