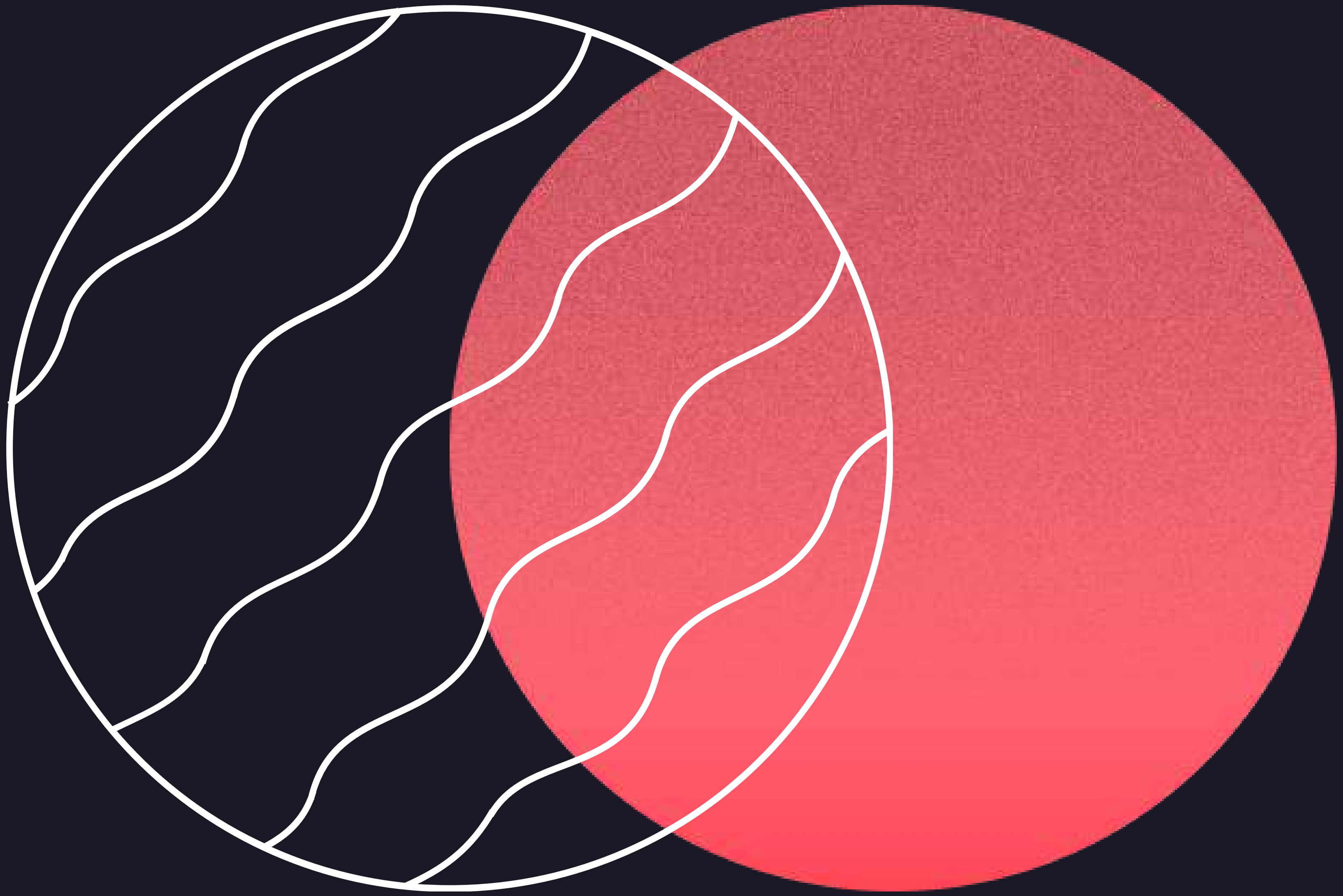


Securing Our Software Supply Chain Using SLSA

By Kaif Ahsan

Hi, I'm Kaif!

Technology &
Cybersecurity enthusiast

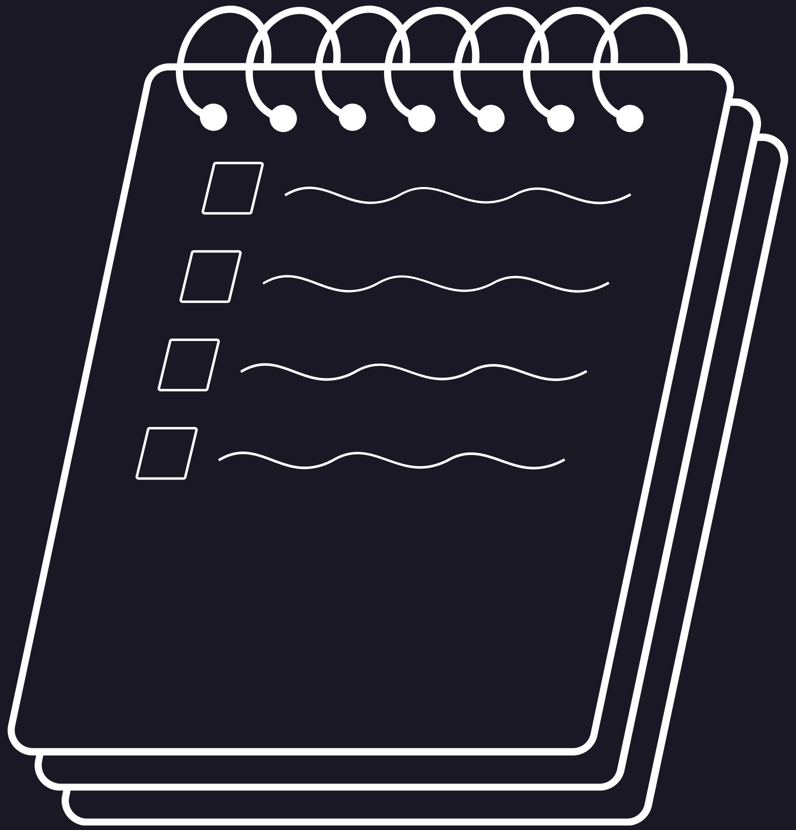
THE UNIVERSITY OF MELBOURNE

MELBOURNE SPACE PROGRAM

Microsoft

ATLASSIAN

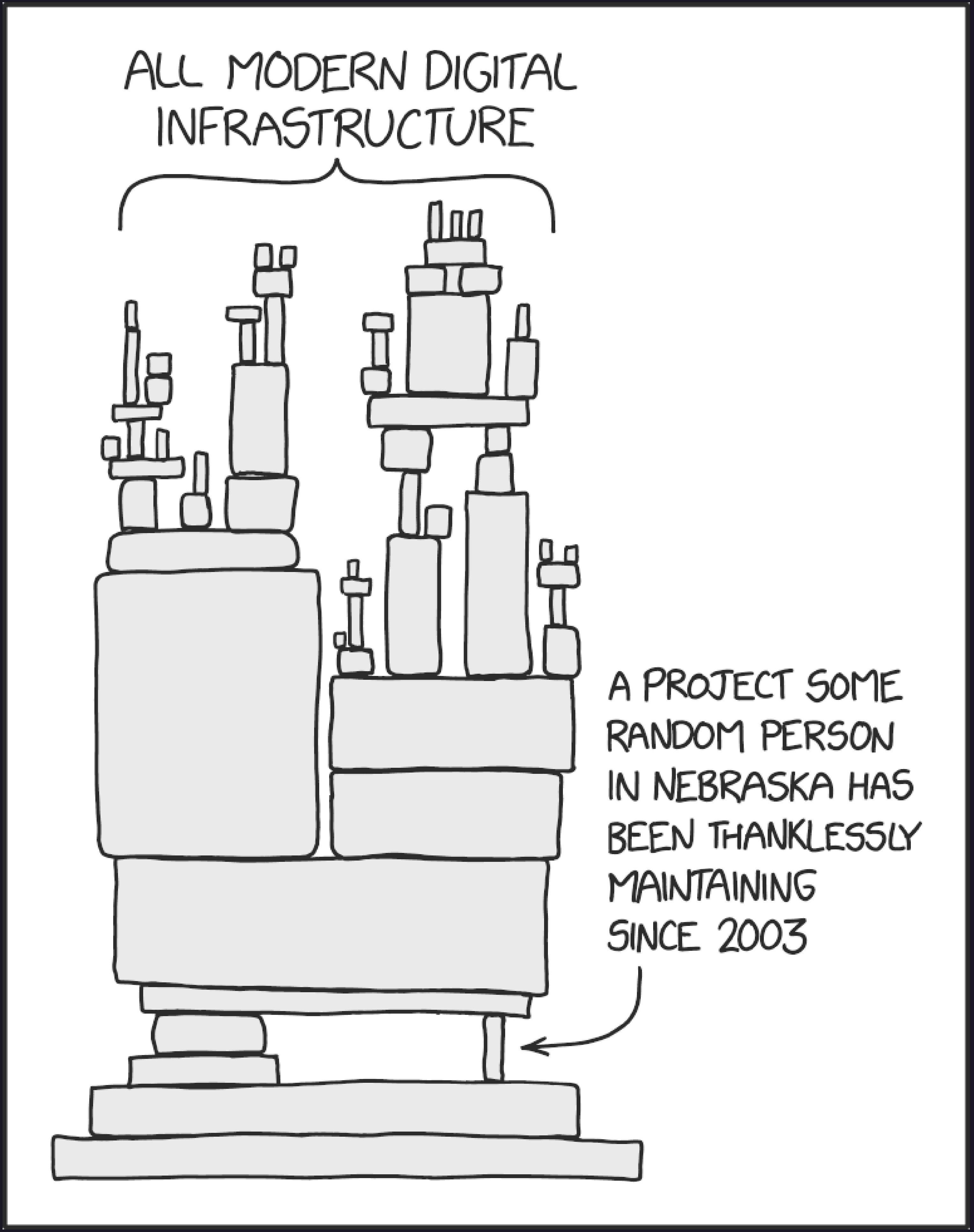
Agenda



- Understand the landscape
- Introduction To SLSA
- Understand the value
- What SLSA does not cover

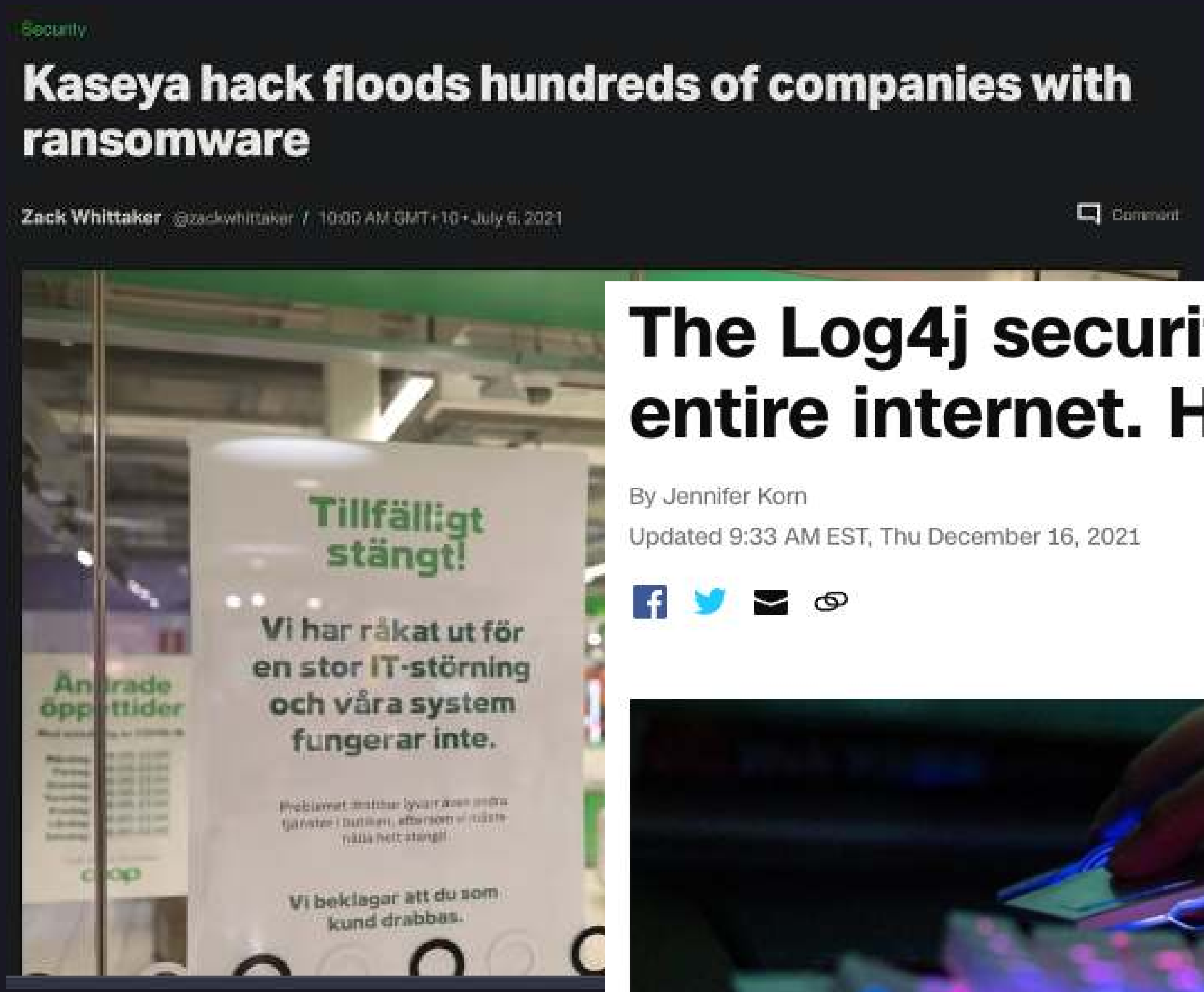
Understanding the landscape

Why reinvent the wheel?


ALL MODERN DIGITAL INFRASTRUCTURE
A PROJECT SOME RANDOM PERSON IN NEBRASKA HAS BEEN THANKLESSLY MAINTAINING SINCE 2003

Source: xkcd

Attacks The Shook The Tech World


Tillfälligt stängt!
Vi har råkat ut för en stor IT-störning och våra system fungerar inte.
Vi beklagar att du som kund drabbas.
Ändrade öppettider
coop

Security

Kaseya hack floods hundreds of companies with ransomware

Zack Whittaker @zackwhittaker / 10:00 AM GMT+10 • July 6, 2021

Comment

HACKING

SolarWinds Hacked, Used in Potentially Massive Supply Chain Attack

ember 14, 2020

npany's IT monitoring and

rch.

e most evasive espionage campaigns

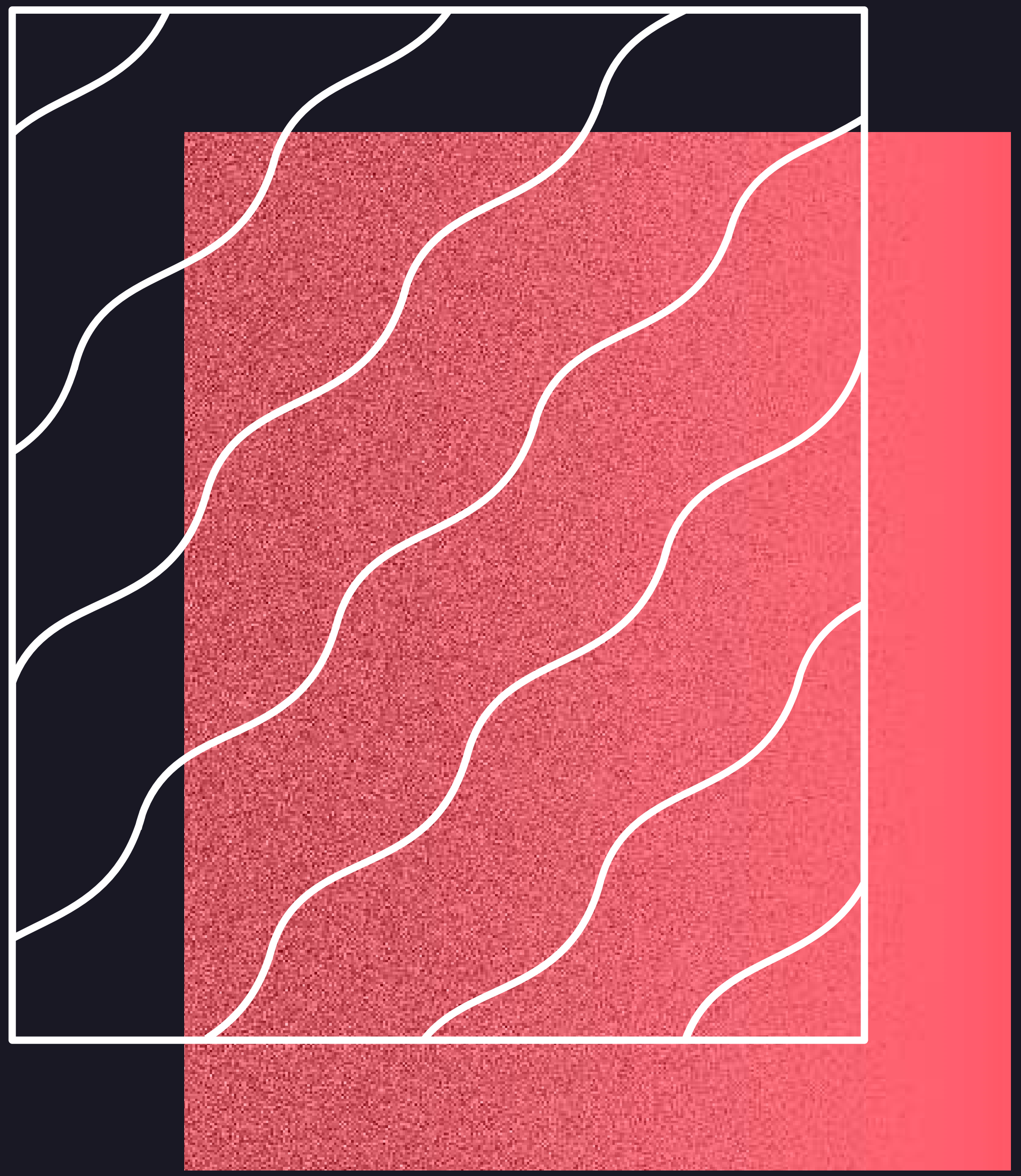
d over the weekend that as part of what

kers managed to breach the U.S. Treasury,

The Log4j security flaw could impact the entire internet. Here's what you should know

By Jennifer Korn

Updated 9:33 AM EST, Thu December 16, 2021



The Broken Supply Chain

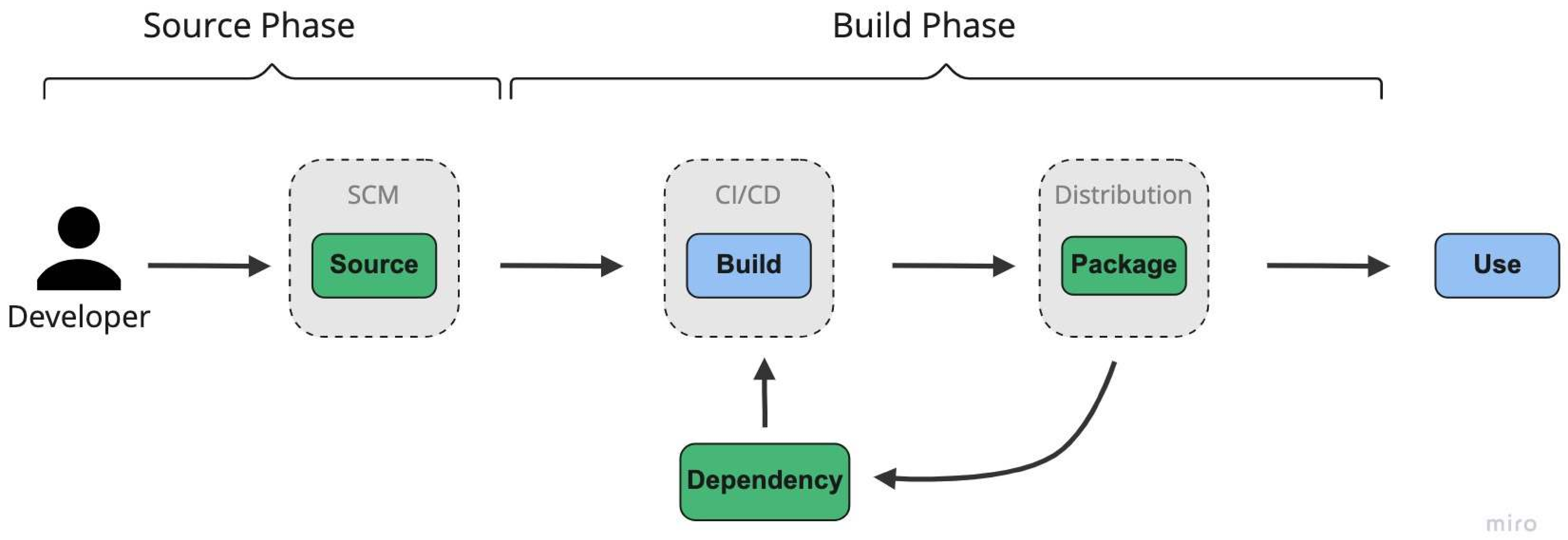
Visibility - How do we comprehensively know what we depend on?

Integrity - How do we know no one tampered with the software?

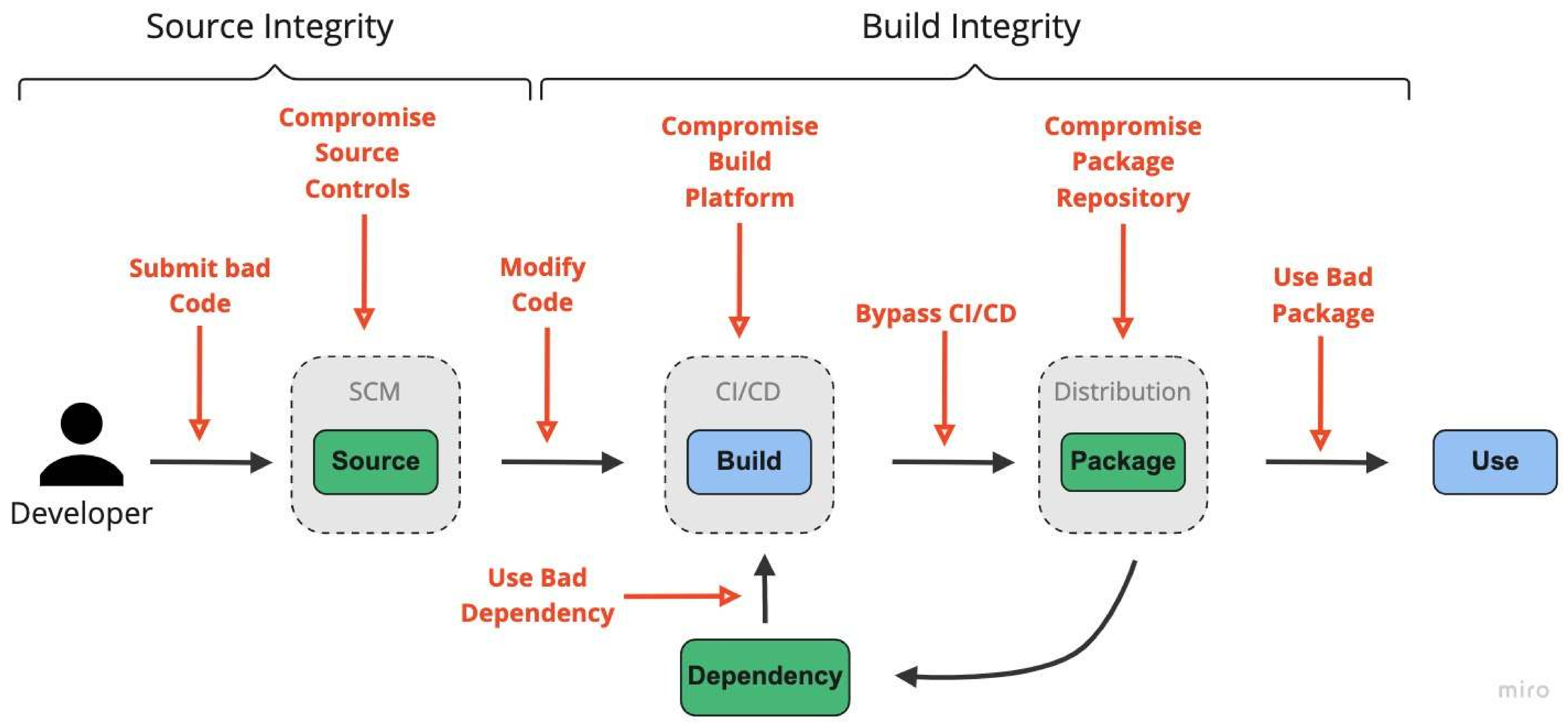
Remediation - If a vulnerability is detected in the supply chain, how quickly can we take action on it?

Ensuring integrity is a big challenge

Typical Software development lifecycle (SDLC)



Potential Threats



How to ensure integrity?

SLSA Framework

Supply-chain Levels for Software Artifacts - SLSA

A checklist of standards and controls to prevent tampering.

It provides you with a roadmap to gradually secure the supply chain.

Cross-organisation and vendor-neutral effort led by OpenSSF.

Applications of SLSA

By First Party

- Reducing risk within an organization from insiders and compromised accounts.

- Organisations of all sizes can follow the checklists according to their need to ensure their software is secure from tampering.

For Open Source

- Reducing the risks from consuming open-source software.

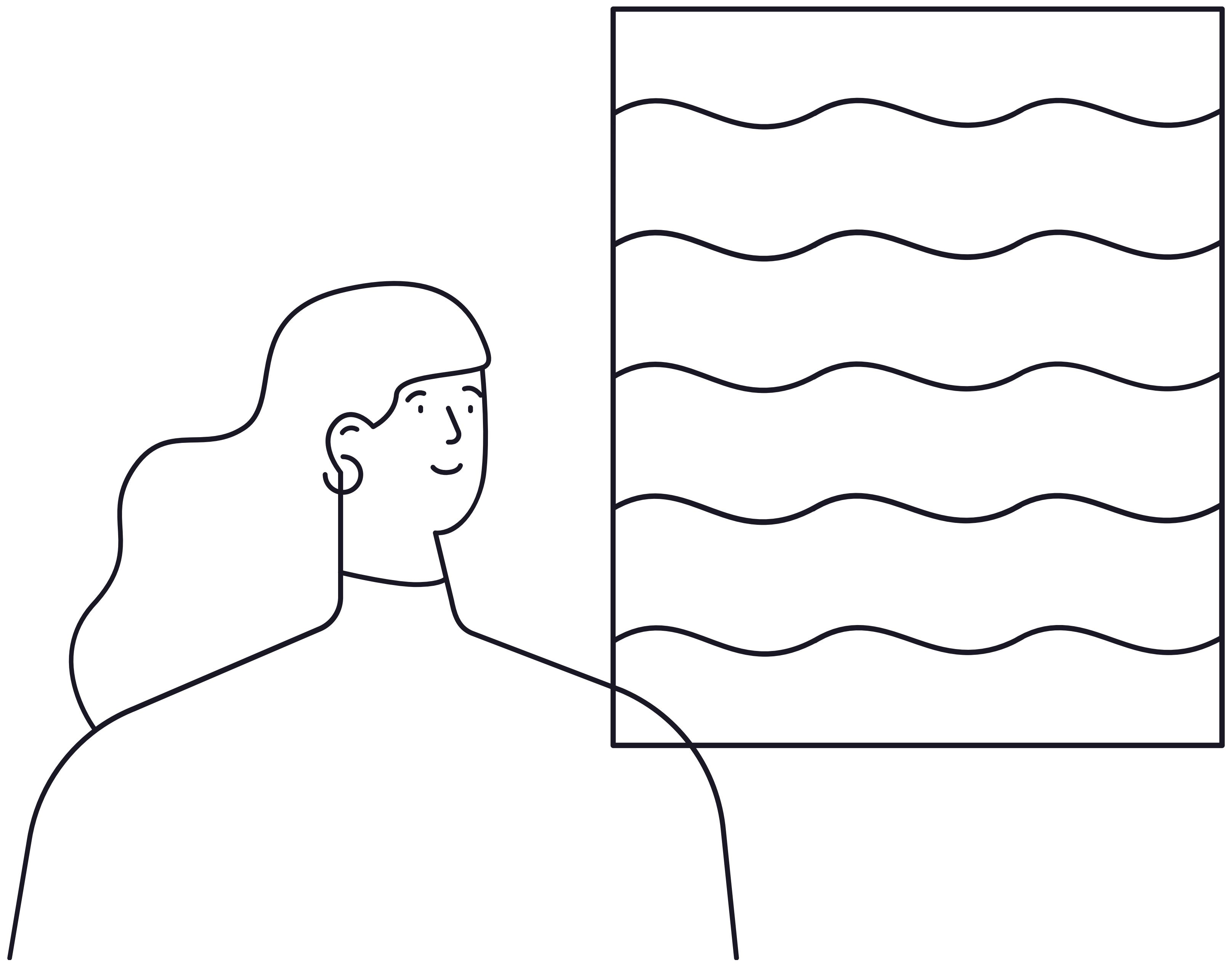
For Vendors

- Reducing the risk from consuming vendor-provided software and services.

- SLSA can be used to validate claims made by vendors and ensure the integrity of the supplied artifacts.

How does SLSA ensure integrity?

Understanding Some Key Terminology



Key Terminologies

Provenance

Verifiable information that ensures the integrity of software. Where, when and how something was produced.

Attestation

Authenticated, machine-readable metadata about one or more software artifacts. For example, build commands and dependencies of an artifact.

Key Terminologies

Artifact

An immutable blob of data. For example, A file, a git commit, a directory of files, a container image etc.

SDLC

Acronym for the software development lifecycle.

How does SLSA ensure integrity?

Two Pillars of SLSA

Attestation

SLSA allows us to effectively verify the integrity of the artifacts at different stages of SDLC

Hardening

SLSA guidelines make various stages of SDLC more resilient to tampering attacks.

SLSA Levels

Level 1

Easy to adopt, giving you supply chain visibility and being able to generate provenance

Level 2

Starts to protect against software tampering and adds minimal build integrity guarantees

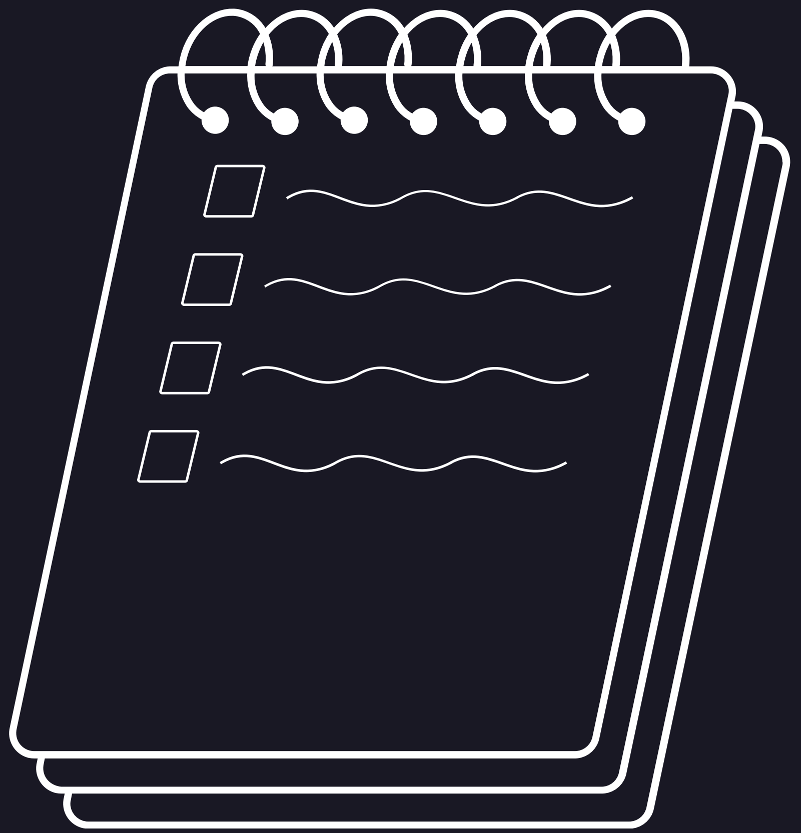
Level 3

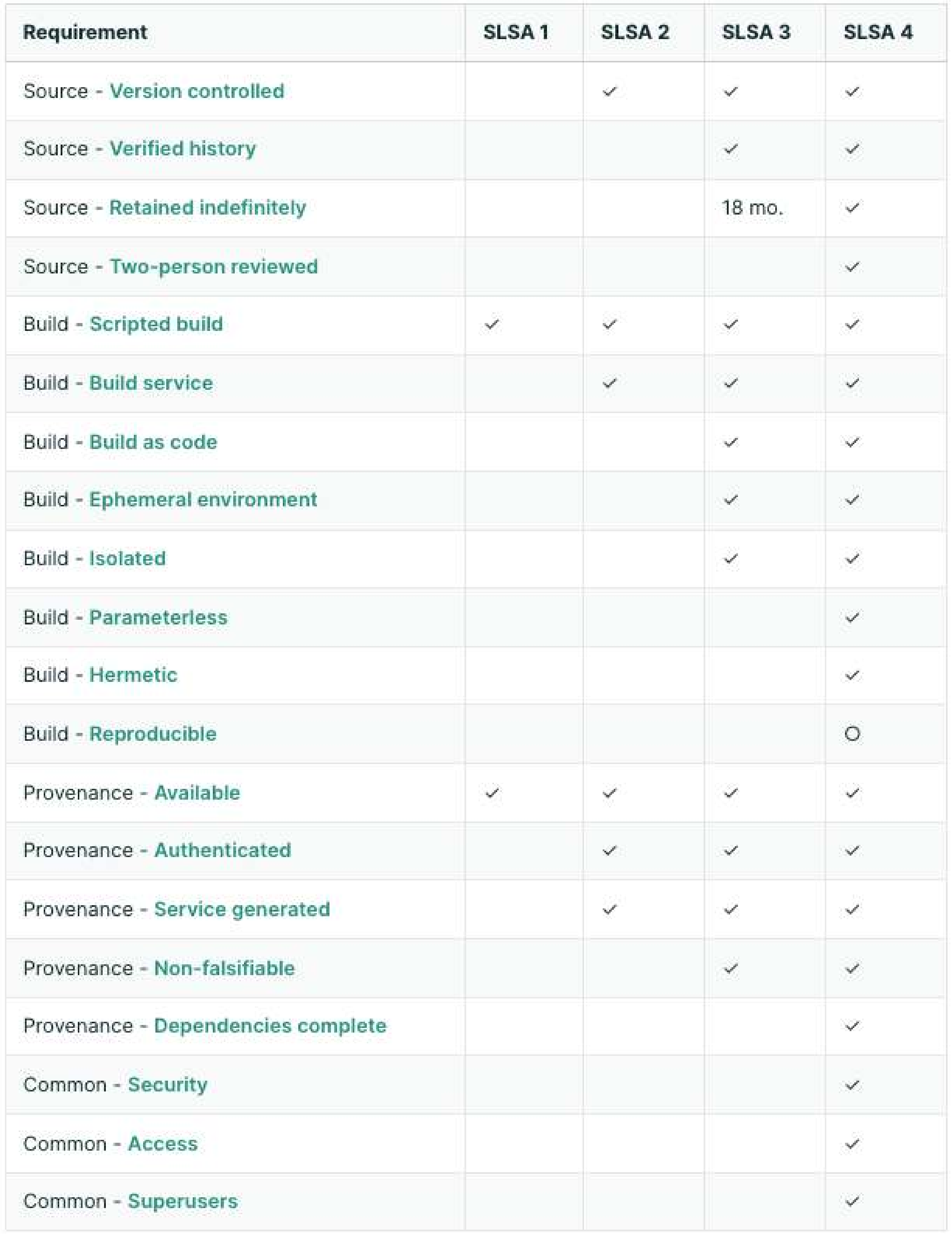
Hardens the infrastructure against attacks, more trust integrated into complex systems and architecture.

Level 4

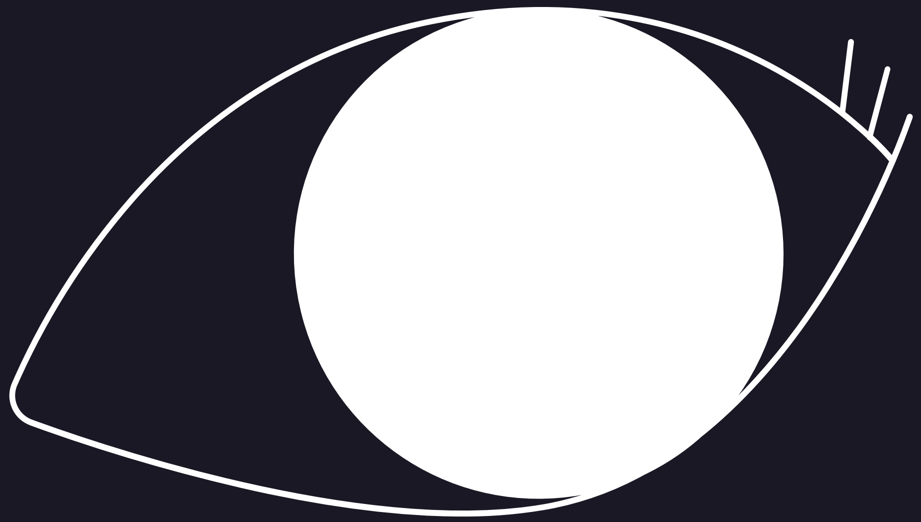
The highest assurances of build integrity and measures for dependency management in place

Overview





Requirement	SLSA 1	SLSA 2	SLSA 3	SLSA 4
Source - Version controlled		✓	✓	✓
Source - Verified history			✓	✓
Source - Retained indefinitely			18 mo.	✓
Source - Two-person reviewed				✓
Build - Scripted build	✓	✓	✓	✓
Build - Build service		✓	✓	✓
Build - Build as code			✓	✓
Build - Ephemeral environment			✓	✓
Build - Isolated			✓	✓
Build - Parameterless				✓
Build - Hermetic				✓
Build - Reproducible				○
Provenance - Available	✓	✓	✓	✓
Provenance - Authenticated		✓	✓	✓
Provenance - Service generated		✓	✓	✓
Provenance - Non-falsifiable			✓	✓
Provenance - Dependencies complete				✓
Common - Security				✓
Common - Access				✓
Common - Superusers				✓



Example Protections by SLSA

Protections Related to Source

Requirement	Description	Level
Version Controlled	Every change to the source is tracked	2
Verified History	Every change in the revision history has at least one strongly authenticated actor and timestamp	3
Retained Indefinitely	The revision and its change history are preserved indefinitely and cannot be deleted.	3
Two-person reviewed	Every change in the revision's history was agreed to by two trusted and strongly authenticated persons prior to submission.	4

Hardening The Source

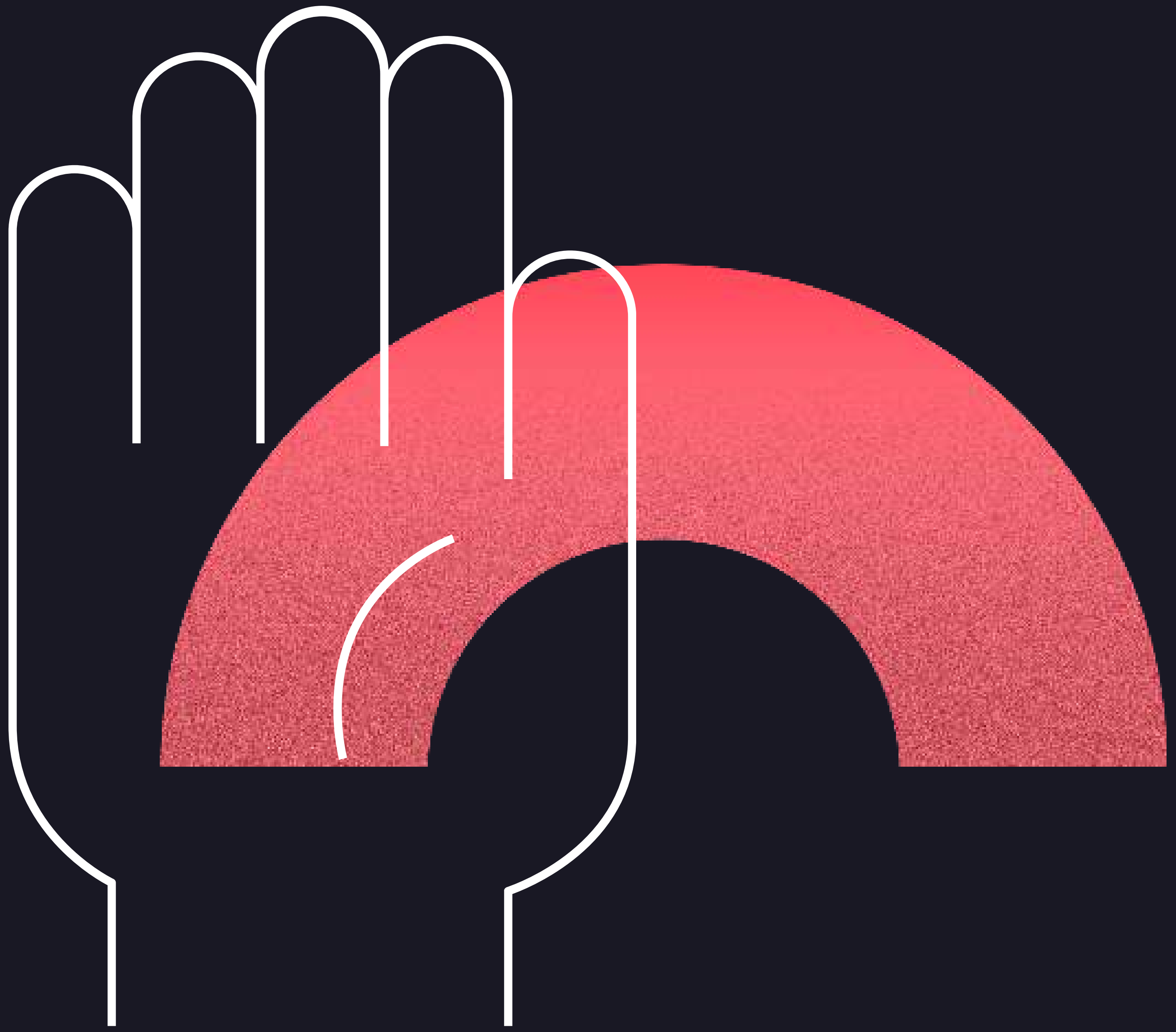
- Have change history so we can verify and validate
- It ensures that only trusted persons can add/update code.
- Unvetted code cannot reach users.

Protections Related to Build

Requirement	Description	Level
Scripted Builds	All Build steps were fully defined in some 'build scripts'.	1
Build Service	All build steps ran using some build service, not on a developer's workstation.	2
Ephemeral environment	The build service ensured that the build steps ran in an short-lifespan environment.	3
Isolated	The build service ensured that the build steps ran in an isolated environment free of influence of other builds.	3
Hermetic	All transitive build steps, sources and dependencies were fully declared up front and build step ran with no network access.	4

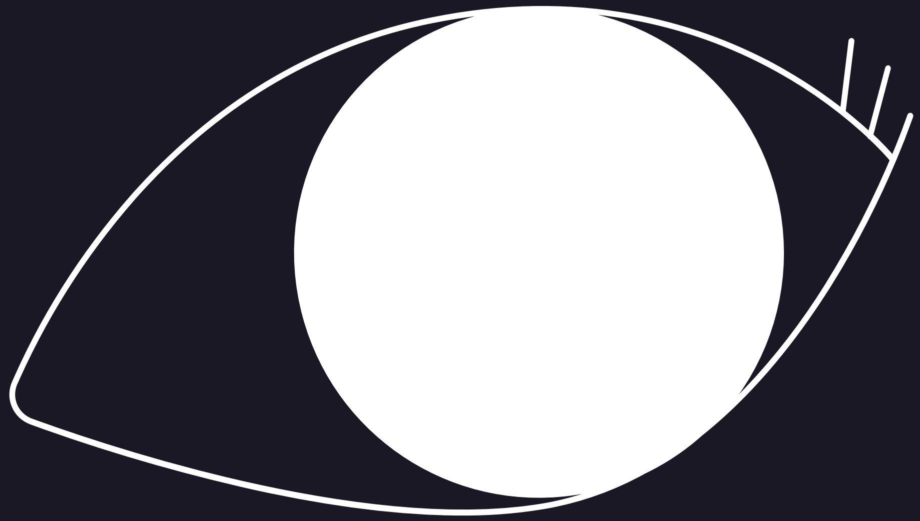
Hardening The Build

- Automate the build process to reduce the scope of intervention.
- Hardening the environment were builds are made.
- Enable the build to be verified by signature or programmatically.



Provenance

- It allows us to verify whether the various steps related to the source, build etc., have been done or not.
- A common language for us to communicate externally and internally about the state of the software.
- Allows consumers and vendors to require and show proof that they are meeting SLSA standards.



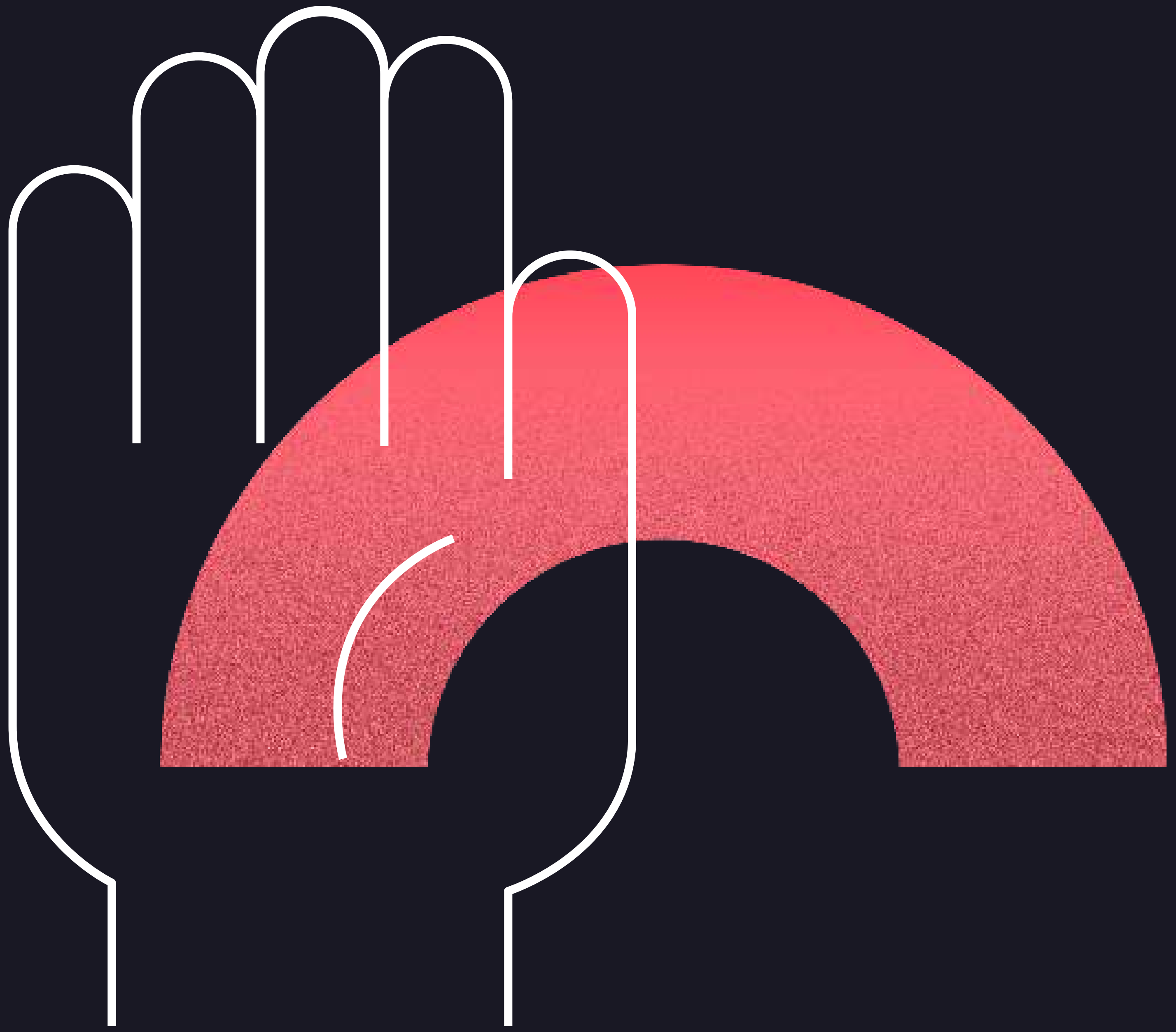
Out of Scope for SLSA

What SLSA Doesn't Cover for Source

- Isn't effective if high priviledged actors collude.
- Vulnerable code.
- Trick the reviewer into approving bad code.
- Compromise the source control system.

What SLSA Doesn't Cover for Build

- If builds are made from unofficial sources
    - Form, branch, tag, build steps etc.
- Vulnerability in the build platform.
- Usage of compromised builds.



Where to find more?

Visit SLSA website - slsa.dev or search "SLSA framework"

Look out for GUAC, goes nicely with SLSA

Thank you for listening!

Shameless Plug

I create short and long-form content on the Everything Cyber youtube channel.

Hands-on labs, thought experiments, interview with industry experts and more

→

Social links & slides