Supply Chain Security

Attacks and Mitigations

Taksh Medhavi

About Me

13 years in cybersecurity

Entrepreneur: Received funding and took exit in 3 years

Pentester: Web/Mobile/Network/Infra/Thick client/ Source code review/ Risk management/ Compliance/ Reverse engineer

6 years in Healthcare Cybersecurity: Philips/ Beckman Coulter/ Siemens

Mentored in **Stanford cybersecurity program** with highest rating by students

Null community member from **2016**

Meditator, Spiritual Seeker, **Art of living Devotee from 24 years and Teacher** from **14 years**

Adventure junkie: Biking, Cycling, Trekking, swimming

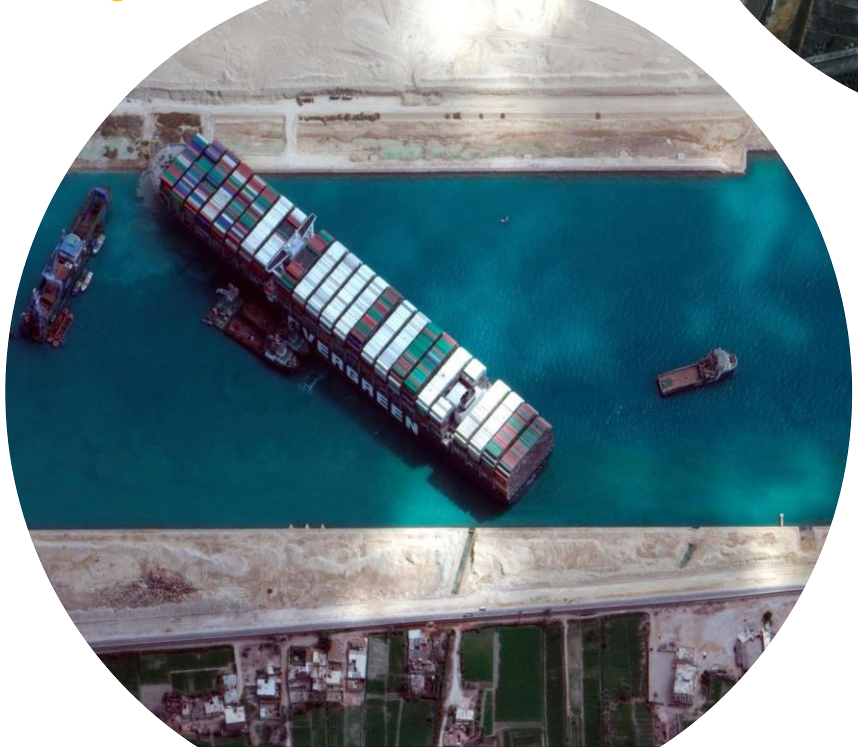
Purpose of this Session

- In an ever-evolving technological world, there is an **Increased Threat Landscape** followed by breaches.
- Key factors in a breach:
  - **Exploiting Trust**
  - **Evolving Threats**
  - **Stealth and Persistence**
  - **Disruption at Scale**
- There is an **Unprecedented Increase in the attack surface** in the domain of supply chain management.

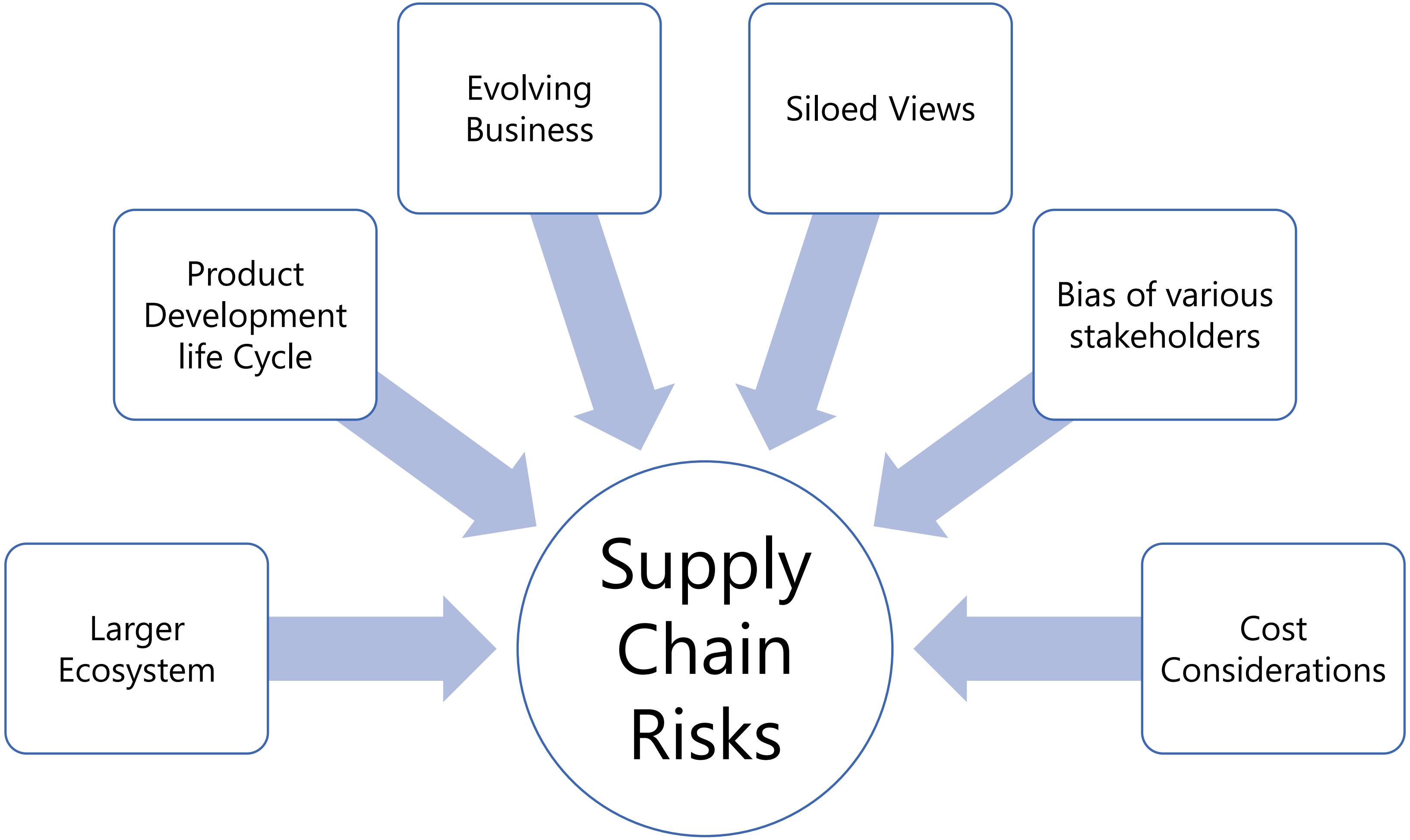
What is Supply Chain?

Supply chain includes all external suppliers, specialists, and resources that support the design, production, and delivery of products or solutions.

By working with experts, companies benefit from rapid growth, faster development, and quicker market releases, while achieving economies of scale, distributing risks, and enhancing product quality with niche expertise.



Supply Chain Landscape And Risk



1. **Disruption** of supply
2. Possibility of processes and supplies turning **Rogue**
3. Risks of **Unsustainable** support dependency
4. **Larger** risk landscape and **Complex** inventory management
5. **Regulatory and Compliance** issues

Type of Security Controls

Directive Controls	**Enforce** regulatory compliance
Deterrent Controls	**Discourage** malicious actions
Preventive Controls	**Block** unauthorized activities
Compensating Controls	Provide **alternate** protection
Detective Controls	**Identify** security breaches
Corrective Controls	**Remedy** security incidents
Recovery Controls	**Restore** system functionality

Attack Surfaces

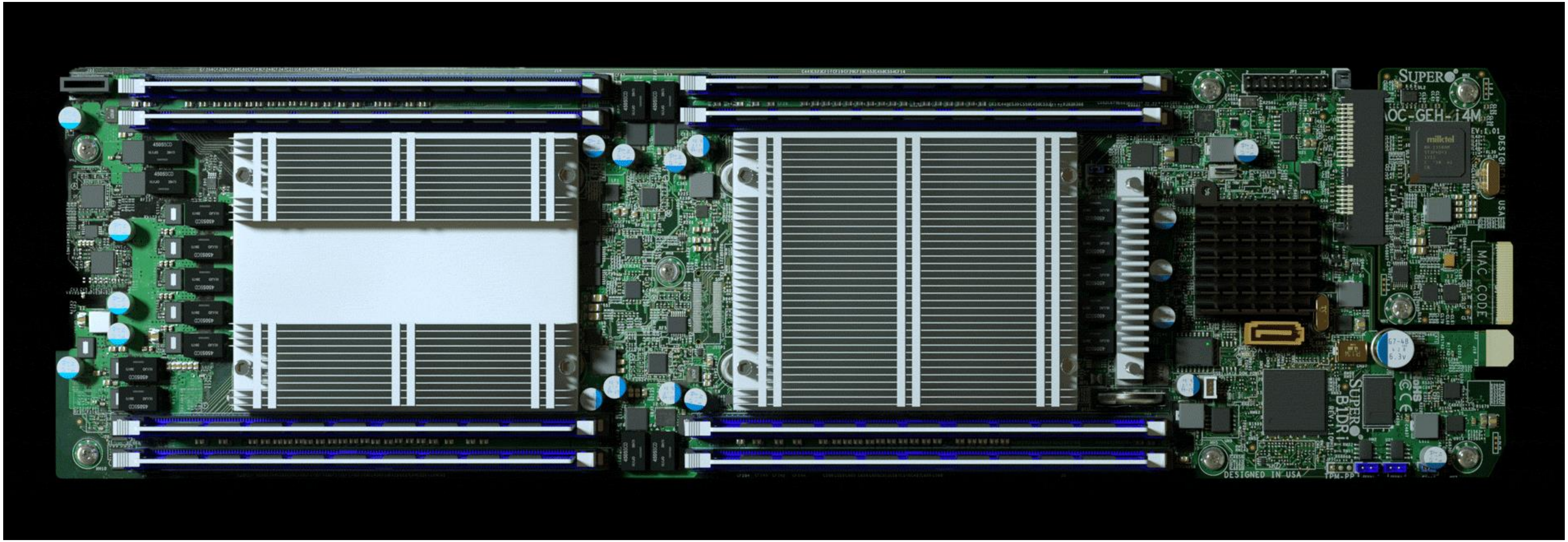
HARDWARE

SOFTWARE

AI/ML

Hardware Supply chain attacks

THE BIG HACK ATTACK (2018)



Chinese spies had inserted microchips into servers used by major companies like Amazon and Apple, potentially compromising data security.

Source: https://www.bloomberg.com/news/features/2018-10-04/the-big-hack-how-china-used-a-tiny-chip-to-infiltrate-america-s-top-companies

Hardware Supply chain Mitigation

Chassis intrusion detection switch

Silicon root of trust

Secure Device Identity and platform certificate

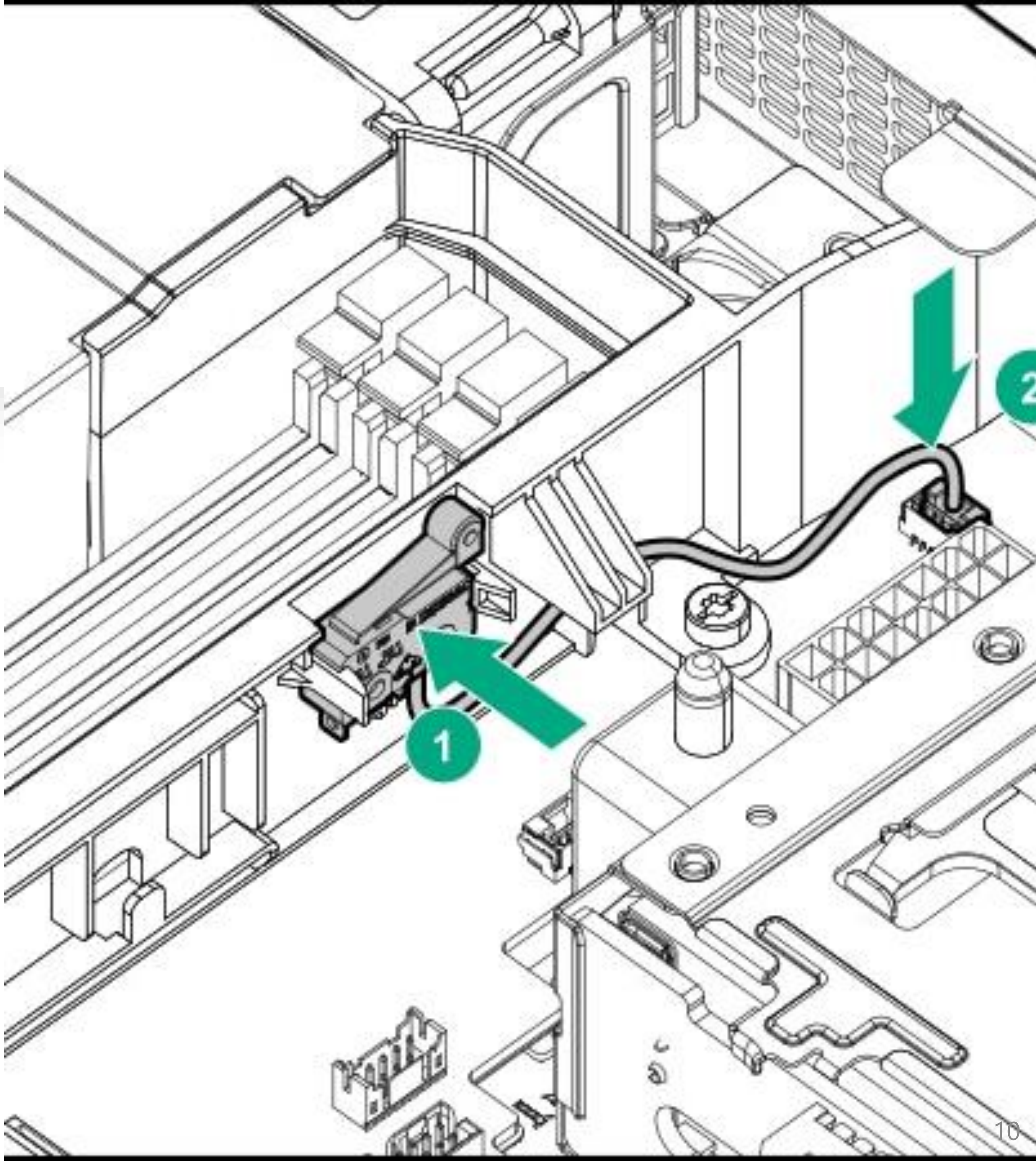
Blockchain asset birth certificate

Asset management as a service offered by hardware vendors

SPDM- Security Protocol and Data Model

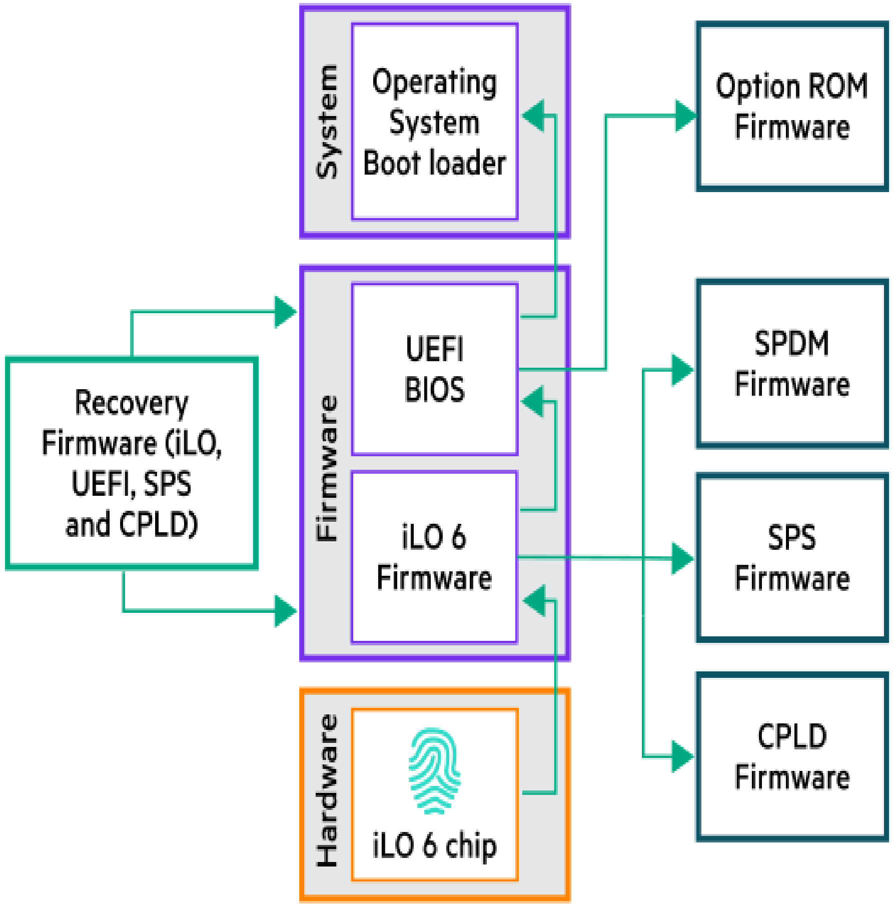
Chassis intrusion detection switch

- Any physical intrusion attempts are detected and logged
- BMC (Baseboard Management Controller) monitors the switch
  Alerting mechanism (syslog, SNMP, alert mail, etc.)
- Audit events are logged even if there is no power to the system
  Provides an ability to know if there were any attempts to open the lid during transit
- detects physical tampering of the infrastructure in supply chain transit



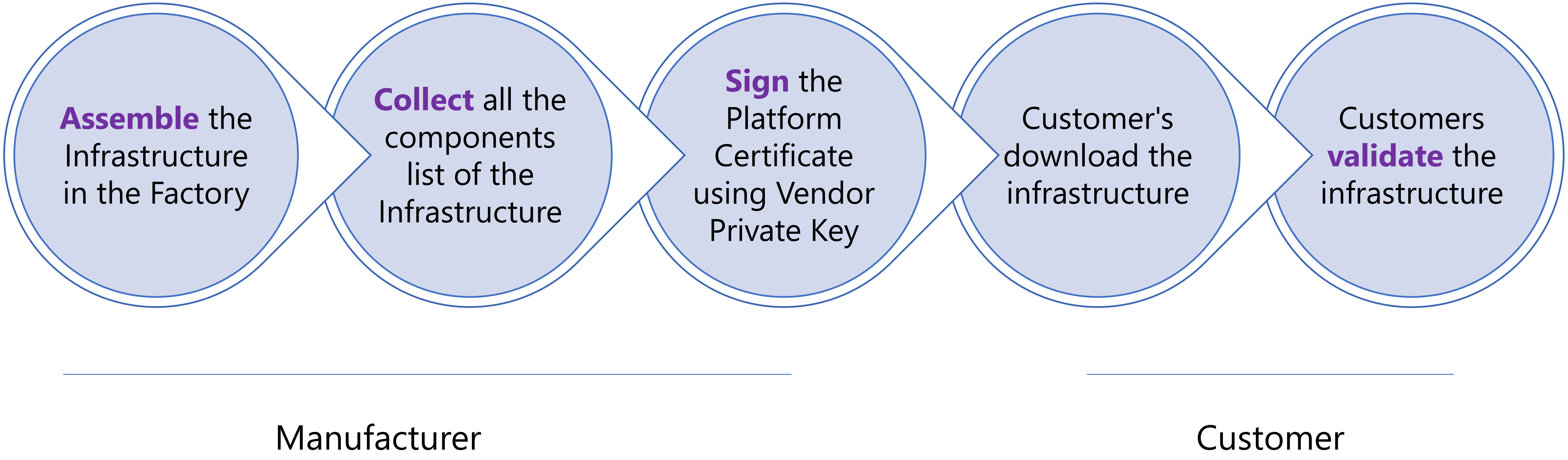
Silicon Root of Trust (S-RoT)

- Ensuring that security is intrinsic to the device and cannot easily be bypassed or manipulated by software attacks.
- Only trusted, signed, and verified firmware or code is executed. Any untrusted or altered code is detected and blocked. It enforces a chain of trust from the hardware to the operating system.
- The device can cryptographically prove that its hardware and software have not been tampered with, offering guarantees to other systems or users interacting with it.
- Examples: Trusted Platform Modules (TPM), Intel's Boot Guard, ARM’s TrustZone, and AMD’s Secure Processor are implementations of S-RoT



Source: https://support.hpe.com/hpesc/public/docDisplay?docId=sd00002198en_us&page=GUID-8D98A576-E52D-4C5B-A733-471CE9EDF90B.html&docLocale=en_US

Secure Device Identity and Platform Certificates



Assemble the Infrastructure in the Factory

Collect all the components list of the Infrastructure

Sign the Platform Certificate using Vendor Private Key

Customer's download the infrastructure

Customers **validate** the infrastructure

Manufacturer

Customer

SPDM (Security Protocol and Data Model)

SPDM provides a framework for secure device communication, authentication, and protection against potential tampering or malicious attacks.

Device Authentication

Ensures trusted device communication via cryptographic methods

Data Confidentiality and Integrity

Protects data with encryption and integrity checks

Attestation

Verifies device hardware and firmware integrity

Secure Firmware Update

Authenticates and secures firmware updates

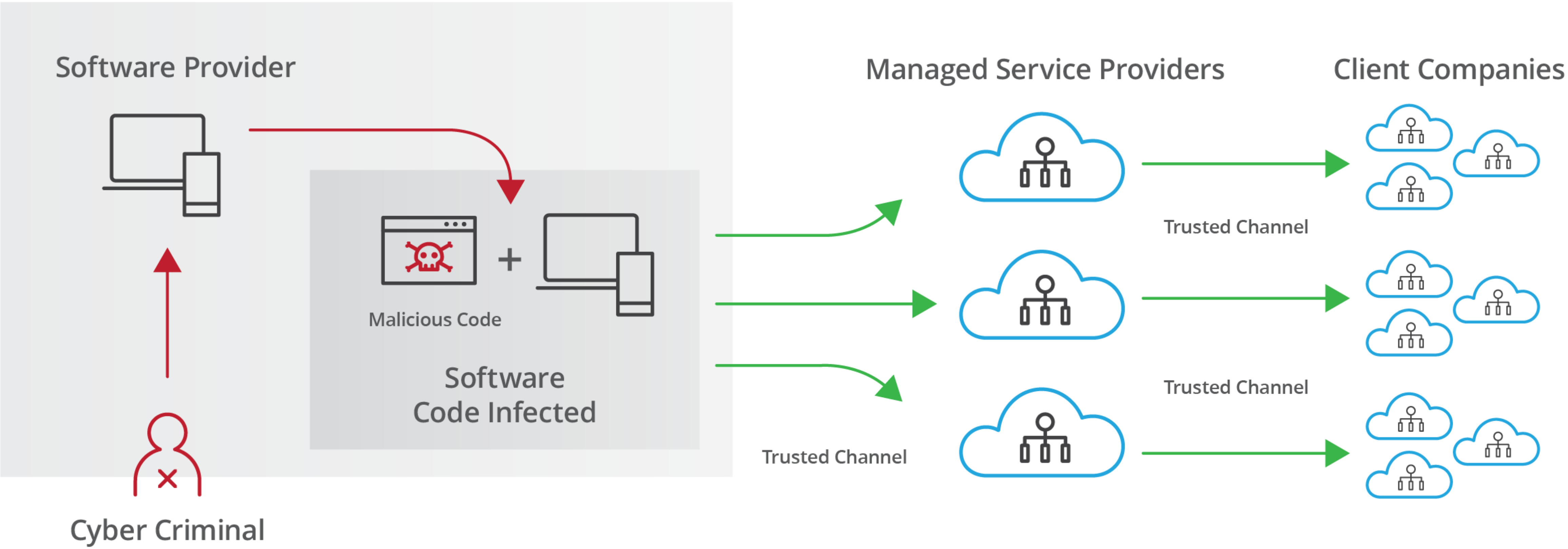
Standardized Communication

Ensures interoperability across diverse hardware

Platform Security Integration

Works with PFR(Platform Firmware Resiliency) and S-RoT (Silicon Root of Trust) for comprehensive security

Software Supply Chain Attack



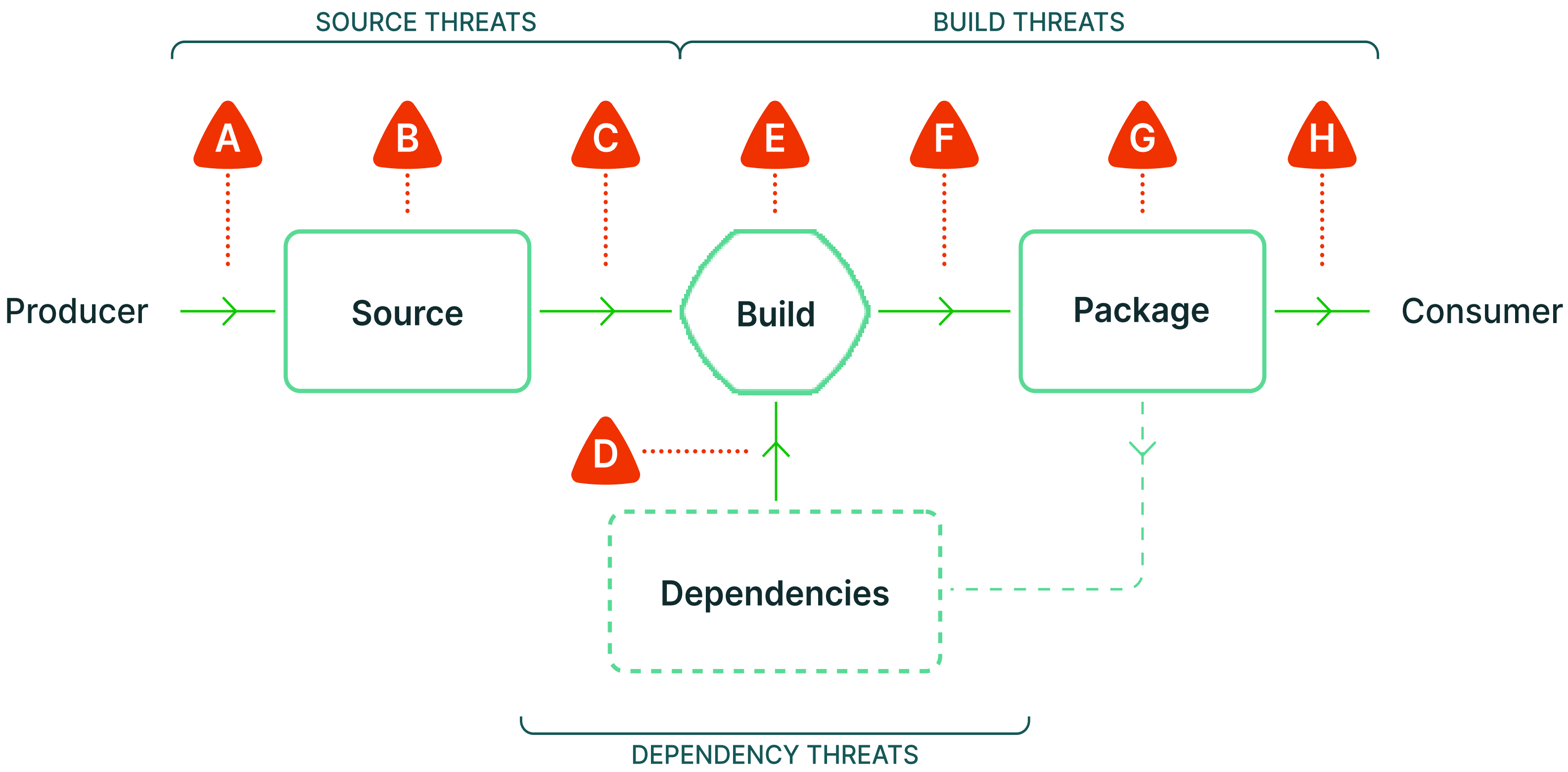
Source: https://www.linkedin.com/pulse/dont-let-your-vendors-put-you-risk-protect-against-supply-schweizer/

SUNBURST

SolarWinds Attack (2020)

- Initial Compromise: gained access to SolarWinds' internal systems & **targeted build environments** (phishing, vuln exploit, or using stolen credentials.)
- **Malicious Code Insertion**: integrated into source code, intentionally evade detection
- Software Update Mechanism: malicious code was included in these **updates with valid digital signature**.
- Malware gave hackers access to **customer IT systems**
- Command and Control (C2): malware established a **covert communication channel**
- Escalation and Exfiltration: move **laterally** & **data exfiltration**

Software Supply Chain Attack Surface



SOURCE THREATS

A Submit unauthorized change

B Compromise source repo

C Build from modified source

DEPENDENCY THREATS

D Use compromised dependency

BUILD THREATS

E Compromise build process

F Upload modified package

G Compromise package registry

H Use compromised package

Source:https://slsa.dev/spec/v1.0/threats-overview

Dependency Threats and Build Threats

Use a compromised build dependency

- The artifact uses libFoo and requires its source code to compile. The adversary compromises libFoo source repository and inserts malicious code. When your artifact builds, it contains the adversary's malicious code.

Use a compromised runtime dependency

- The artifact dynamically links libBar and requires a binary version to run. The adversary compromises libBar build process and inserts malicious code. When your artifact runs, it contains the adversary's malicious code.

Upload modified package

- Build with untrusted CI/CD
- Upload package without provenance
- Tamper with artifact after CI/CD
- Tamper with provenance

Use compromised package (Typo squatting)

- expres (missing an 's'), expresss (extra 's'), or expreess (double 'e')

Compromise build process

- Compromise project owner
- Compromise other build
- Steal cryptographic secrets
- Poison the build cache
- Compromise build platform admin

Compromise package registry

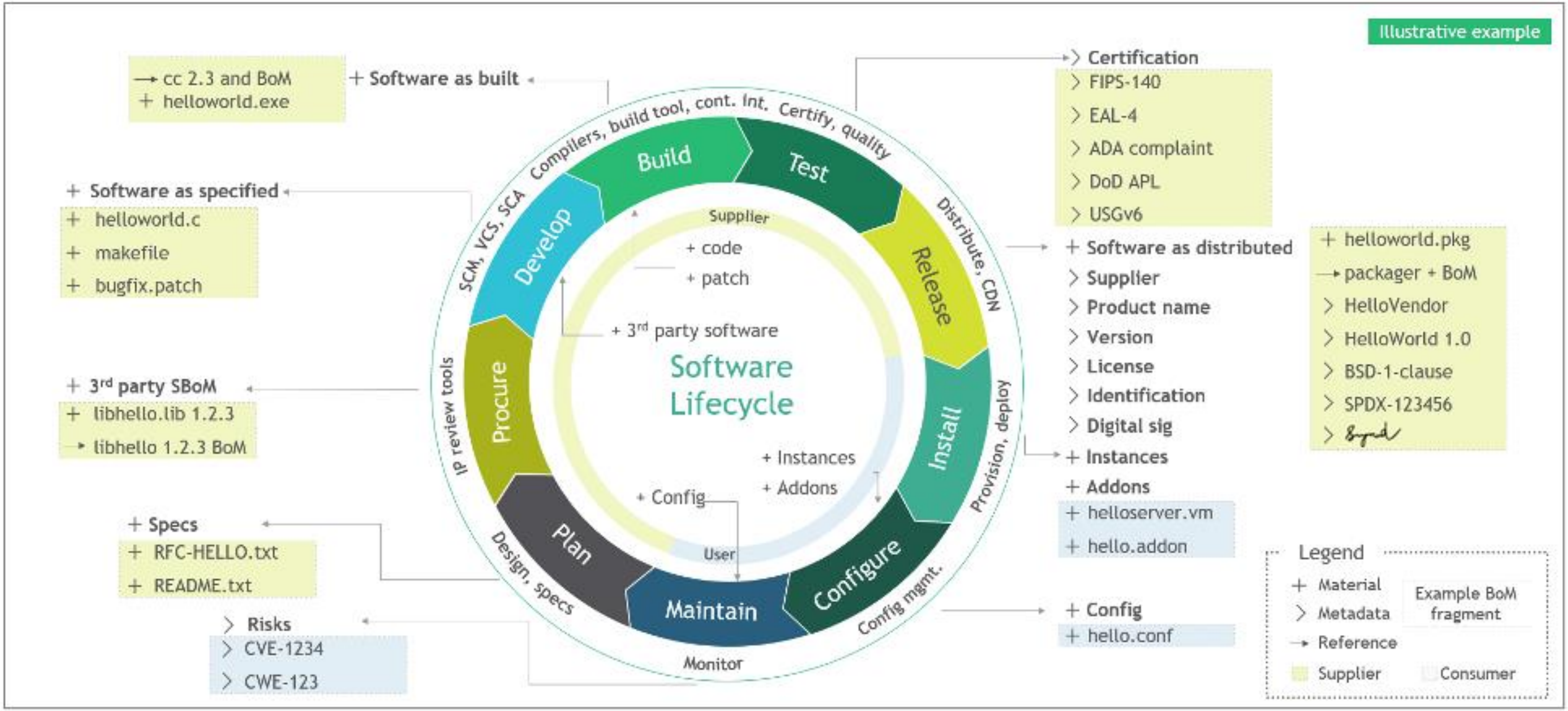
- Stop serving artifact
- Stop serving provenance

Security of Software Artifacts

Software artifacts are critical components that must be protected and verified to ensure the integrity, authenticity, and security of software throughout its lifecycle. e.g., Source Code, Executable Files, Libraries and Dependencies, Configuration Files, Build Artifacts, Installation Packages, Release Notes

Source Code Integrity	Ensure signed, version-controlled, audited code.
Securing Dependencies	Monitor and verify third-party libraries.
Build Artifacts Integrity	Secure binaries with signing and reproducibility.
Package and Distribution Security	Use code signing, encryption, trusted repositories.
Vulnerability Scanning and Patching	Regularly scan and securely update artifacts.
Version Control Metadata	Protect and validate version control metadata.
Transparency and Traceability	Track artifact origin and modifications.

Software Bill of Material (SBOM)



+ Software as built
- → cc 2.3 and BoM
- + helloworld.exe

+ Software as specified
- + helloworld.c
- + makefile
- + bugfix.patch

+ 3rd party SBoM
- + libhello.lib 1.2.3
- → libhello 1.2.3 BoM

+ Specs
- + RFC-HELLO.txt
- + README.txt

> Risks
- > CVE-1234
- > CWE-123

> Certification
- > FIPS-140
- > EAL-4
- > ADA complaint
- > DoD APL
- > USGv6

+ Software as distributed
> Supplier
> Product name
> Version
> License
> Identification
> Digital sig

- + helloworld.pkg
- → packager + BoM
- > HelloVendor
- > HelloWorld 1.0
- > BSD-1-clause
- > SPDX-123456
- >

+ Instances
+ Addons
- + helloserver.vm
- + hello.addon

+ Config
- + hello.conf

Legend
- + Material
- > Metadata
- → Reference
- Example BoM fragment
- 🟨 Supplier
- ⬜ Consumer

Source: https://www.nist.gov/itl/executive-order-14028-improving-nations-cybersecurity/software-security-supply-chains-softw
1

Software Supply Chain Security Strategy

Challenges:

Lack of visibility into infrastructure and environment

Dependency on third parties

Diversity of attack types: obfuscation, bitcoin miners, noisy techniques

Detection is **difficult**

Approach:

Understand the main **causes and sources** of attacks

Prepare defenses **preventative control and detective control**

Solution:

Regular **SAST** and **SCA** scans to identify build and source dependencies and build **robust SBOM**

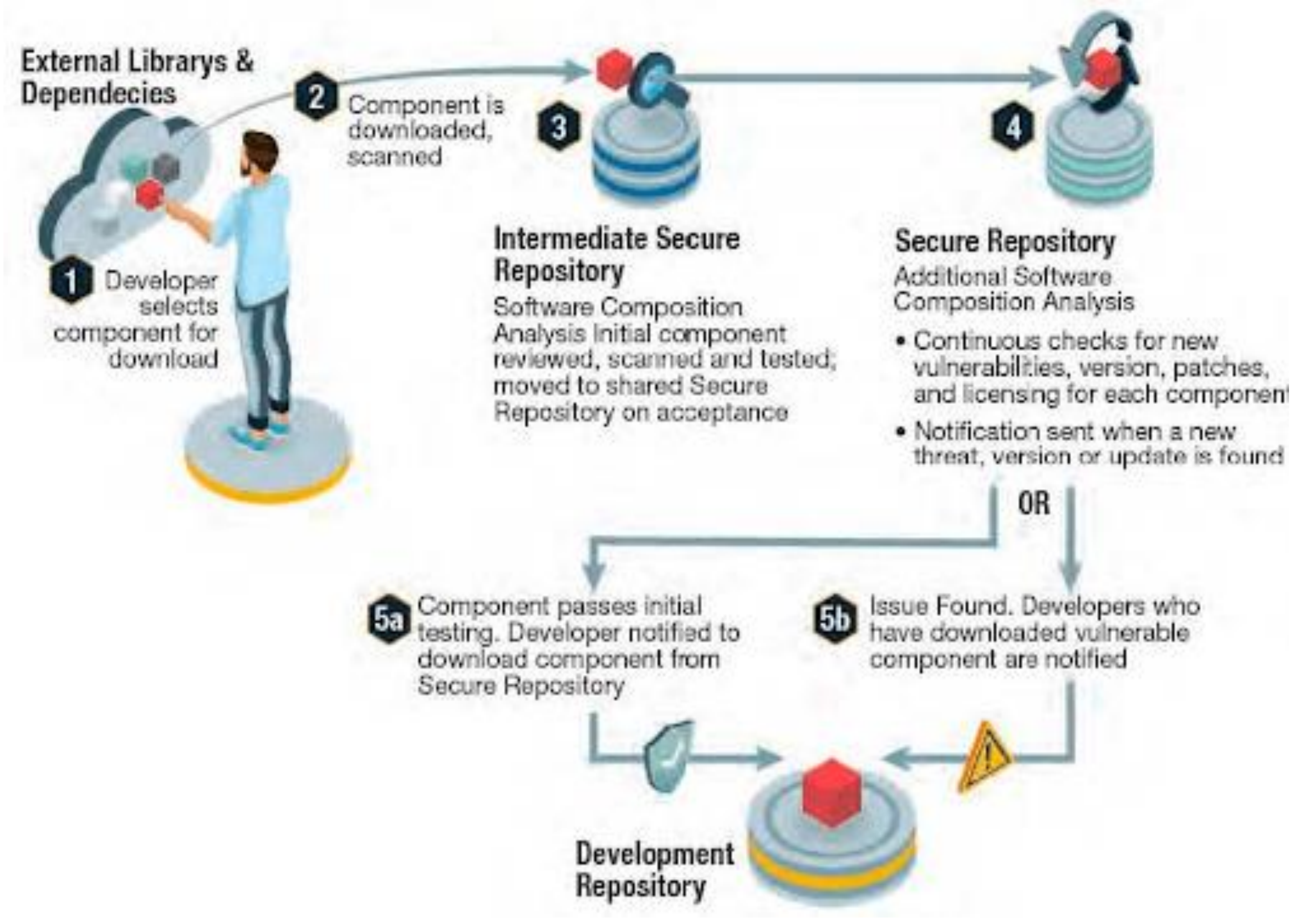
Governments and industry groups are developing new **standards**, **guidelines**, and **compliance frameworks**

Third Party Risk Management (**TPRM**)

The use of **AI** in the software development life cycle is gaining traction

Open-source Software Management

- Open-Source Software Management
  - **License**
  - **License Compliance**
  - **Export Controls**
- Creating and Maintaining a Company **Internal Secure Open-source Repository**
- Maintenance, Support and Crisis Management
- Vulnerability and Risk Assessment
- **SBOM** Creation, Validation and Artifacts



External Librarys & Dependecies

1 Developer selects component for download

2 Component is downloaded, scanned

3 Intermediate Secure Repository

Software Composition Analysis Initial component reviewed, scanned and tested; moved to shared Secure Repository on acceptance

4 Secure Repository

Additional Software Composition Analysis

- Continuous checks for new vulnerabilities, version, patches, and licensing for each component
- Notification sent when a new threat, version or update is found

OR

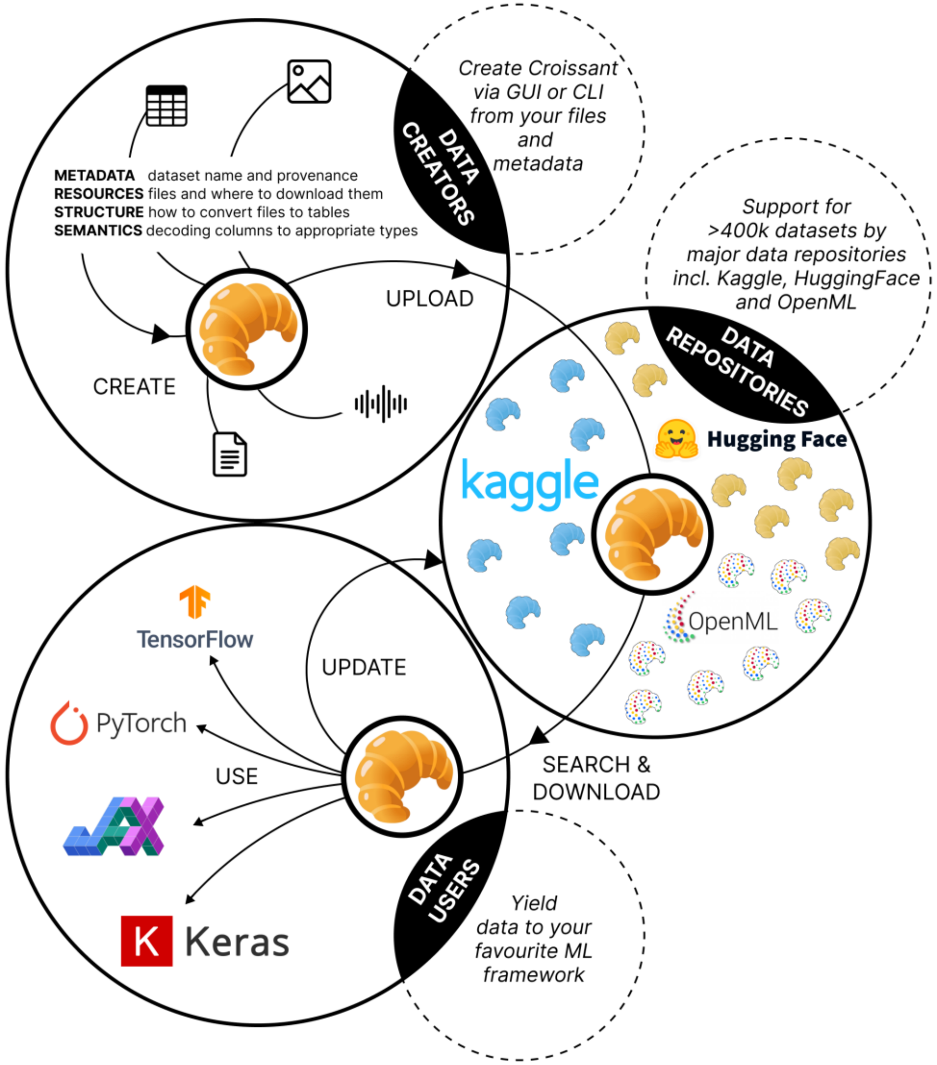
5a Component passes initial testing. Developer notified to download component from Secure Repository

5b Issue Found. Developers who have downloaded vulnerable component are notified

Development Repository

Source: https://www.cisa.gov/sites/default/files/publications/ESF_SECURING_THE_SOFTWARE_SUPPLY_CHAIN_DEVELOPERS.PDF

AI Supply chain security


Create Croissant via GUI or CLI from your files and metadata
DATA CREATORS
METADATA dataset name and provenance
RESOURCES files and where to download them
STRUCTURE how to convert files to tables
SEMANTICS decoding columns to appropriate types
CREATE
UPLOAD
Support for >400k datasets by major data repositories incl. Kaggle, HuggingFace and OpenML
DATA REPOSITORIES
kaggle
Hugging Face
OpenML
UPDATE
TensorFlow
PyTorch
USE
Keras
SEARCH & DOWNLOAD
DATA USERS
Yield data to your favourite ML framework

Source: The Croissant lifecycle and ecosystem

Guiding Principles

- **Protecting integrity** for the production systems which process, train, or serve AI models.
- **Cataloguing** provenance for all datasets and AI models.
- **Protecting models** against tampering and datasets against poisoning.
- Discovering and **patching or replacing buggy or vulnerable artifacts**
- Preventing accidental or malicious **data rights infringement**

- **Data Provenance:** Recording the **source of all data** examples used during training and evaluation of models
- **Model Provenance:** This metadata document **cryptographically** binds a model to the service account.
- **Explicit Provenance Logging:** Recording **lineage relationships** in I/O libraries such as data ingestion or model checkpointing libraries.
- **Infrastructure Log Harvesting:** AI workflows like training or data enrichment jobs provide a **manifest of inputs and outputs** and a sandbox restricts any access outside of the manifest while recording every input and output

Mitigation Strategy from Governance View

Curation focuses on **assessing and managing the risk of third-party software** from providers to consumer and its acceptability.

Creation focuses on **secure development and the protection of software artifacts** and the development pipeline.

Consumption validates **integrity** of software through **verification, provenance and traceability.**

Questions?

Thank You!

Linked In:

linkedin.com/in/takshmedhavi