Uptane: Open Source Summit Japan

Securing Software Updates and Supply Chains on Connected Vehicles

2022-12-7

Justin Cappos
New York University

Agenda

- *What Uptane is, what it does, and how it works*
- *Uptane prevents or deflects specific attacks*
- *Fundamental security assumptions and best practices*
- ***Break***
- *International standards and national and regional regulations*
- *Emerging critical issues*
- *Closing thoughts*

Please feel free to ask questions during the presentation

What Uptane is, What it Does and How it Works

Who Cares about Hacking Cars?

2015: Guys in tracksuits

Present: Attackers with nation-state level resources

Attacker Goals

Read the contents of updates to discover confidential information, reverse-engineer firmware, or identify security fixes to determine the fixed security vulnerability.

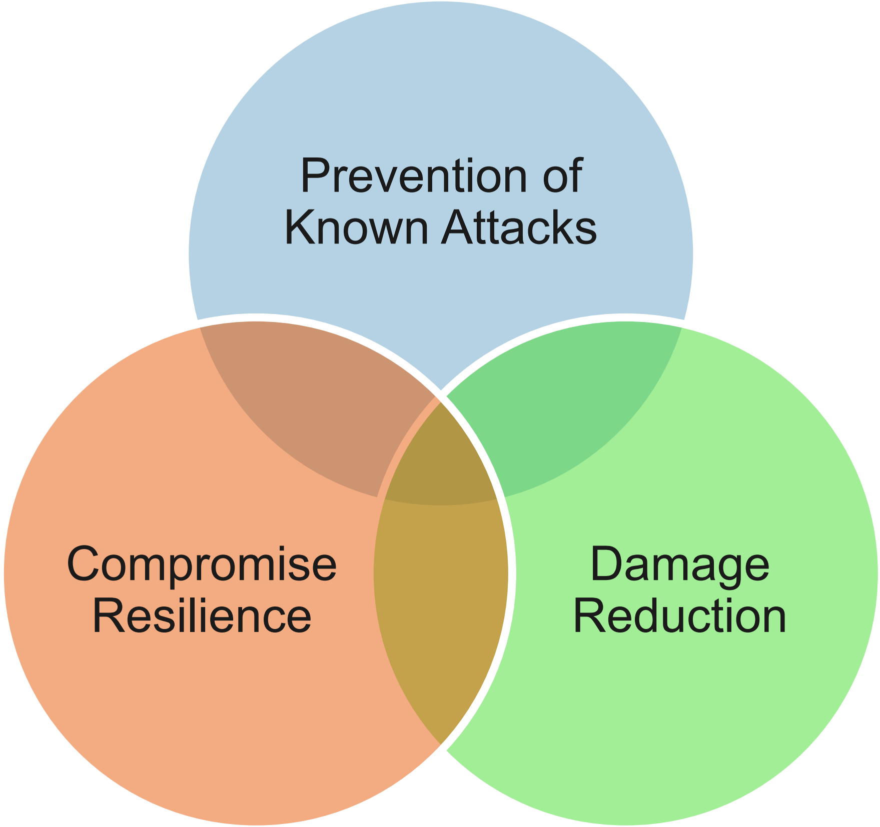
Deny installation of updates to prevent vehicles from fixing software problems.

Disrupt ECUs in the vehicle, denying use of the vehicle or of certain functions.

Control ECUs within the vehicle, and possibly the vehicle itself.

Uptane Goals

- Prevent known attacks on software update systems
- Provide compromise resilience and security by design
- Minimize damage from a compromised signing key or repository



Separation of Roles

Root

(Root of Trust)

Timestamp

(Freshness)

Snapshot

(Consistency)

Targets

(Authenticity)

Offline and Online Keys on Repos both fail

The OEM needs to tell ECUs which software is authentic and should be installed

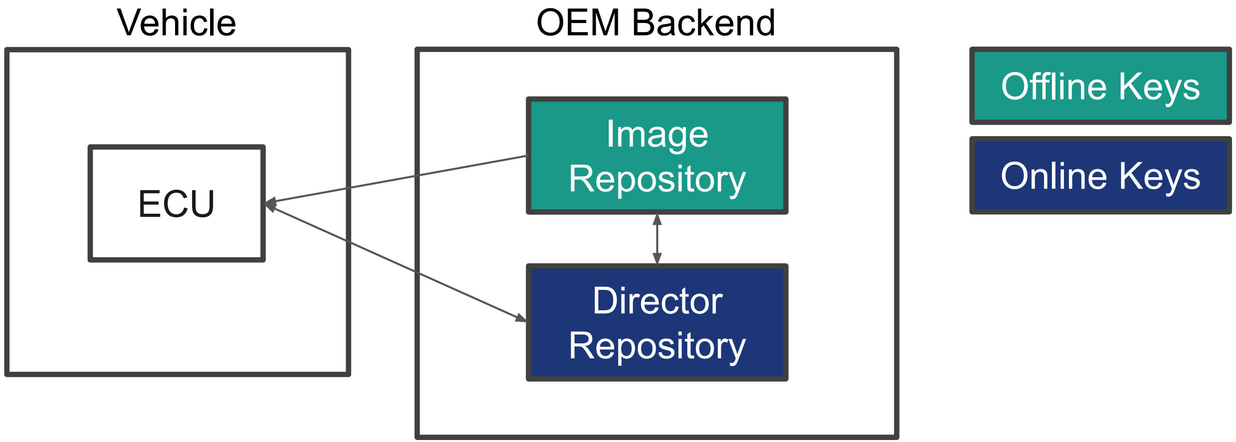
If the keys for authenticity are kept **online** (on the repository, even in a HSM, etc.):

- A repository hack compromises all users

If the keys that instruct what software to install are **offline** (e.g., Yubikey kept in a locked desk drawer):

- It is completely unusable, because the key needs to be used repeatedly.

Offline and Online Keys

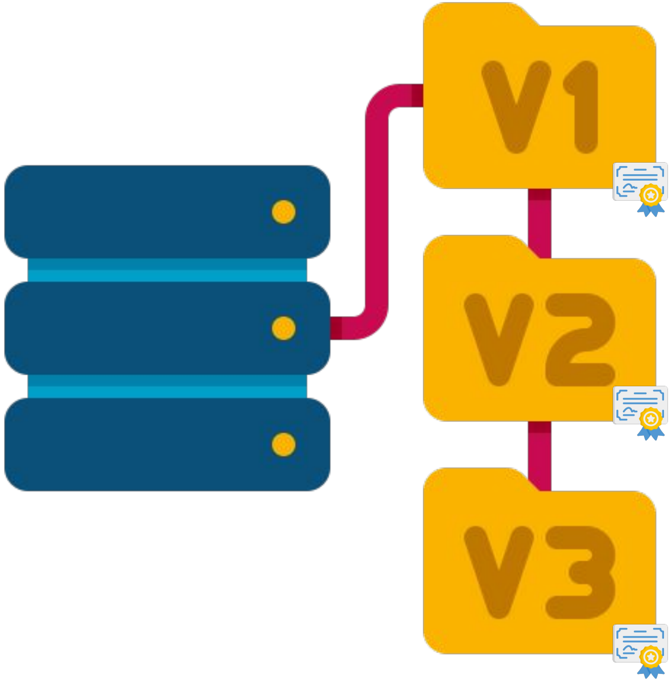


Uptane uses two repositories to provide OEMs with both **security** and **flexibility**!

Image Repository

Authenticity of software images

1) Human managed
2) Offline keys
3) Infrequent updates
4) Provides flexible delegation for image signing

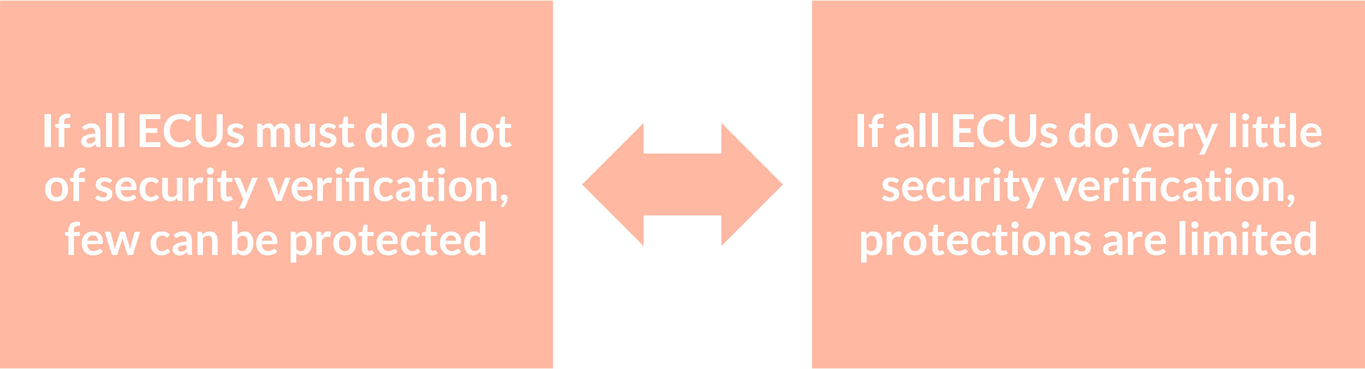


Director Repository

Which software should be installed

1) Automated
2) Online Keys
3) Frequent Requests
4) Generates signed vehicle specific manifest
5) Let’s OEM control what images are installed
6) Works in coordination with a vehicle configuration database

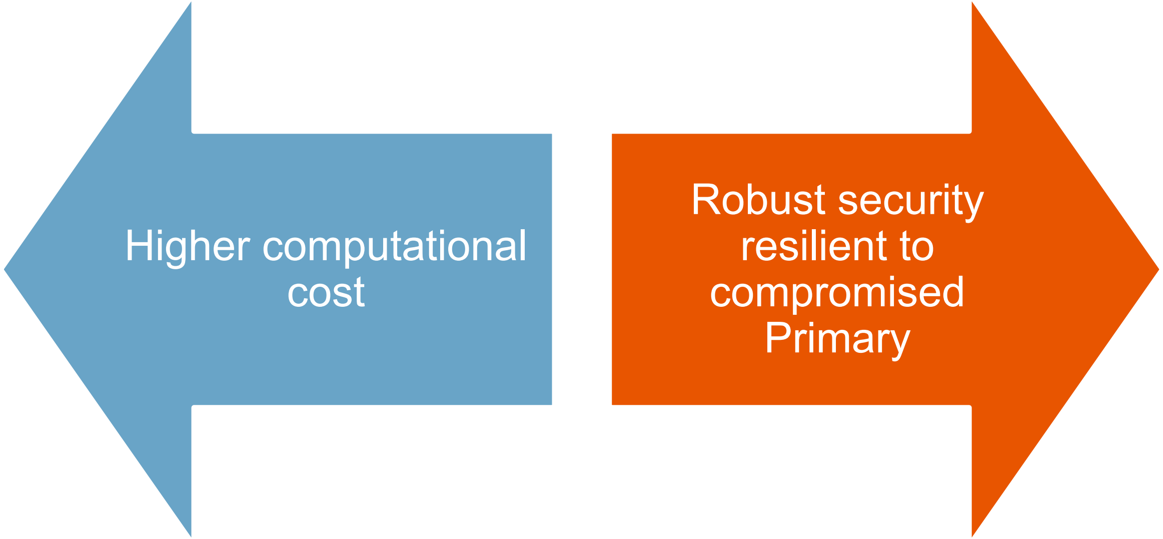
How much work should an ECU do?



If all ECUs must do a lot of security verification, few can be protected

If all ECUs do very little security verification, protections are limited

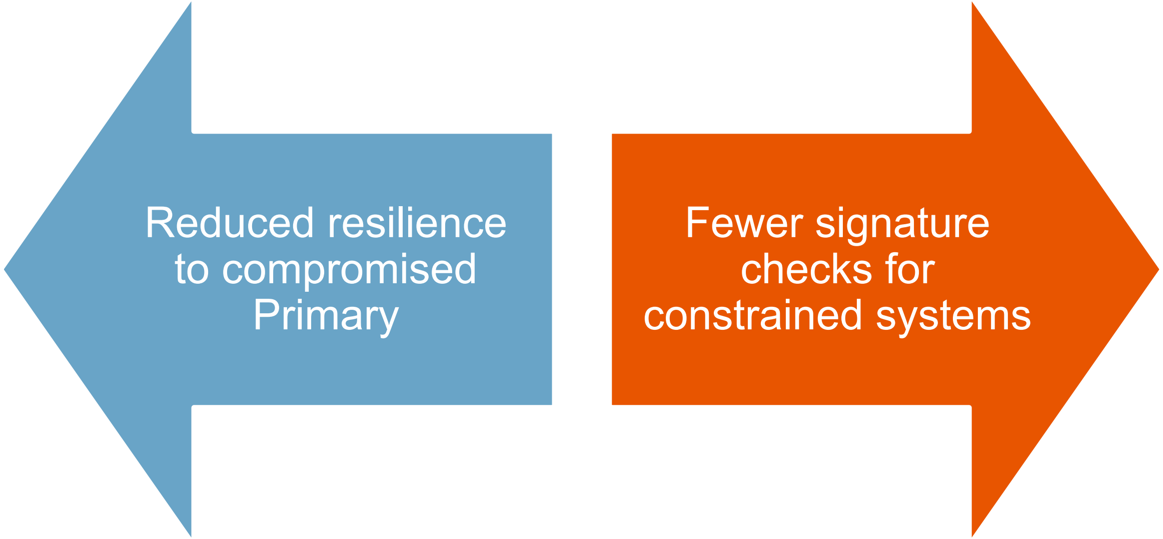
Full Verification of Secondaries



Higher computational cost

Robust security resilient to compromised Primary

Partial Verification of Secondaries



Reduced resilience to compromised Primary

Fewer signature checks for constrained systems

Uptane Ecosystem

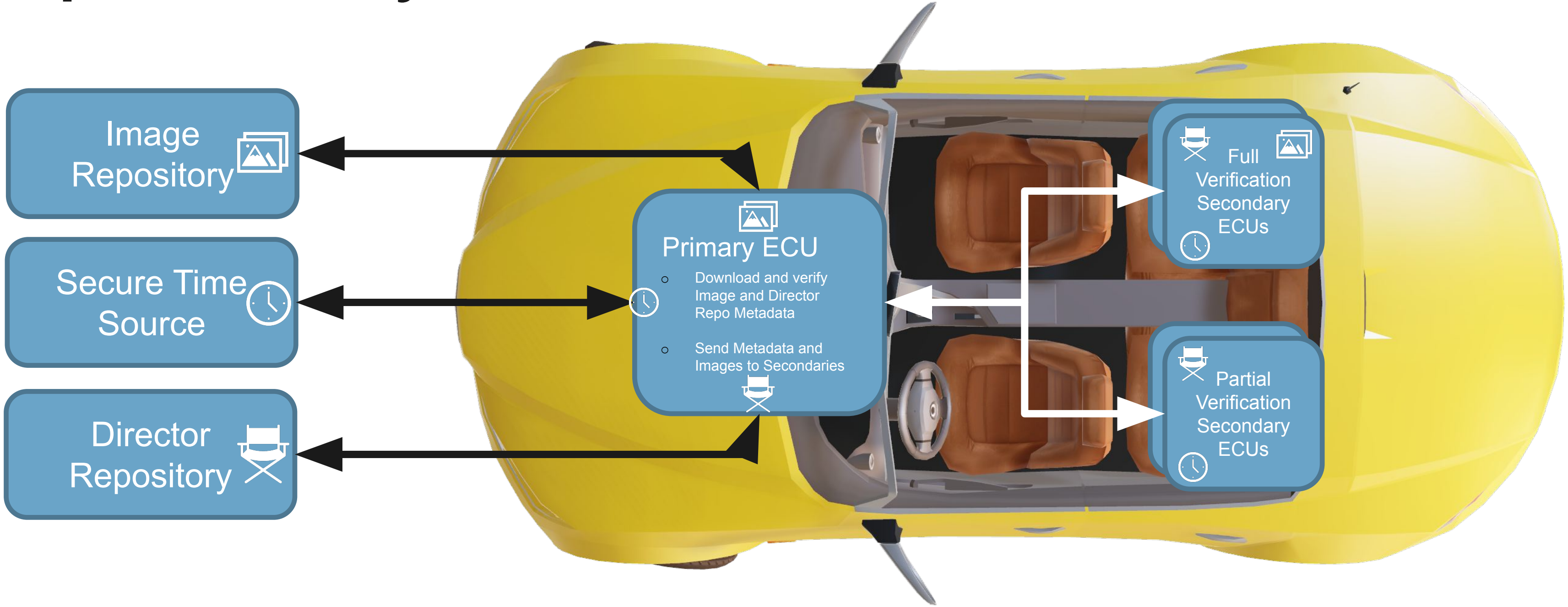


Image Repository

Secure Time Source

Director Repository

Primary ECU

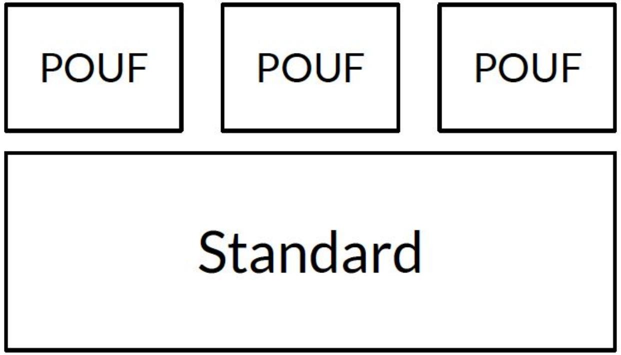
- Download and verify Image and Director Repo Metadata
- Send Metadata and Images to Secondaries

Full Verification Secondary ECUs

Partial Verification Secondary ECUs

Uptane POUFs (Protocols, Operations, Usage, and Formats)

- A profile layer on top of the Uptane Standard
- Allows for interoperable Uptane implementations
- Describes an implementation
  - Choices made from the Uptane Standard and Deployment Considerations
  - Networking information, file storage and data definitions



POUF

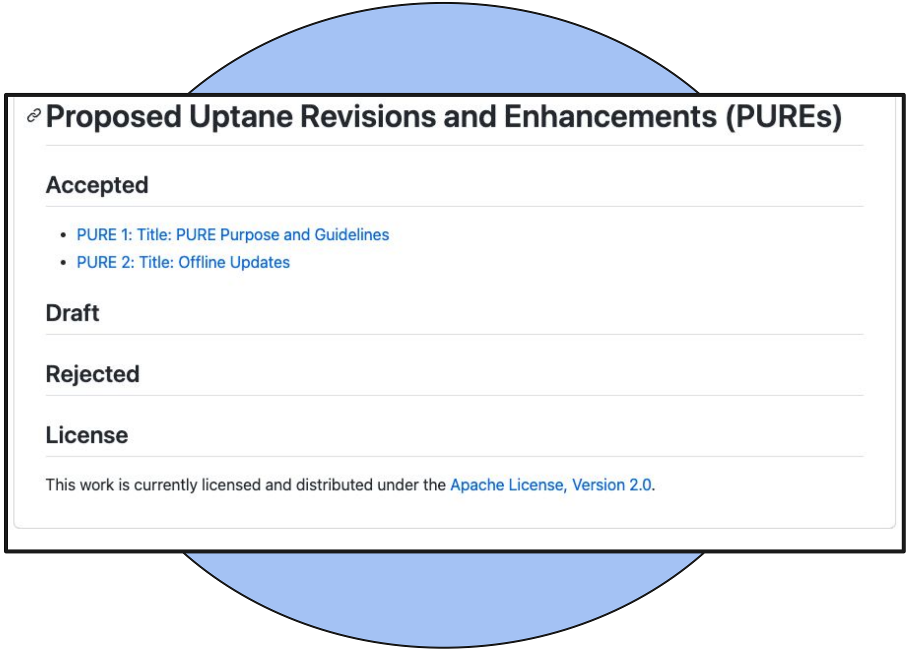
POUF

POUF

Standard

PUREs

- Modeled on TAPs from The Update Framework
- A formal method for the community to propose additions or modifications of the Uptane Standard
- Two PUREs approved to date



Proposed Uptane Revisions and Enhancements (PUREs)

Accepted

- PURE 1: Title: PURE Purpose and Guidelines
- PURE 2: Title: Offline Updates

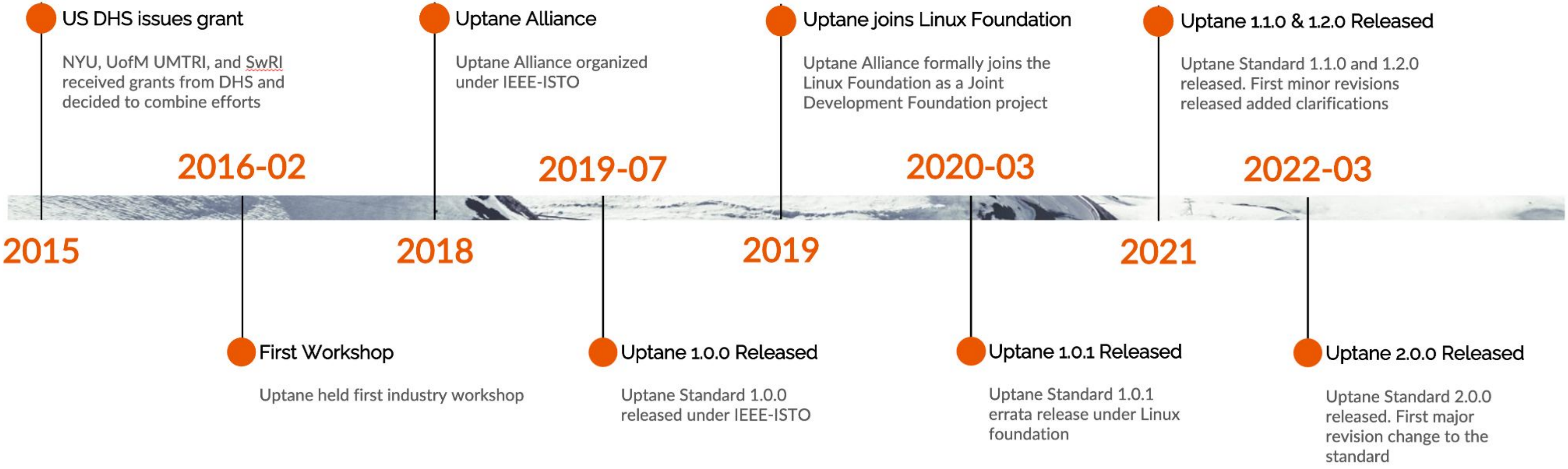
Draft

Rejected

License

This work is currently licensed and distributed under the Apache License, Version 2.0.

Timeline for Uptane Development



2015

US DHS issues grant

NYU, UofM UMTRI, and SwRI received grants from DHS and decided to combine efforts

2016-02

First Workshop

Uptane held first industry workshop

2018

Uptane Alliance

Uptane Alliance organized under IEEE-ISTO

2019-07

Uptane 1.0.0 Released

Uptane Standard 1.0.0 released under IEEE-ISTO

2019

Uptane joins Linux Foundation

Uptane Alliance formally joins the Linux Foundation as a Joint Development Foundation project

2020-03

Uptane 1.0.1 Released

Uptane Standard 1.0.1 errata release under Linux foundation

2021

Uptane 1.1.0 & 1.2.0 Released

Uptane Standard 1.1.0 and 1.2.0 released. First minor revisions released added clarifications

2022-03

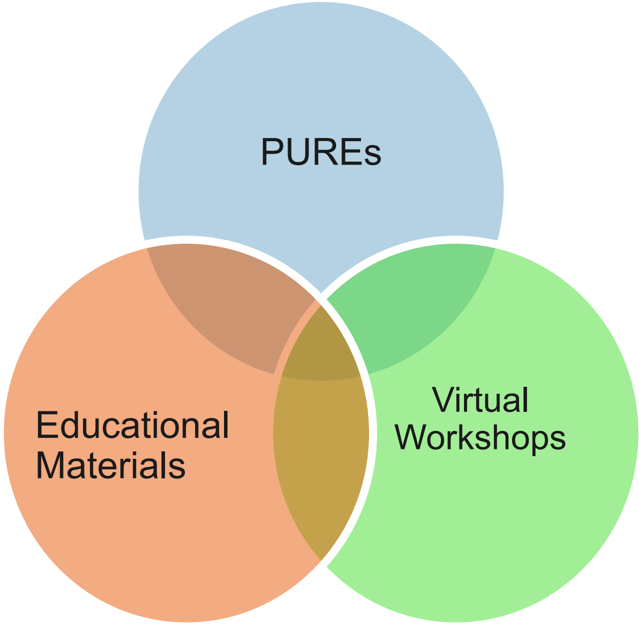
Uptane 2.0.0 Released

Uptane Standard 2.0.0 released. First major revision change to the standard

Outreach Beyond Standards

Recent Uptane initiatives in addition to issuing Standards:

- Establishing policy for accepting proposed contributions to the Standard (PUREs)
- Publishing whitepapers, videos, and tutorials to address new or emerging areas of concern in cybersecurity
- Sharing Uptane stories across borders

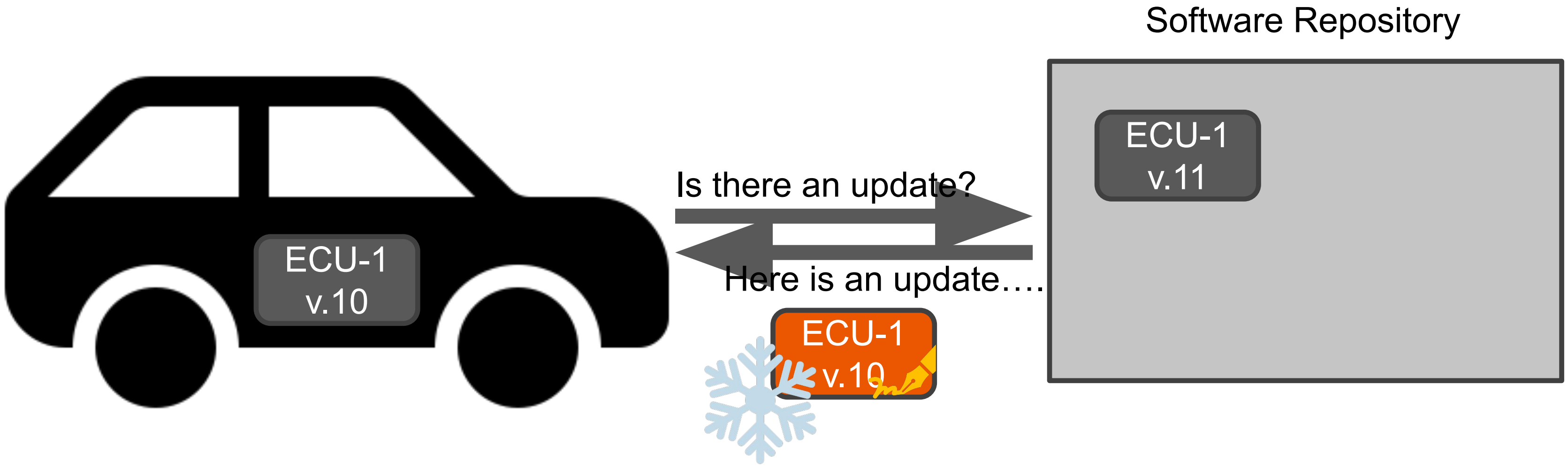


Uptane Mitigates Specific Attacks

Uptane protects against four categories of attacks

- Read updates
- Deny updates
- Deny functionality
- Loss of control

Freeze Attack



Software Repository

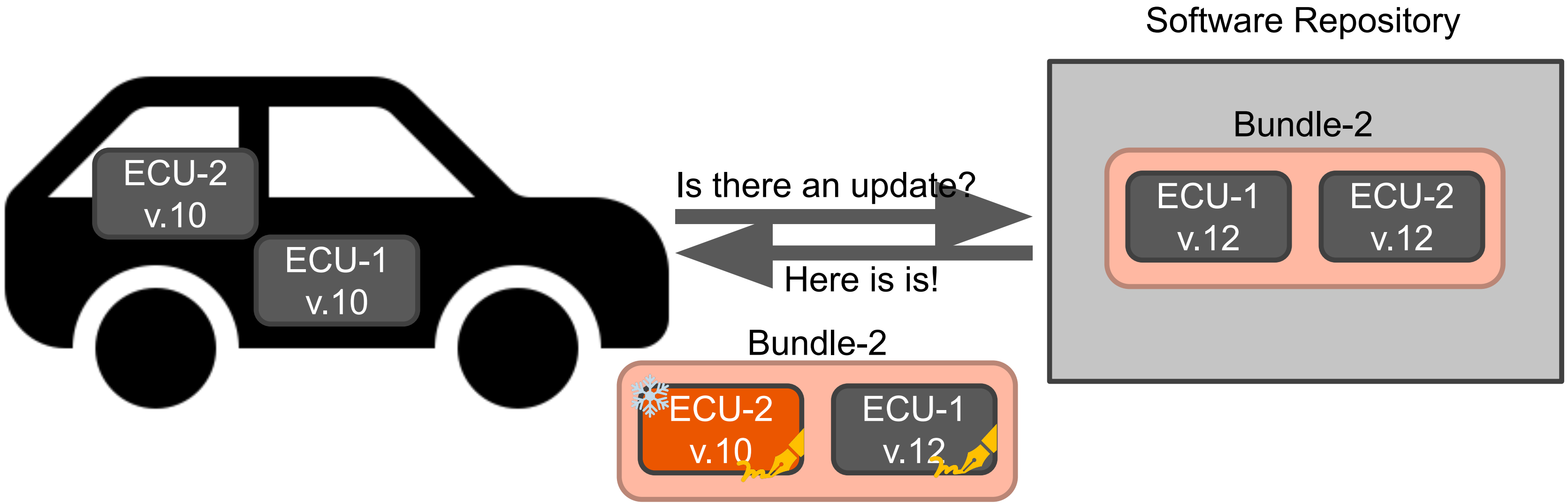
Is there an update?

Here is an update….

Uptane Protections: Freeze Attack

- The Timestamp metadata includes a timestamp with a short expiration date
- Vehicle can detect that the timestamp is invalid

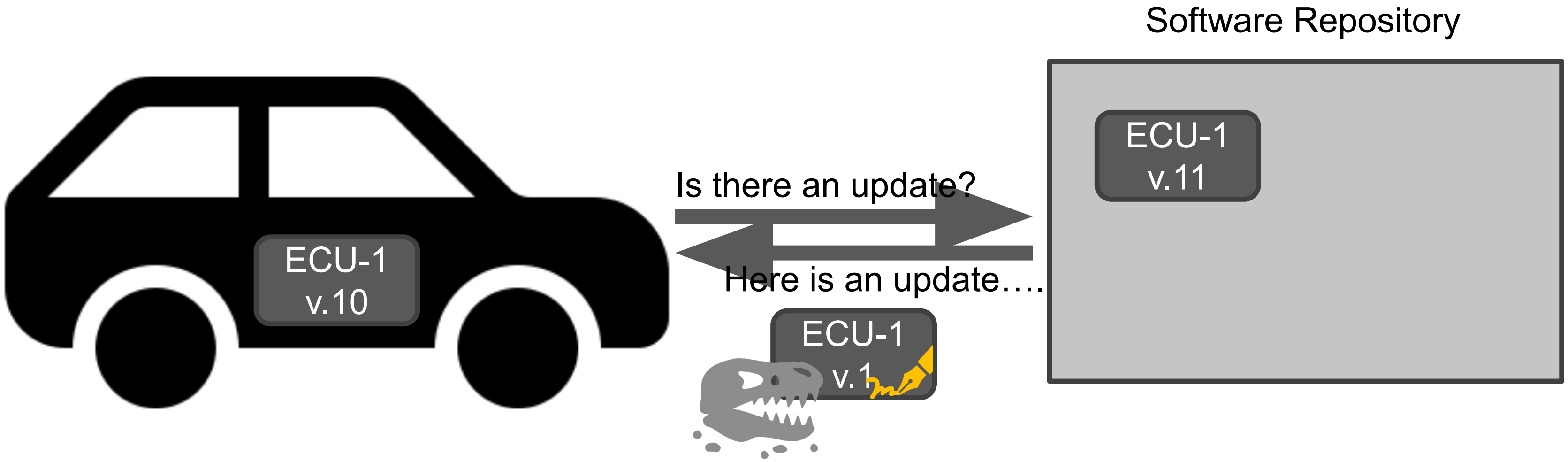
Partial Freeze Attack



Uptane Protections: Partial Freeze Attack

- Snapshot metadata lists all current targets metadata
- Timestamp signs digest of snapshot

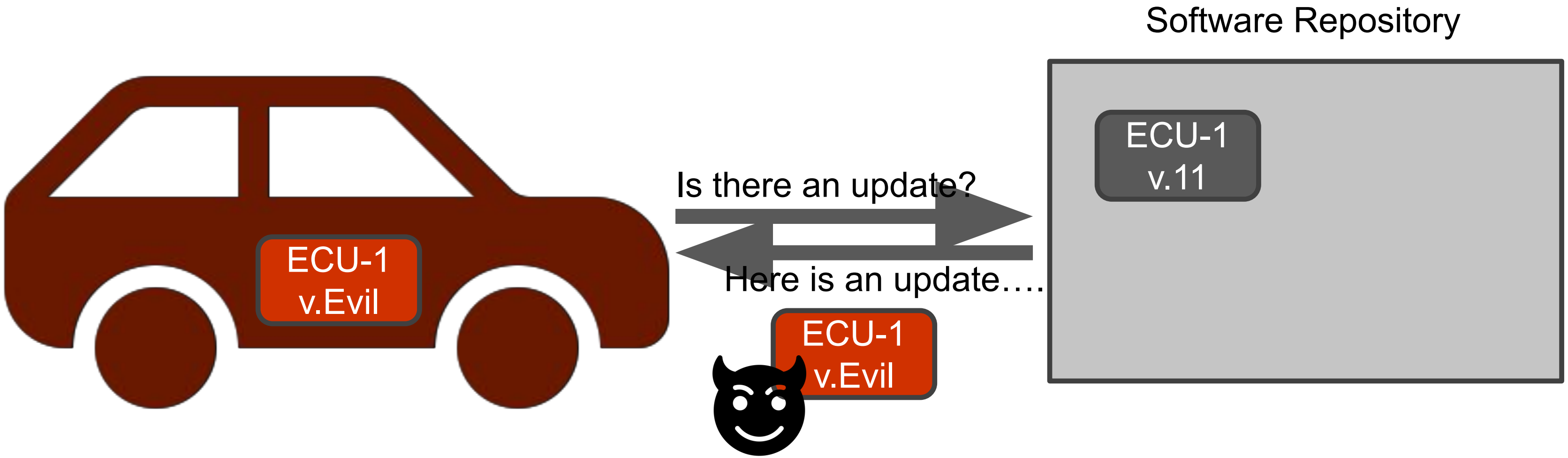
Rollback Attack



Uptane Protections: Rollback Attack

- Snapshot metadata lists all current targets metadata
- Vehicle checks that all versions numbers are strictly increasing

Arbitrary Software Attack



Uptane Protections: Arbitrary Software Attack

- Targets metadata signs the contents of all updates
- Signed by both repositories
    - Image repository uses offline keys
    - Director repository directs updates

TLP: WHITE

Uptane

Uptane Community Considers Additional Device Security Issues

1) Securing interfaces

2) Choosing strong crypto algorithms

3) Ensuring entropy

4) Protecting keys

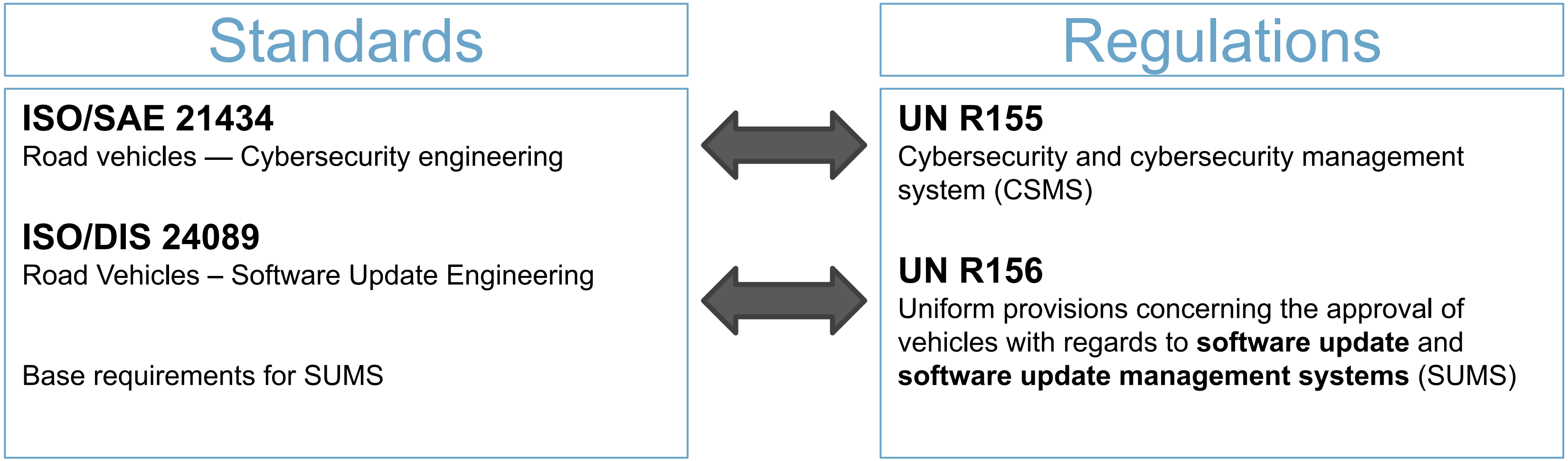
5) Preventing data leaks

6) Securing the time source

Break

Emerging Critical Issues

Alignment with Standards and Regulations



Standards

ISO/SAE 21434
Road vehicles — Cybersecurity engineering

ISO/DIS 24089
Road Vehicles – Software Update Engineering

Base requirements for SUMS

Regulations

UN R155
Cybersecurity and cybersecurity management system (CSMS)

UN R156
Uniform provisions concerning the approval of vehicles with regards to **software update** and **software update management systems** (SUMS)

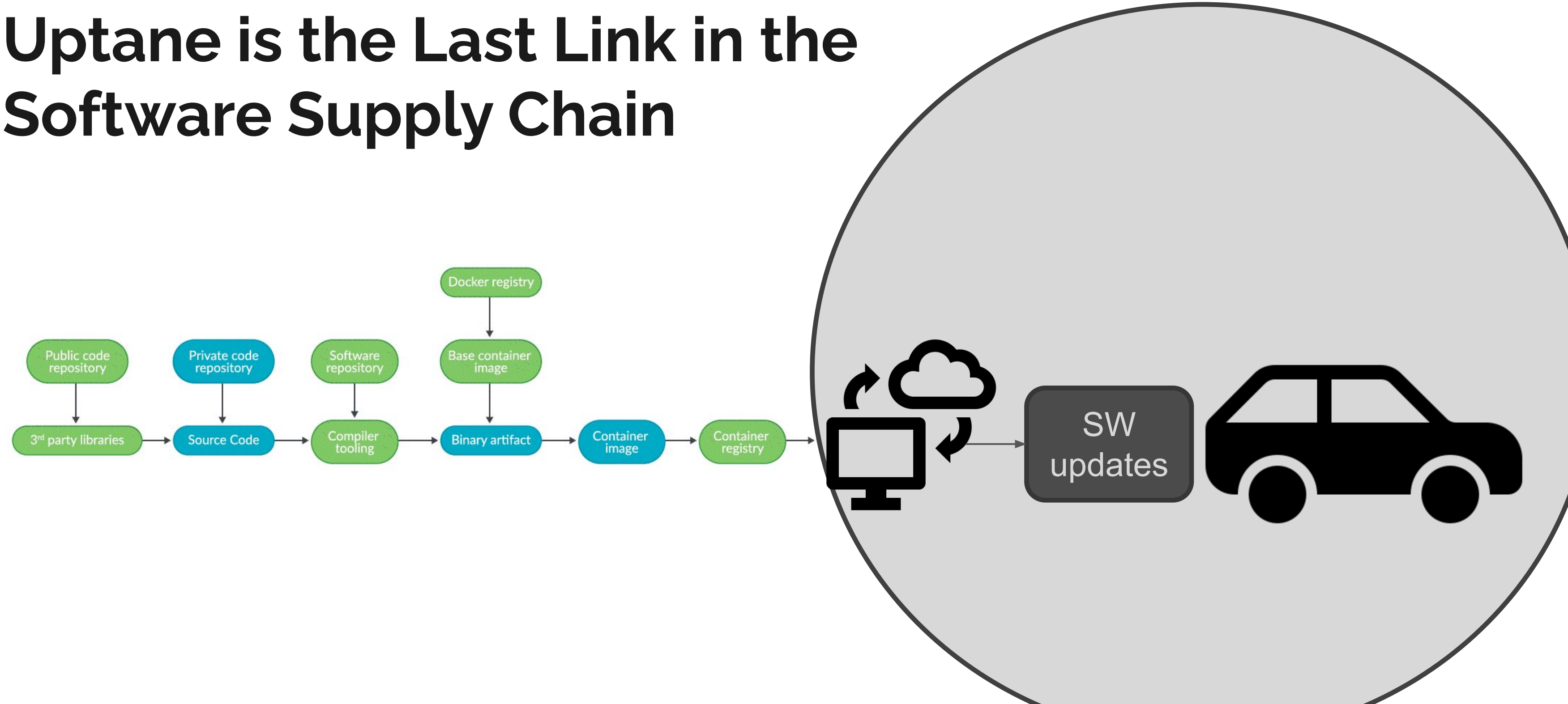
Software Bill of Materials (SBOM)

SBOM is a nested inventory of software components

SBOMs are a key building block in software supply chain security

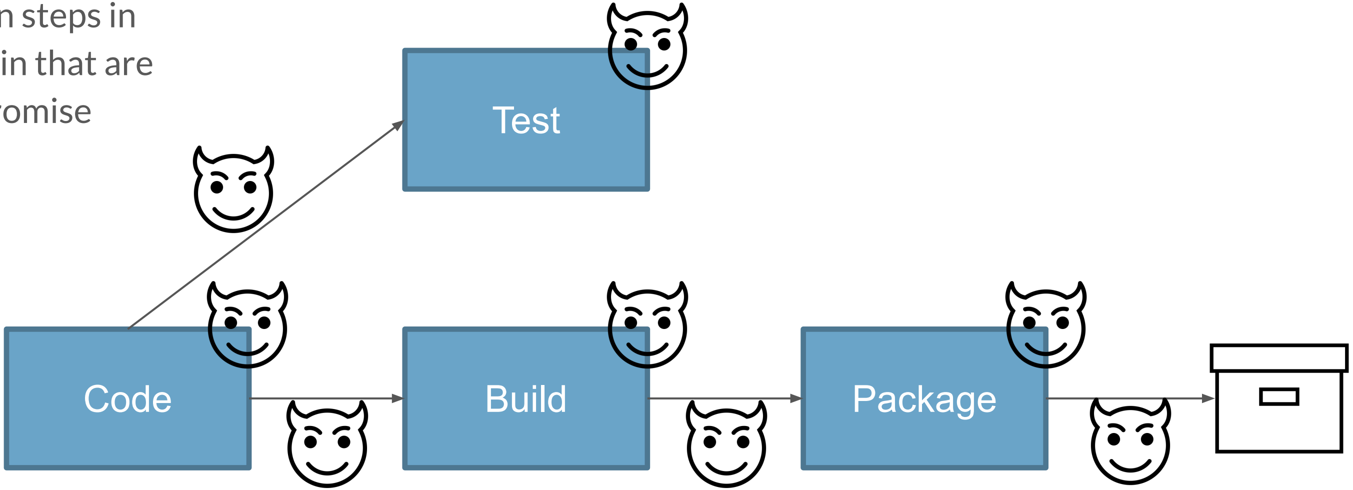
Auto-ISAC is creating a best practice document about automotive SBOMs

Uptane is the Last Link in the Software Supply Chain



Securing the Software Supply Chain

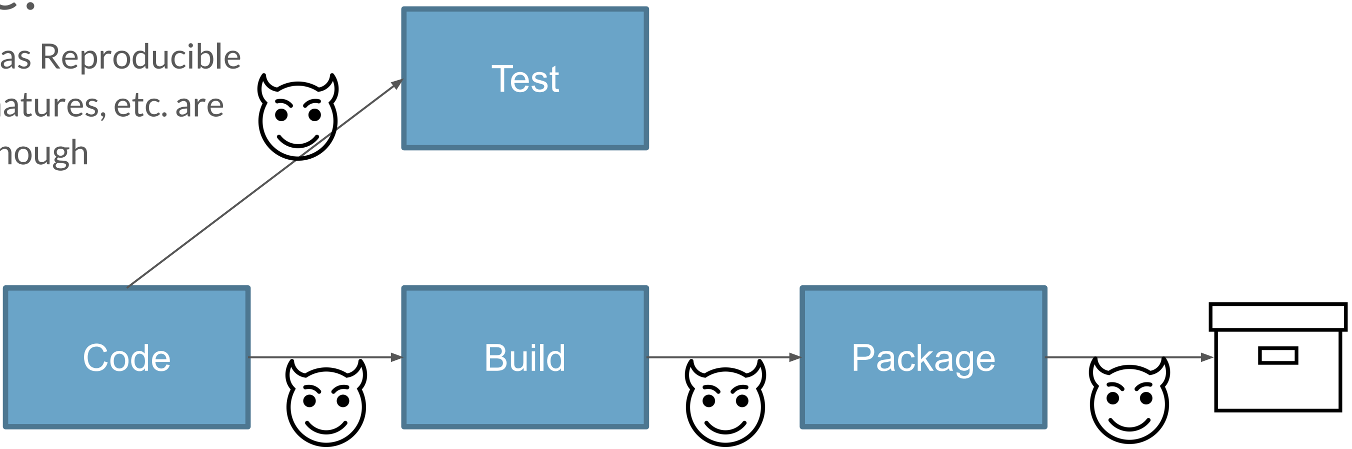
Aim to verifiably sign steps in software supply chain that are vulnerable to compromise



Gaps Between Steps in Supply Chain?

Compliance?

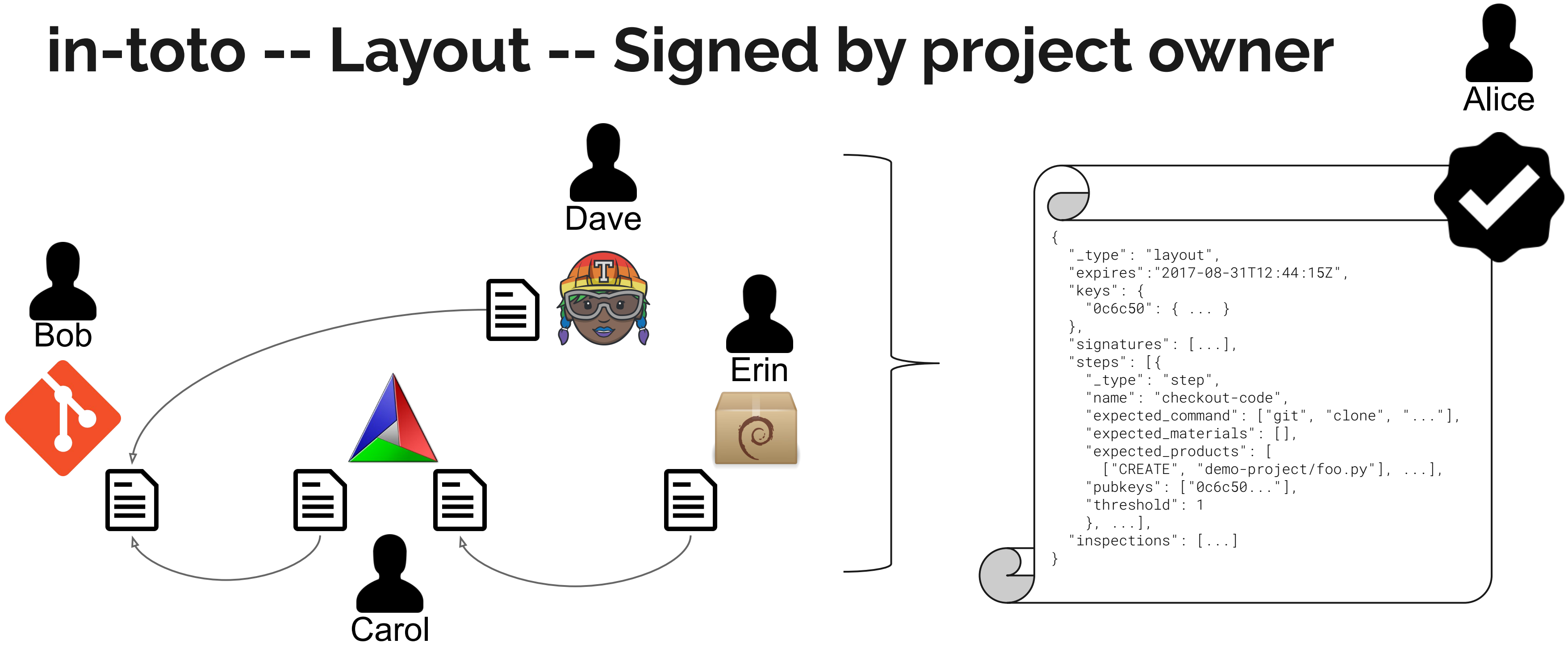
Spot solutions such as Reproducible Builds, Commit Signatures, etc. are necessary but not enough



in-toto

- Verifiably define the steps of the software supply chain
- Verifiably define the authorized actors
- Guarantee everything happens according to definition and nothing else

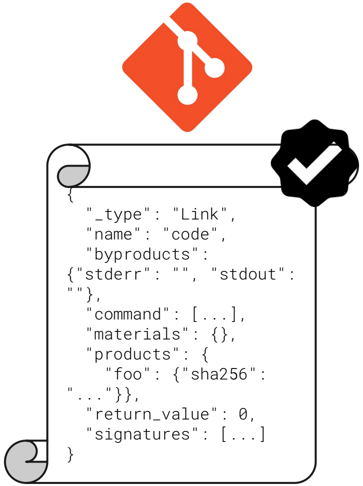
in-toto -- Layout -- Signed by project owner



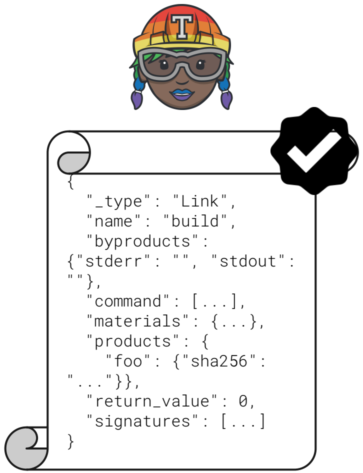
```
{
  "_type": "layout",
  "expires":"2017-08-31T12:44:15Z",
  "keys": {
    "0c6c50": { ... }
  },
  "signatures": [...],
  "steps": [{
    "_type": "step",
    "name": "checkout-code",
    "expected_command": ["git", "clone", "..."],
    "expected_materials": [],
    "expected_products": [
      ["CREATE", "demo-project/foo.py"], ...],
    "pubkeys": ["0c6c50..."],
    "threshold": 1
  }, ...],
  "inspections": [...]
}
```

in-toto -- Links -- Signed evidence for each step

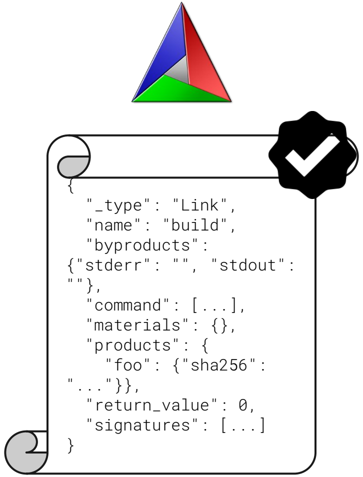
```
$ in-toto-run -- ./do-the-supply-chain-step
```



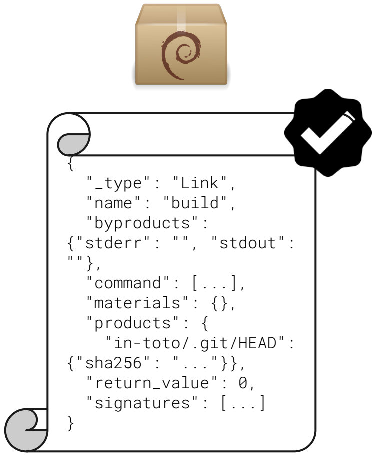
```
{
  "_type": "Link",
  "name": "code",
  "byproducts":
{"stderr": "", "stdout":
""},
  "command": [...],
  "materials": {},
  "products": {
    "foo": {"sha256":
"..."}},
  "return_value": 0,
  "signatures": [...]
}
```



```
{
  "_type": "Link",
  "name": "build",
  "byproducts":
{"stderr": "", "stdout":
""},
  "command": [...],
  "materials": {...},
  "products": {
    "foo": {"sha256":
"..."}},
  "return_value": 0,
  "signatures": [...]
}
```



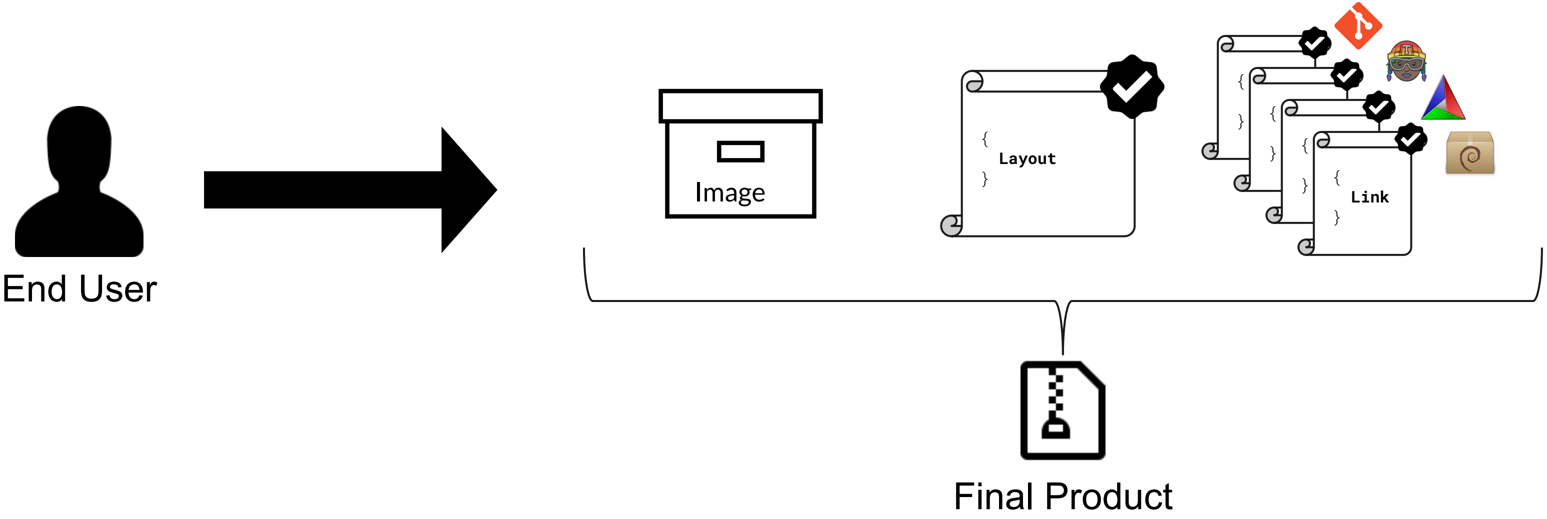
```
{
  "_type": "Link",
  "name": "build",
  "byproducts":
{"stderr": "", "stdout":
""},
  "command": [...],
  "materials": {},
  "products": {
    "foo": {"sha256":
"..."}},
  "return_value": 0,
  "signatures": [...]
}
```



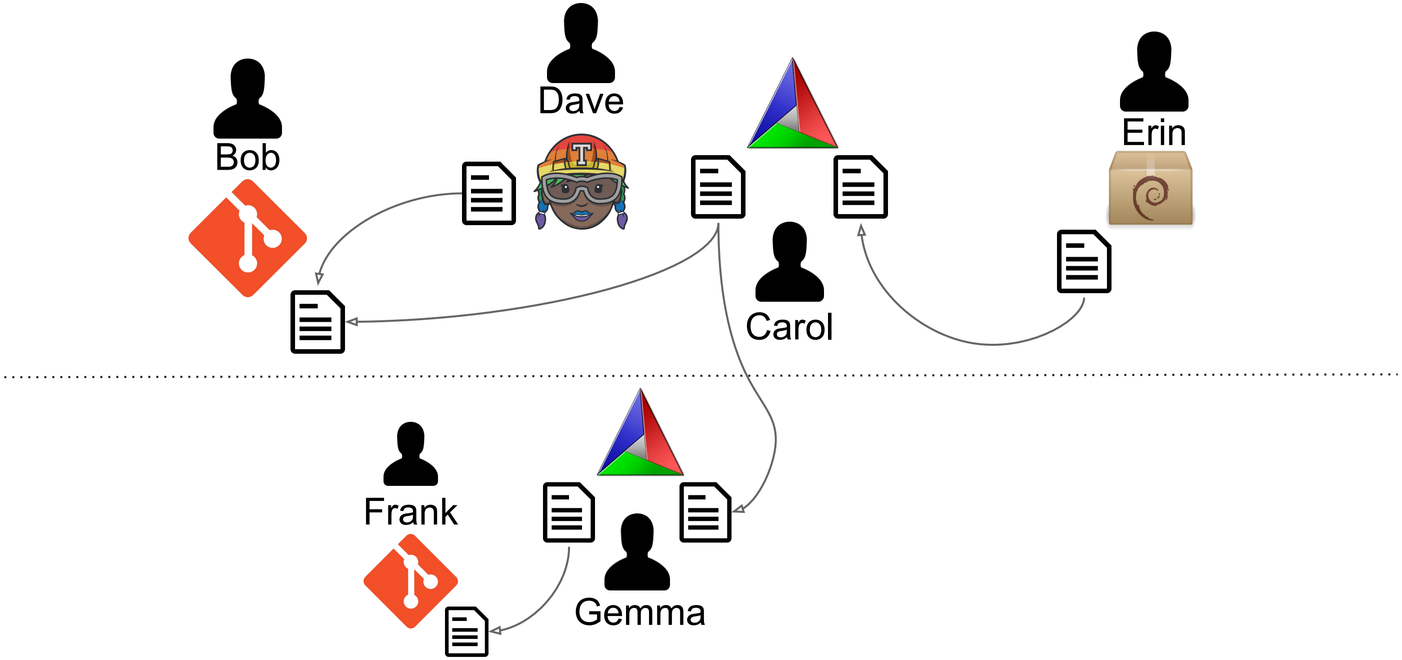
```
{
  "_type": "Link",
  "name": "build",
  "byproducts":
{"stderr": "", "stdout":
""},
  "command": [...],
  "materials": {},
  "products": {
    "in-toto/.git/HEAD":
{"sha256": "..."}},
  "return_value": 0,
  "signatures": [...]
}
```

in-toto Verification

```
$ in-toto-verify --layout <layout> --key <pub key>
```



What about vendor supply chains?



in-toto Integrations and Adoptions

debian
Reproducible Builds
solarwinds
archlinux
DATADOG
controlplane
kubernetes

Scudo = in-toto + Uptane

- Securely distributing metadata for all images before installation
- Identifying ECU responsibilities for in-toto verification on vehicles
- Providing support for vehicles with constrained ECUs
- Supporting vendor supply chains

Uptane Adoptions Outside Automotive

Add Content

Closing Thoughts

Conclusions

- At this time, the Uptane Standard is mature and has been deployed in real-world systems.
- Uptane can be used to guide to develop and deploy secure SOTA.
- Using some Uptane ideas is better than not using them at all.

Conclusions

- Uptane will continue to refine and improve the specification, increasingly focusing on motivation and education
  - Mapping of threats to Uptane modules/requirements to understand what individual Uptane modules/requirements contribute to overall system security (similar to a TARA)
  - Provide strategies to transition from existing SOTA systems to Uptane systems (or improve existing systems with ideas from Uptane)
  - Refine guidance in the Deployment Best Practices
  - Focus on aftermarket devices/systems

Uptane Roadmap Planning

1st quarter 2023	2nd quarter 2023	3rd quarter 2023	4th quarter 2023	1st quarter 2024	2nd quarter 2024
Release V.2.1 of Standard and Deployment Best Practices	Hold in-person community meeting (North America)	Hold virtual workshop (Europe)	Release V.2.2 of Standard/ Deployment	Hold virtual community meeting	Release V.3.0.0 of Standard/ Deployment
Release whitepaper on transitioning to Uptane		Release whitepaper on compliance with regulations and standards		Release whitepaper on aftermarket materials	Hold in-person community meeting (North America)
Hold virtual community meeting					

Please contact us if you are interested to join, contribute and/or learn more:

https://uptane.github.io/participate.html

Thank you.

Appendix

Educational Materials

- Whitepapers, Videos, Tutorials, etc.
- Communicating emerging issues in automotive cybersecurity
- Promoting awareness of cybersecurity issues to the automotive community
- Addressing software supply chain issues
- Topics for upcoming whitepapers: Compliance with regulations and standards, Security issues in the use of aftermarket materials, Transitioning to Uptane

Uptane

Uptane: Securing delivery of softw
updates for ground vehicles

The first in a series of whitepapers on
emerging and critical issues in automotive cyber

Prepared by the Uptane Standards Group

Released June 12, 2021

Uptane：
地上車両を対象とするソフトウェア
アップデート配信のセキュリティ対策

Uptane ホワイトペーパー　《シリーズ 第一弾》

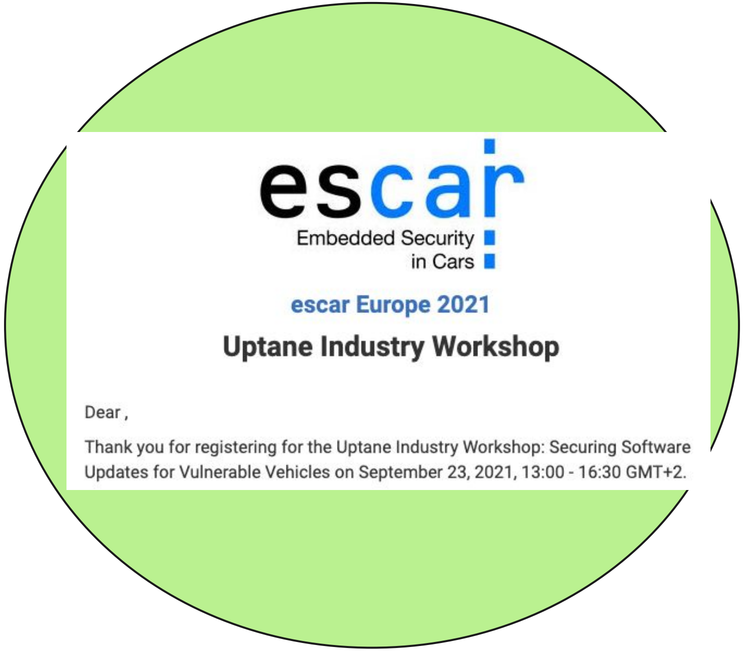
Uptane

Scudo: A Proposal for Resolving
upply Chain Insecurities
in Vehicles

nd in a series of whitepapers on
tical issues in automotive cybersecurity

Industry Workshops

- Offering virtual workshops to reach a global audience at only a fraction of the cost of in-person meetings
- Effective option for a Covid-impacted world
- Two workshops have already been held, one for North America in May 2020 and another for Europe in September 2021
- Soliciting community input on how and when to hold Industry Workshops



escar

Embedded Security in Cars

escar Europe 2021

Uptane Industry Workshop

Dear ,

Thank you for registering for the Uptane Industry Workshop: Securing Software Updates for Vulnerable Vehicles on September 23, 2021, 13:00 - 16:30 GMT+2.

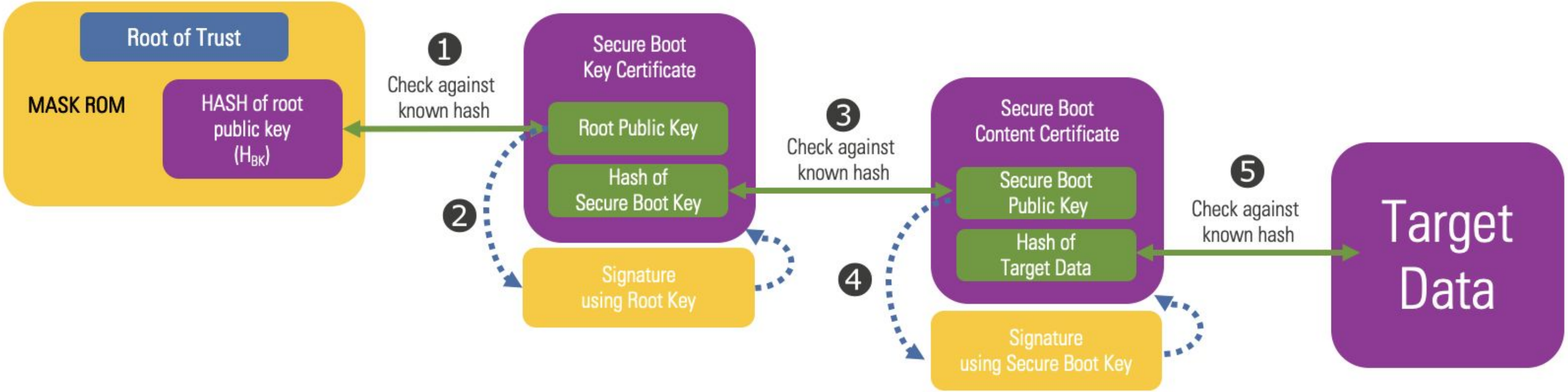
Appendix:
Security Assumptions and Best Practices

Secure Device Provisioning

How do you initially provision software (including Uptane) on to a device?

- Devices need a mechanism to securely program the initial software and root keys
- Usually root keys are fused in device or are set in OTP flash
- How to protect those keys?

Hardware Assisted Secure Boot

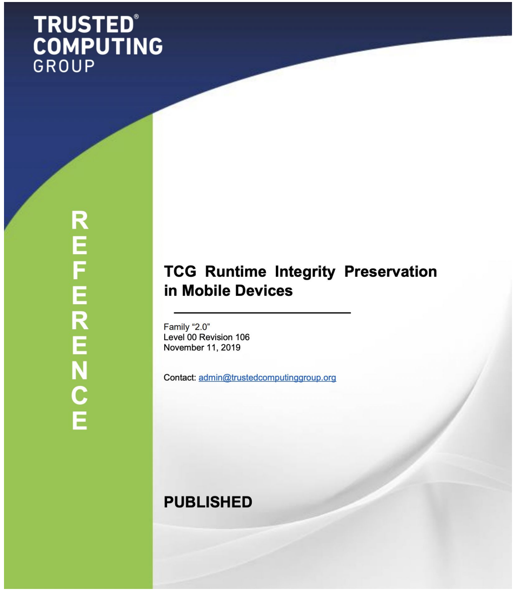


Example Secure Boot Sequence

Hardware Assisted Runtime Integrity

Once secure boot is complete, how to maintain integrity during runtime?

- Runtime integrity tries to answer this question



TRUSTED® COMPUTING GROUP

REFERENCE

TCG Runtime Integrity Preservation in Mobile Devices

Family "2.0"
Level 00 Revision 106
November 11, 2019

Contact: admin@trustedcomputinggroup.org

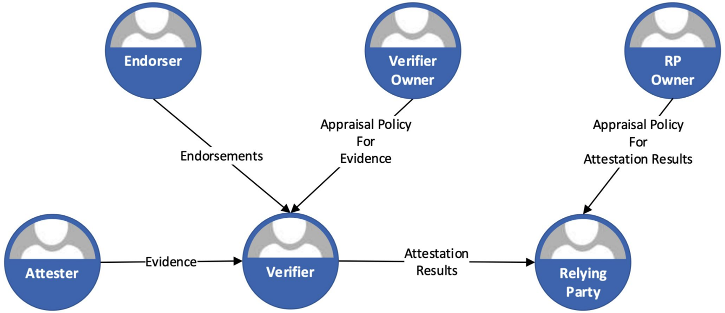
PUBLISHED

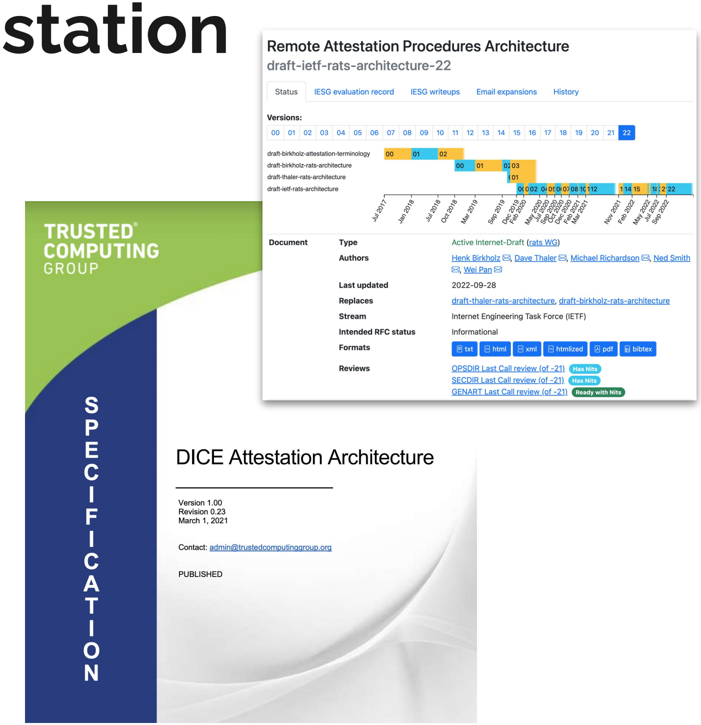
https://trustedcomputinggroup.org/wp-content/uploads/TCG_MPWG_RIP_r106_published.pdf

Hardware Assisted Device Attestation

How to determine when a device can be trusted?

- TCG DICE Attestation Architecture
- IETF Remote Attestation Procedures Architecture (RATS)





Remote Attestation Procedures Architecture

draft-ietf-rats-architecture-22

Document		
	Type	Active Internet-Draft (rats WG)
	Authors	Henk Birkholz ✉, Dave Thaler ✉, Michael Richardson ✉, Ned Smith ✉, Wei Pan ✉
	Last updated	2022-09-28
	Replaces	draft-thaler-rats-architecture, draft-birkholz-rats-architecture
	Stream	Internet Engineering Task Force (IETF)
	Intended RFC status	Informational
	Formats	txt html xml htmlized pdf bibtex
	Reviews	OPSDIR Last Call review (of -21) Has Nits SECDIR Last Call review (of -21) Has Nits GENART Last Call review (of -21) Ready with Nits

TRUSTED® COMPUTING GROUP

SPECIFICATION

DICE Attestation Architecture

Version 1.00
Revision 0.23
March 1, 2021

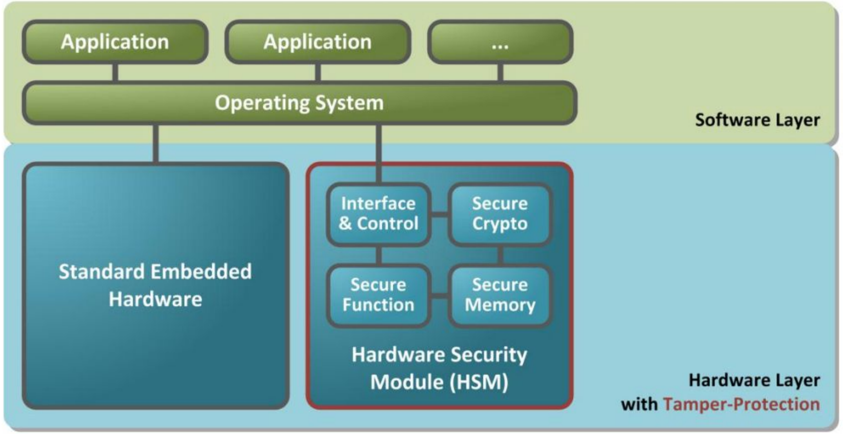
Contact: admin@trustedcomputinggroup.org

PUBLISHED

https://trustedcomputinggroup.org/wp-content/uploads/DICE-Attestation-Architecture-r23-final.pdf

Hardware Protected Security Environments

- Secure segmentation
- Crypto acceleration
- Secure storage of keys
- Secure boot
- SAE J3101



ECU Hardening

- Hardware resistance to fault injection attacks
- Secure coding practices to resist FI attacks
- Tamper protection
- Constant time algorithms
- Security testing of update system

Tool Hardening

Make tools unappealing for attackers

- No secret keys in tool
- No secret algorithms in tool
- Authenticate user roles
- Authenticate communication with ECU
- Authenticate communication with backend
- Use end-to-end encryption of binaries (Tool doesn’t need the unencrypted binary image)
- Take advantage of certificates

Aftermarket - Current Scenario

Aftermarket companies (think of Mopar and AutoCare) are providing services and equipment that are outside of the OEM sphere

- Following end-of-life support from OEMs
- Adding functionality to a vehicle through aftermarket vendors

Owners/car enthusiasts are customizing cars

- Following the right to repair
- Successfully reverse-engineering
- Configuration adjustments

Concerns

Aftermarket concerns

- Responsibility for component operation
- Integration with existing components

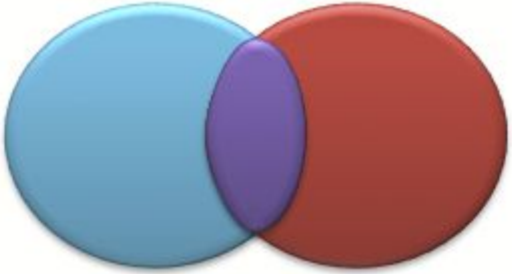
OEM concerns

- Responsibility for safe operation of entire vehicle
  - Including right to repair
  - After end-of-life (minor)
- IP protection

Shared concerns

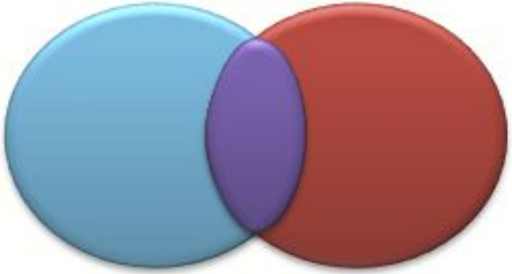
- Secure the vehicle from both electronic and physical intrusion

Alternatives



1. Aftermarket/owner operates independently
   a. OEM and aftermarket/owner operate mutually exclusive ECUs
   b. May not have their own Primary
2. Responsibility (keys, code) is shifted at specific times
   a. End of life
      i. Ownership of update servers would need to be delegated (modify Uptane Standard)
   b. Upon customization of critical safety functions
      i. Perhaps a digital “void warranty” if safety critical firmware is modified
   c. Authorized custom shop is given a key role that allows specific adjustments
3. Aftermarket/customer is integrated
   a. Leverage existing Director/Image servers
      i. Aftermarket may be an optional supplier
   b. Operate their own servers
      i. Authorize additional servers for specific functionality/ECUs

Alternatives



1. Aftermarket/owner operates independently
   a. OEM and aftermarket/owner operate mutually exclusive ECUs
   b. May not have their own Primary
2. Responsibility (keys, code) is shifted at specific times
   a. End of life
      i. Ownership of update servers would need to be delegated (modify Uptane Standard)
   b. Upon customization of critical safety functions
      i. Perhaps a digital “void warranty” if safety critical firmware is modified
   c. Authorized custom shop is given a key role that allows specific adjustments
3. Aftermarket/customer is integrated
   a. Leverage existing Director/Image servers
      i. Aftermarket may be an optional supplier
   b. Operate their own servers
      i. Authorize additional servers for specific functionality/ECUs

Next steps

1. Identify/verify concerns of different stakeholders
2. Rank/identify new alternatives
3. Recommend modifications to Uptane standard

Software Supply Chain Security

SBOMs have emerged as key building blocks in software supply chain security

It is a nested inventory, a list of ingredients that make up software artifacts

Regulations like Executive Order 14028 on improving the nation's cybersecurity call for them

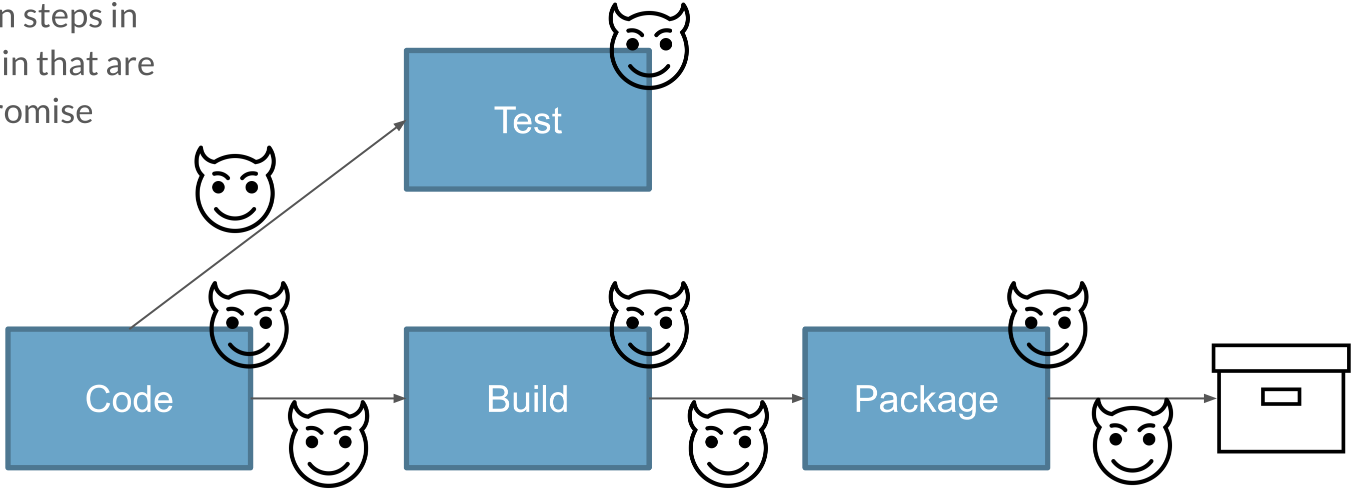
References:

CISA SBOM-A-RAMA

NTIA - The Minimum Elements For a Software Bill of Materials (SBOM)

Securing the Software Supply Chain

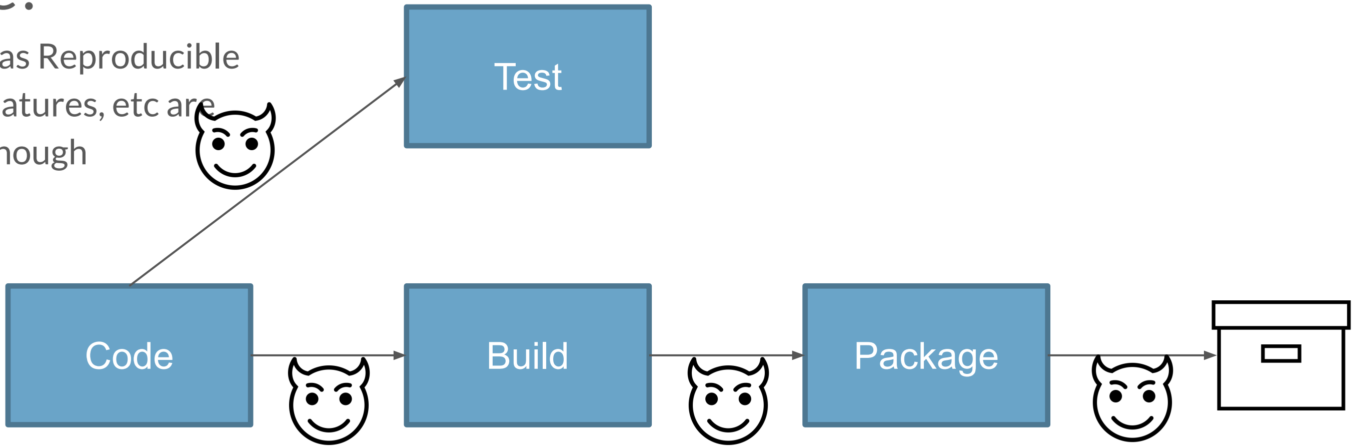
Aim to verifiably sign steps in software supply chain that are vulnerable to compromise



Gaps Between Steps in Supply Chain?

Compliance?

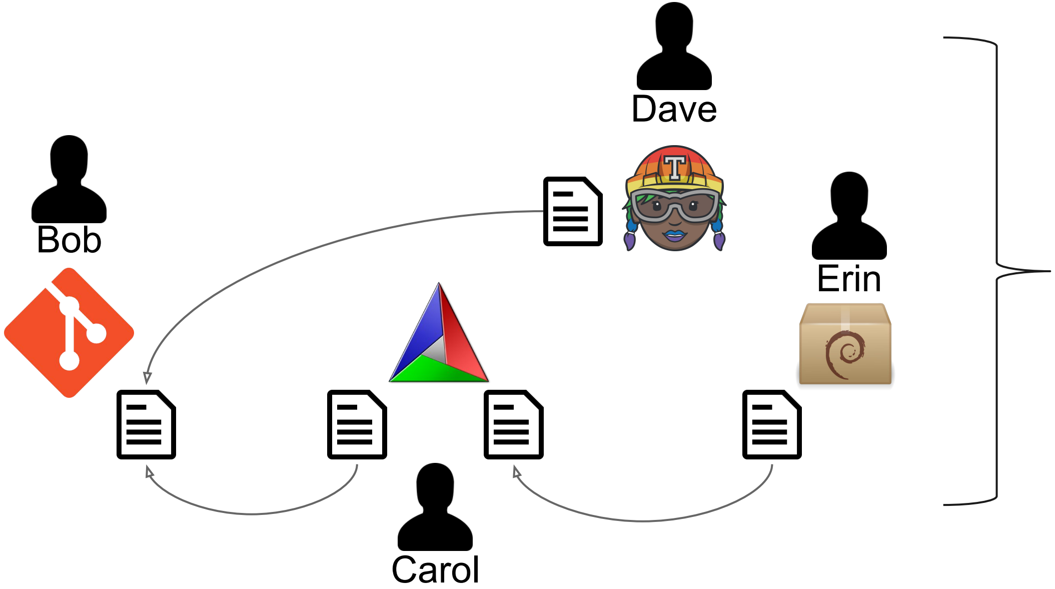
Spot solutions such as Reproducible Builds, Commit Signatures, etc are necessary but not enough

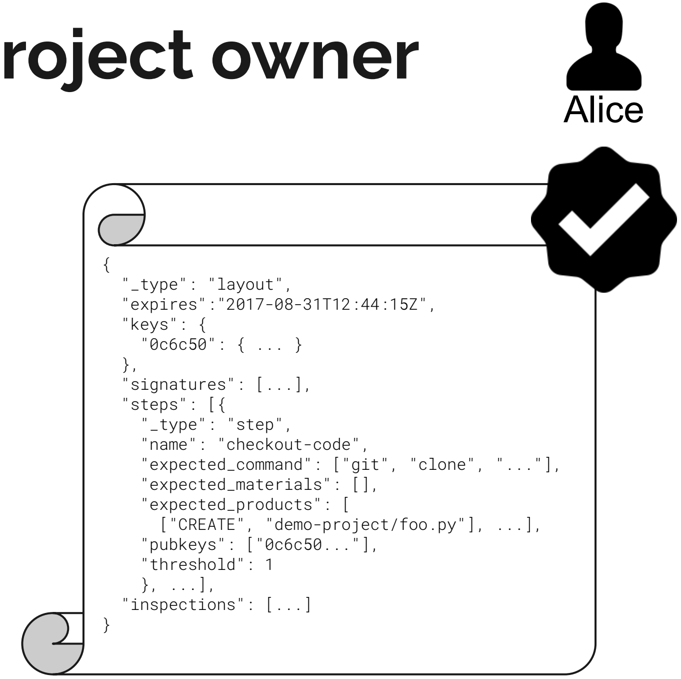


in-toto

- Verifiably define the steps of the software supply chain
- Verifiably define the authorized actors
- Guarantee everything happens according to definition and nothing else

in-toto -- Layout -- Signed by project owner

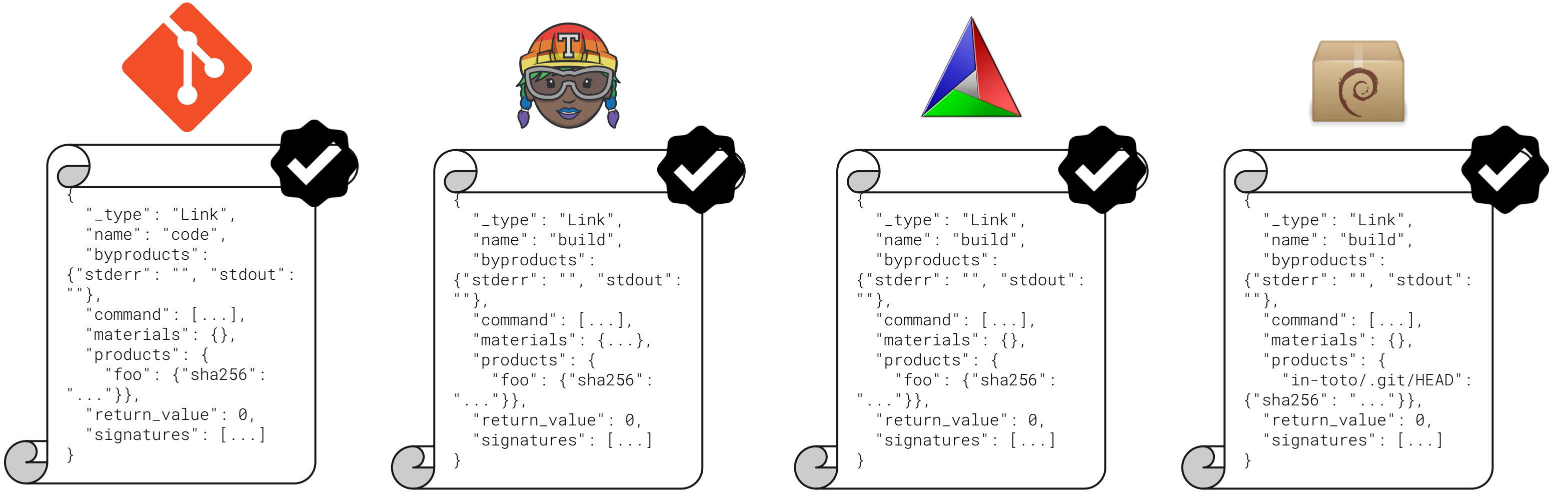




```
{
  "_type": "layout",
  "expires":"2017-08-31T12:44:15Z",
  "keys": {
    "0c6c50": { ... }
  },
  "signatures": [...],
  "steps": [{
    "_type": "step",
    "name": "checkout-code",
    "expected_command": ["git", "clone", "..."],
    "expected_materials": [],
    "expected_products": [
      ["CREATE", "demo-project/foo.py"], ...],
    "pubkeys": ["0c6c50..."],
    "threshold": 1
  }, ...],
  "inspections": [...]
}
```

in-toto -- Links -- Signed evidence for each step

```
$ in-toto-run -- ./do-the-supply-chain-step
```



```
{
  "_type": "Link",
  "name": "code",
  "byproducts":
{"stderr": "", "stdout":
""},
  "command": [...],
  "materials": {},
  "products": {
    "foo": {"sha256":
"..."}},
  "return_value": 0,
  "signatures": [...]
}
```

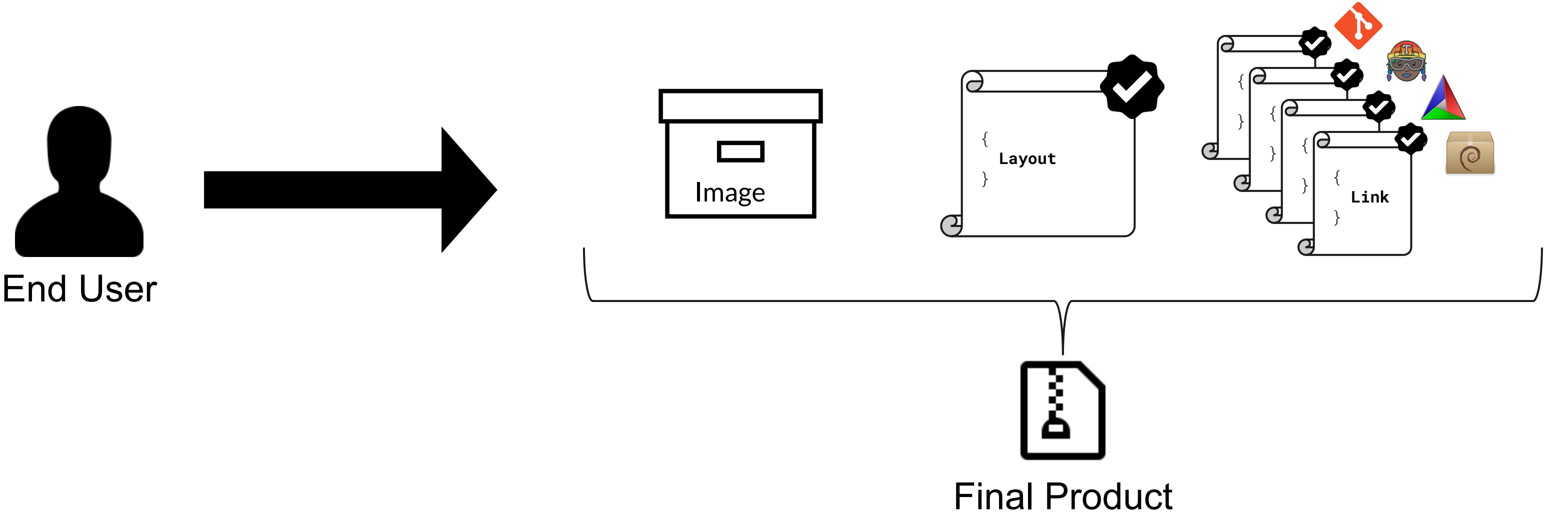
```
{
  "_type": "Link",
  "name": "build",
  "byproducts":
{"stderr": "", "stdout":
""},
  "command": [...],
  "materials": {...},
  "products": {
    "foo": {"sha256":
"..."}},
  "return_value": 0,
  "signatures": [...]
}
```

```
{
  "_type": "Link",
  "name": "build",
  "byproducts":
{"stderr": "", "stdout":
""},
  "command": [...],
  "materials": {},
  "products": {
    "foo": {"sha256":
"..."}},
  "return_value": 0,
  "signatures": [...]
}
```

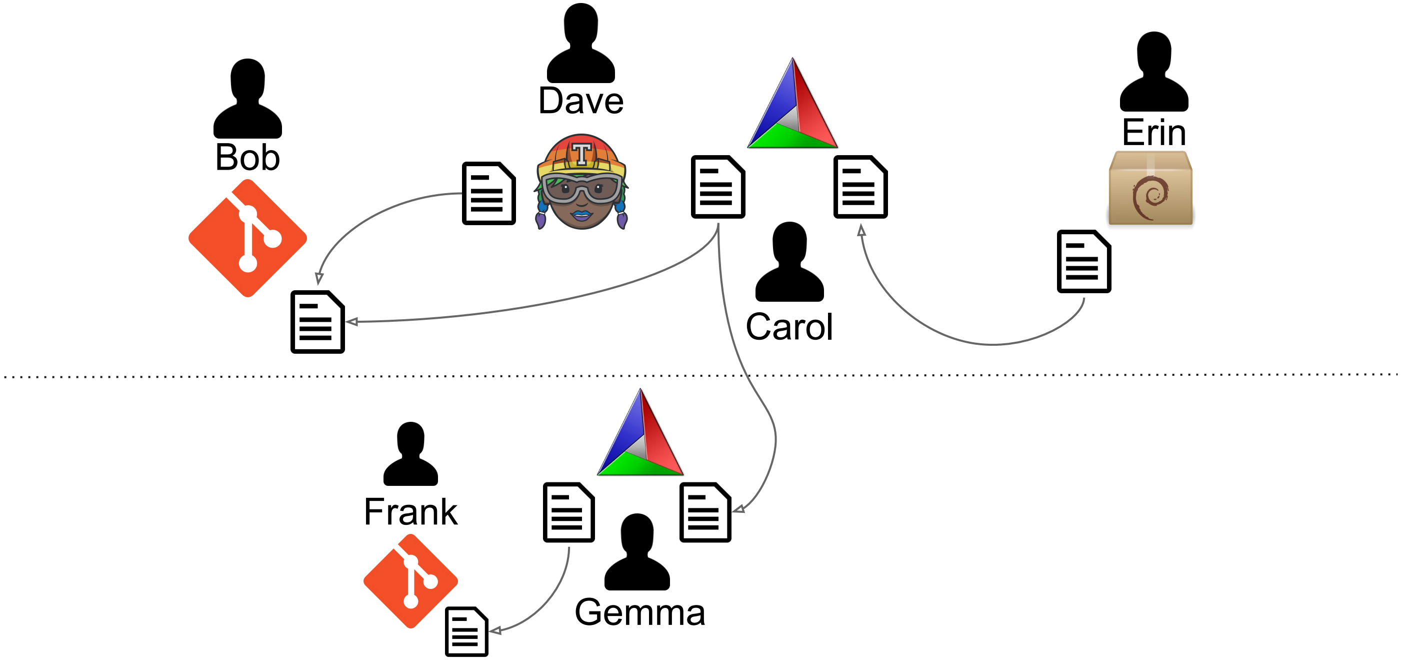
```
{
  "_type": "Link",
  "name": "build",
  "byproducts":
{"stderr": "", "stdout":
""},
  "command": [...],
  "materials": {},
  "products": {
    "in-toto/.git/HEAD":
{"sha256": "..."}},
  "return_value": 0,
  "signatures": [...]
}
```

in-toto Verification

```
$ in-toto-verify --layout <layout> --key <pub key>
```



What about vendor supply chains?



in-toto Integrations and Adoptions

debian
Reproducible Builds
solarwinds
archlinux
DATADOG
controlplane
kubernetes

Software Bill of Materials (SBOM)

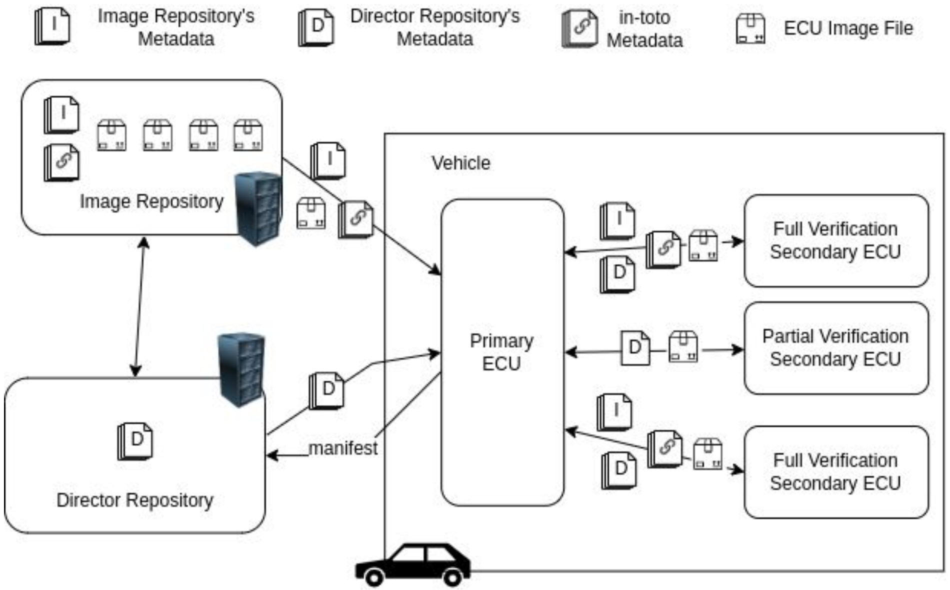
SBOM is a nested inventory of software components

SBOMs are a key building block in software supply chain security

Auto-ISAC is creating a best practice document about automotive SBOM

References:
CISA SBOM-A-RAMA
NTIA - The Minimum Elements For a Software Bill of Materials (SBOM)

Scudo = in-toto + Uptane


I
Image Repository's Metadata
D
Director Repository's Metadata
in-toto Metadata
ECU Image File
I
Image Repository
D
Director Repository
manifest
Vehicle
Primary ECU
Full Verification Secondary ECU
Partial Verification Secondary ECU
Full Verification Secondary ECU

Scudo = in-toto + Uptane

Successful integrations of in-toto and TUF in use in production:
https://www.datadoghq.com/blog/engineering/secure-publication-of-datadog-agent-integrations-with-tuf-and-in-toto/

Integrated in-toto with Uptane considers the nuances of the auto industry:
https://uptane.github.io/papers/scudo-whitepaper.pdf

More advanced specification of Scudo available as an upcoming Uptane PURE:
https://github.com/uptane/pures/pull/9