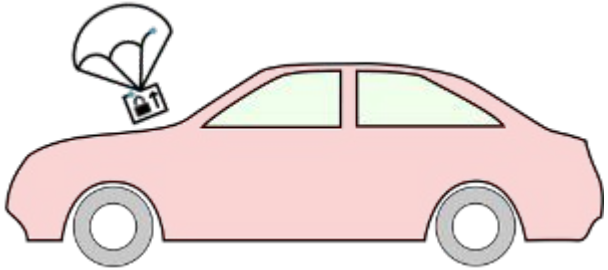
Uptane: Lessons Learned

Securing Over-the-Air Updates



Patti Vacek
unu

André Weimerskirch
Lear Corporation

uptane.github.io

Outline

- Background
  - How did we get here? What have we learned along the way?
- Secondary verification -- full vs partial
  - How does Uptane handle the variety of Secondaries in the market?
- Evolving from legacy systems
  - How should a new implementer use the Standard? Where should they start?
- Open topics and audience questions

Background

- Let’s assume you know the basics…
  - But we can offer more detail if anything isn’t clear
- Uptane has evolved since it’s inception in 2016
  - And it’s still evolving -- there are plenty of open topics!
  - There is a changelog for versions 1.1.0 and 1.2.0; see also the timeline
- Some of the earliest feedback we got was around Secondary verification
- There are still plenty of questions about how to adopt Uptane in existing systems

Full verification of Secondaries

- In a perfect Uptane model, all Secondaries perform full verification.
- This means after the Primary verifies all metadata and sends the relevant parts to the Secondary, the Secondary also verifies all of the metadata (Root, Timestamp, Snapshot, and Targets metadata from both the Director and Image repositories).
- This of course requires some memory and computational effort, particularly for cryptographic functions to verify signatures, that might not be readily available on every Secondary.
- Implementers told us early on this might not be practical in many cases.

Partial verification of Secondaries

- For resource-constrained Secondaries, we introduced the concept of partial verification.
- The Primary still verifies all metadata.
- However, the Secondary only needs to receive and verify the Targets metadata from the Director repository. This requires storing only one file and performing one cryptographic signature check.
- Similar to what many systems are already doing to check firmware validity.

Extension of partial verification

- Of course, after partial verification was introduced, some implementers asked if they could do more than just the bare minimum, such as also receiving and verifying the Root metadata from the Director repository.
- The Standard was adapted to be clear that there is no problem with Secondary verification that is somewhere between the bare minimum and full verification.

Legacy Secondaries

- What about Secondaries with no capacity for Uptane verification?
- As a first step, the Primary can still verify metadata for these Secondaries.
- Later, these Secondaries can be upgraded to at least perform partial verification.

Examples

- Full verification: aktualizr-secondary
- Partial verification: uptiny
- libaktualizr is extensible and can also support legacy Secondaries and anything between the bare minimum of partial verification up to full verification.
- See also the Uptane reference implementation.
- Automotive Grade Linux also includes aktualizr.

Evolving from Legacy Systems

- **Today's reality**: Most stakeholders already have a SOTA strategy and system in place, together with a security protocol. But with a legacy system in place, it is hard to upgrade to Uptane conformant SOTA security.
- We performed interviews to understand the barriers and received recommendations how to make it easier to evolve legacy systems to Uptane.
- We then used this feedback to determine what a stakeholder needed to achieve Uptane conformance.
- **Value of this discussion**: This discussion will help stakeholders understand the principles of Uptane, define a roadmap for migration, and collect feedback for the Uptane team.

What is Uptane (and what is it not)?

- Uptane is a standard that defines a high-level architecture and security requirements but not interoperability
  - No byte-level interoperability standard like, say, TLS but a set of requirements like, say, a building code.
  - No prescriptive standard, e.g., Uptane does not require the use of a particular signature algorithm.
- Byte-level interoperability is introduced by POUFs (protocol, operations, usage, formats)
- Uptane does not interfere with any particular SOTA technology (e.g., delta updates or the use of a standard on-board diagnostic tester simulated by a central module in the vehicle).

Stakeholder Feedback

- There is no reason to understand Uptane since a secure SOTA mechanism is already in place
- It is unclear how to integrate Uptane into an existing legacy system, and there is no support offered.
- Uptane’s requirements are too strict (all-or-nothing approach)
- Not enough security suppliers buy-into Uptane and offer Uptane conformant solutions
- On the plus side: SOTA solutions are generally not set in stone and can be regularly updated.

Stakeholder Feedback - Our Thoughts

- There is no reason to understand Uptane since a secure SOTA mechanism is already in place.
  - Maybe - If you are concerned about reaching the state-of-the-art security level, you should perform a risk assessment of your legacy system with the Uptane threats (Section 4.3 in v1.2.0) in mind.

4.3. Description of threats

Uptane's threat model includes the following types of attacks, organized according to the attacker goals listed in Section 4.1.

4.3.1. Read updates

- *Eavesdrop attack:* Read sensitive or confidential information from an update intended to be encrypted for a specific ECU. (Note: Not all implementations will have a need to protect information in this way.)

4.3.2. Deny installation of updates

An attacker seeking to deny the installation of updates may attempt one or more of the following strategies:

- *Drop-request attack:* Block network traffic outside or inside the vehicle.
- *Slow retrieval attack:* Slow down network traffic, in the extreme case sending barely enough packets to avoid a timeout. Similar to a drop-request attack, except that both the sender and receiver of the traffic still think network traffic is unimpeded.
- *Freeze attack:* Continue to send a properly signed, but old, update bundle to the ECUs, even if newer updates exist.
- *Partial bundle installation attack:* Install a valid (signed) update bundle, and then block selected updates within the bundle.
- Conduct a denial of service attack against the Uptane repositories or infrastructure.

4.3.3. Interfere with ECU functionality

Attackers seeking to interfere with the functionality of vehicle ECUs in order to cause an operational failure or unexpected behavior may do so in one of the following ways:

- *Rollback attack:* Cause an ECU to install a previously valid software revision that is older than the currently installed version.
- *Endless data attack:* Send a large amount of data to an ECU until it runs out of storage, possibly causing the ECU to fail to operate.
- *Mix-and-match attack:* Install a malicious software bundle in which some of the updates do not interoperate properly. This may be accomplished even if all of the individual images being installed are valid, as long as valid versions exist that are mutually incompatible.

4.3.4. Control an ECU or vehicle

Full control of a vehicle, or one or more ECUs within a vehicle, is the most severe threat.

- *Arbitrary software attack:* Cause an ECU to install and run arbitrary code of the attacker's choice.

Stakeholder Feedback - Our Thoughts

- It is unclear how to integrate Uptane into an existing legacy system, and there is no support offered.
  - True - Uptane requires a significant time investment to understand in order to apply it to a legacy system.
  - It is possible though. A stakeholder can do either of the following:
    i. Go over all Uptane requirements and check which requirements the legacy system does not satisfy. Then define a roadmap to fill all identified gaps.
    ii. Perform a risk assessment of the legacy system with the Uptane threats in mind. Then identify the Uptane requirements that will mitigate the identified risks, and define a roadmap around those identified Uptane requirements.
  - This will be rather cumbersome since there are no supporting tools/templates in place and there is no comprehensive mapping available from threats/attacks to Uptane requirements.

Stakeholder Feedback - Our Thoughts

- Uptane's requirements are too strict (all-or-nothing approach)
  - Maybe - Allowing for tailoring (justification why a requirement is covered by adequate means) should be considered.
  - This enables well thought-through legacy systems to claim Uptane conformance.
- Not enough security suppliers buy-into Uptane and offer Uptane solutions
  - SOTA is rather critical, and even a weakness (vs. vulnerability) might not be acceptable. Ask your supplier to provide a risk assessment that takes into account the Uptane security threats.
- On the good side: SOTA solutions are not put in stone and can be regularly updated.
  - Hooray - This might allow at least some stakeholders to define a roadmap and become Uptane conformant over time.

What do we need?

- Checklist and guidance to evaluate an existing legacy solution against Uptane conformance
  - A checklist that lists all Uptane requirements and maps those to the threat they counter.
  - Ideally as an ISO 21434 conformant TARA template that includes Uptane threats
- Guidance how to migrate to Uptane step-by-step, ideally displaying the transition from common SOTA architectures to Uptane
  - Requires your input for a common SOTA architecture
- Introduce tailoring, i.e., allow justification why a requirement is covered adequately.

Open Challenges

- Consider future architectures
  - For instance, how do we apply Uptane to a vehicle with two Primary ECUs?
- Can Uptane help with heterogeneous vehicle architectures?
  - For instance, consider self-driving technology that overlays the standard vehicle E/E architecture and origins from at least two sources: How do we update both in-sync?
- Can Uptane help with aftermarket devices?
  - Imagine aftermarket devices that have interdependencies with the software running in the vehicle.

Thank you!

Audience Questions

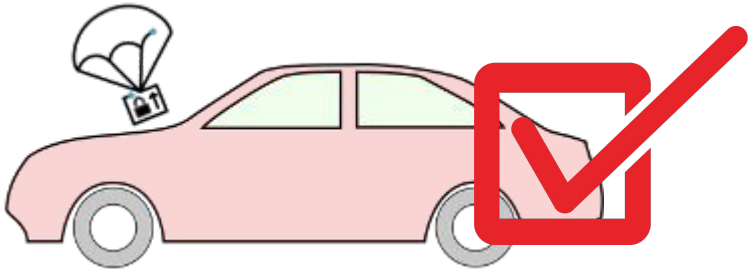
Additional topics

- Atomic updates and installation priority
- Bandwidth efficiency
- Transport-layer security
- Metadata initialization at manufacturing time
- Location-based updates
- ...

Get Involved With Uptane!

- Workshops
- Technology demonstration
- Compliance tests
- Standardization (IEEE / ISTO)
- Join our community! (email: jcappos@nyu.edu or go to the Uptane forum)

https://uptane.github.io/



Thank you!

André Weimerskirch
aweimerskirch@lear.com

Patti Vacek
patti@unumotors.com

uptane.github.io