Release Integrity Plan

Revision

Version 3
11/15/21 9:10 AM

SME

Charles Wilson

Abstract

This document describes the methodology to ensure that the software to be deployed has sufficient controls applied to it to ensure its integrity.

Group / Owner

devops / Information Systems Security Developer

Motivation

This document is motivated by the need to have formal processes in place to control the software deployed into safety-critical, cyber-physical systems for certification of compliance to standards such as ISO 21434 and 26262.

License

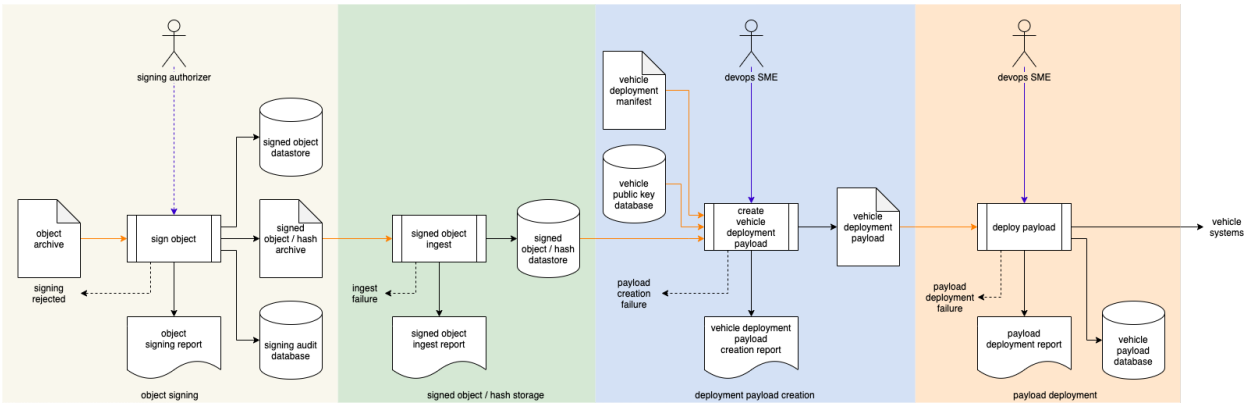
Overview

Note: This document has overlap with those relating to toolchain management.

There are several controls which need to be in place in order to secure the software to be deployed. These include:

- Code signing
- Hash tracking
- Credential management
- Root-of-trust implementation
- Secure deployment support
- Unauthorized alteration countermeasures

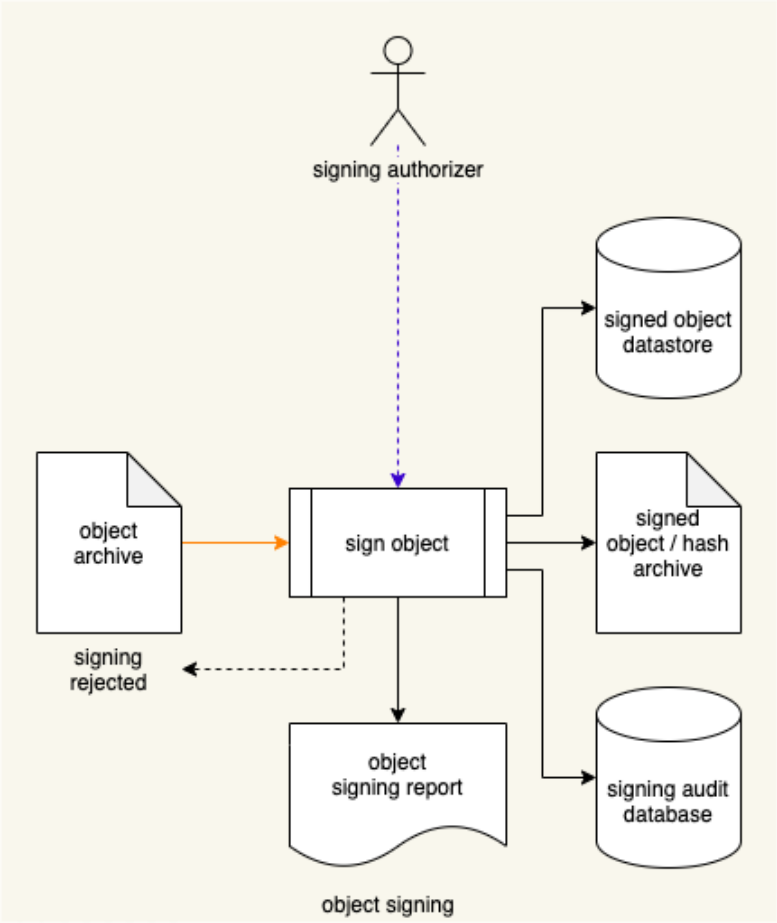
Below is the overview of the process used to create and deploy the deployment payload securely.



Process

Object Signing

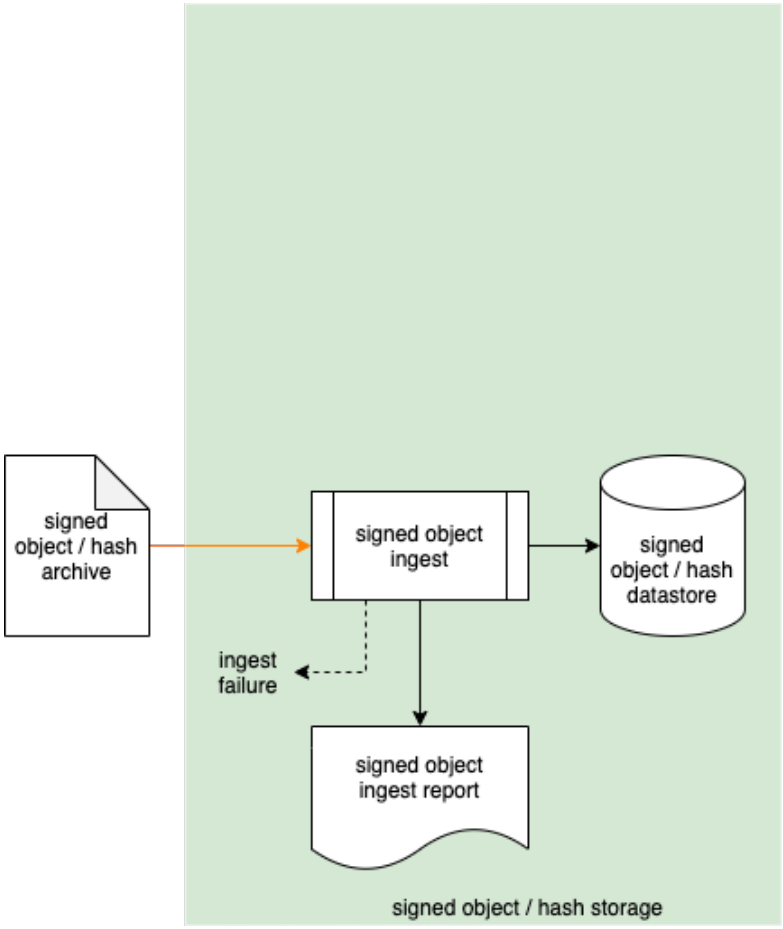
Inputs	Object archive
Outputs	signed object datastore signed object / hash archive signing audit database object signing report
Participants	Signing Authorizer (optional)



The code signing infrastructure takes an object archive and sign it. This may require the authorization of the Signing Authorizer. Invocation of the code signing infrastructure will always generate an object signing report and add a log entry to the signing audit database. Should signing be unsuccessful, a signing rejected notification is generated. Should signing be successful, a signed object / hash archive will be produced, and a copy stored in the signed object datastore.

Signed Object / Hash Storage

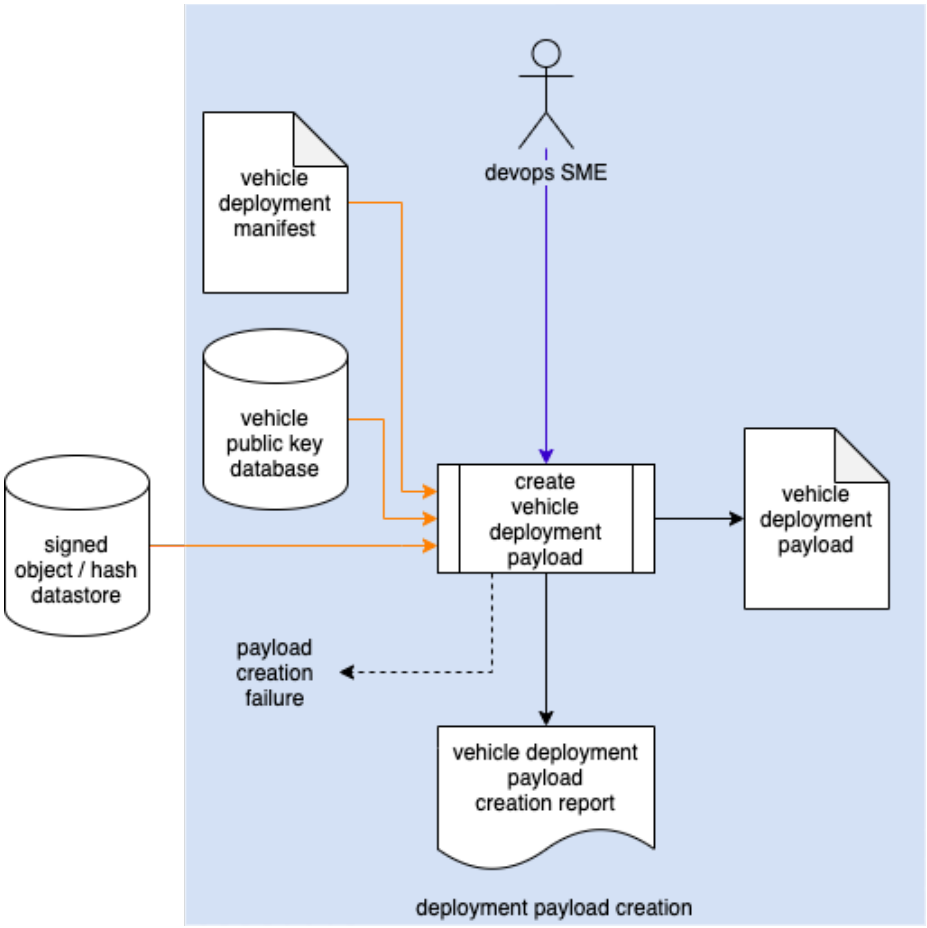
Inputs	signed object / hash archive
Outputs	signed object / hash database signed object ingest report
Participants	**none**



The signed object / hash archive is taken as input to the signed object ingest process. Invocation of the signed object ingest will generate a signed object ingest report. Should ingest be unsuccessful, an ingest failure notification is generated. Should ingest be successful the object / hash archive will be added to the signed object / hash datastore.

Deployment Payload Creation

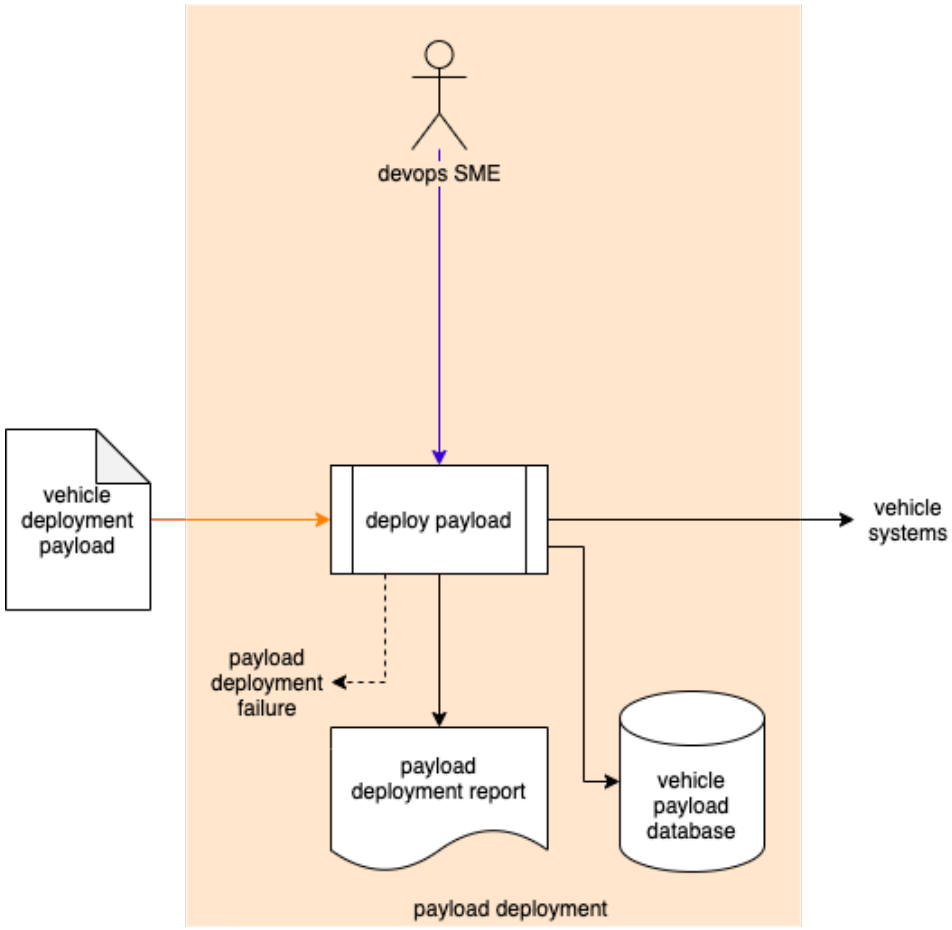
Inputs	signed object / hash database vehicle deployment manifest vehicle public key database
Outputs	vehicle deployment payload vehicle deployment payload creation report
Participants	Devops SME



The Security SME, together with Development SME(s), review the candidate development tool to determine whether it has acceptable security controls in place. The specifics of “acceptable” is based on the tool and its application. If the tool is deemed unacceptable, it is rejected and may not be used. A report documenting the nature and outcome of the review will be generated.

Payload Deployment

Inputs	Vehicle deployment payload
Outputs	vehicle deployment payload vehicle payload database payload deployment report
Participants	Devops SME



The Security SME, together with Development SME(s), review the candidate development tool to determine whether it has acceptable security controls in place. The specifics of “acceptable” is based on the tool and its application. If the tool is deemed unacceptable, it is rejected and may not be used. A report documenting the nature and outcome of the review will be generated.

References

1. **Deployment Plan** (AVCDL secondary document)