Secure Software Distribution in Aviation Context

Dr. David von Oheimb and Dr. Rainer Falk
Siemens Corporate Technology, IT Security

http://www.sae.org/events/atc/
Overview

- Software Distribution Systems
- Technical Challenges
- Security Mechanisms
- Conclusion
Motivation for software distribution

• In aircraft and other mobile systems, more and more software is used.
  • BMW estimation of 2007: 25+% of the value of a modern car is software
  • “An airplane is a supercomputer with wings”
• Specific software and increasingly standard components are employed.
  • For example, proprietary device controller and generic network stacks
• Software needs to be updated more and more often:
  • Bug fixes, both self-induced and inherited ones
  • Enhancements/Updates due to evolving requirements
• Software controls critical parts of mobile systems w.r.t. safety and business.
  • For instance: velocity control, emergency modes, maintenance utilities
• Consequently, software distribution in the field is essential and critical.
**Electronic Distribution of Software (EDS) for aircraft**

**Transition** from media-based (CD-ROMs etc.) to networked SW transport

*EDS* is a system for **storage and distribution of airplane software assets**, including *Loadable Software Airplane Parts (LSAP)* and airplane health data.

![Diagram of EDS process](Diagram.png)
EDS system architecture

A complex distributed store-and-forward middleware (incl. web services) with heterogeneous components (incl. off-the-shelf and open-source SW)

Figure is heavily simplified and not up-to-date!
Overview

- Software Distribution Systems
- Technical Challenges
- Security Mechanisms
- Conclusion
Challenges for software update process in the field

- **Quality and compatibility** of software update items
  - New SW must work correctly and fit in the context
- **Patch management** including version control
  - When to apply which update on which devices
- **Efficiency** of software transport (bandwidth) and installation (down time)
- **Total costs** of the upgrade process including all parties
  - High degree of automation and use of existing infrastructure desirable
- **Safety hazards** due to tampering/sabotage, business risks due to disruption of service, denial of liability, or product counterfeiting
  - Main problem: transport over open, untrusted networks
- **Regulators or customers may require safety/security certification**
  - Software update must be dealt with as part of the overall system/process
Safety-related security threats for EDS in aviation

**Attacker’s objective:** lower airplane safety margins by tampering software that will be executed onboard an aircraft

<table>
<thead>
<tr>
<th>Correlation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Corruption/Injection</td>
<td>Diversion</td>
</tr>
</tbody>
</table>

**Diagram:**
- **Stored Assets**
  - Part #1
  - Part #m
- **Distributed Assets**
  - Part #h
  - Part #m
- **Remote Access**
- **External Adversary**
- **Local Access**
- **Internal Adversary**

**Keywords:**
- Diverted SW
- Corrupted SW
- Outdated SW
- Disclosed SW
- Missing SW
- Wrong SW
- Part #1
- Part #m
- Part #h
- Part #n
Overview

- Software Distribution Systems
- Technical Challenges
- Security Mechanisms
- Conclusion
Generic Software Distribution System (SDS)

Consider any IT system with networked devices in the field performing safety-critical and/or security-critical tasks and requiring the option to update software components.

Software Distribution System (SDS):
System providing secure distribution of software from software suppliers ultimately to target devices in the field.

Possibly many suppliers, several operators, indirect trust relations.
Generic core of SDS: Software Signer Verifier (SSV)

Each node in the SDS chain runs an SSV instance, used to:

- **introduce** software into the SDS, adding signature + optional encryption
- **verify** the signature on software received from other SSVs, checking software integrity, plus sender authenticity and authorization
- **approve** software, adding e.g. an authorized signature or tag, after e.g. checking security policy, static code analysis, runtime check injection, …
- **deliver** software to the installation target, optional decryption
SDS/SSV main security mechanism: cryptography

Main security mechanism: asymmetric cryptography applied on SW item

- Sender S encrypts software with public key of receiver R: \( K(R) \)
  → only the intended receiver R can read it (using R’s private key \( K(R)^{-1} \))

- Sender S signs software and meta-data with private key of itself: \( K(S)^{-1} \)
  → any receiver can verify (using S’s public key \( K(S) \)) if it comes from S

\[ S \text{ sends to } R : \{ SW \}^{}_{K(R)} \cdot \{ \text{hash}(SW) \cdot R \cdot \ldots \}^{}_{K(S)^{-1}} \]

- \( SW \) a software item including its identity etc.
- \( \text{hash}(M) \) the hash value (i.e. crypto checksum) of content \( M \)
- \( M \cdot N \) the concatenated contents \( M \) and \( N \)
- \( \{ M \}^{}_{K(R)} \) content \( M \) encrypted with public key of \( R \)
- \( \{ M \}^{}_{K(S)^{-1}} \) content \( M \) digitally signed with private key of \( S \)

**Alternative**: secure transport channels, e.g. TLS between trusted parties
SDS/SSV: issues and solutions wrt. cryptography

- Major issue: key management
  - Field devices must have authentic public key of SW distributor
  - For encryption, distributor must know all public keys of devices
  - More complex in multi-stage distribution (including indirect trust)
  - Most involved: revocation and update of keys stored in the field
  - Lightweight custom solutions or existing Public Key Infrastructure (PKI)
- Long-lived (10+ years) assurance required
  - Use strong algorithms with long keys and perform scheduled re-keying
- Secure key storage and prevent side-channel attacks on private keys
  - For high assurance: hardware modules and special algorithms
- Efficiency and bandwidth/storage space minimization
  - Hybrid encryption with asymmetrically encrypted symmetric transport key
SDS: issues and solutions w.r.t. target devices

• In some cases, code quality/security can not be guaranteed
  • Isolate malicious behavior with sandboxing/virtualization
• SW maybe not installable at all times or manual processing required
  • Ensure that target is in suitable state (e.g. authorized maintenance mode)
• New version may have specific configuration requirements
  • Check if other installed SW items have conflicting versions or status
• Data/configurations of previous SW versions may be incompatible
  • Delete conflicting parts or apply transformation during installation
• SW installation might fail after partial update. Fallback strategies:
  • Keep previous configuration until correct update has been verified
  • Retry transmission/installation of new SW
  • Provide alternative source (of new version or emergency substitute)
Overview

- Software Distribution Systems
- Technical Challenges
- Security Mechanisms
- Conclusion
Conclusion

- Software distribution in aviation context needs to cover complex global dynamic heterogeneous architectures
- Main aspects: configuration management and safety/security
- Major challenge is management of cryptographic keys
- Additional issues: configuration & installation at target devices
- High assurance requires special solutions with HW support
- Maximal confidence can be obtained by security certification