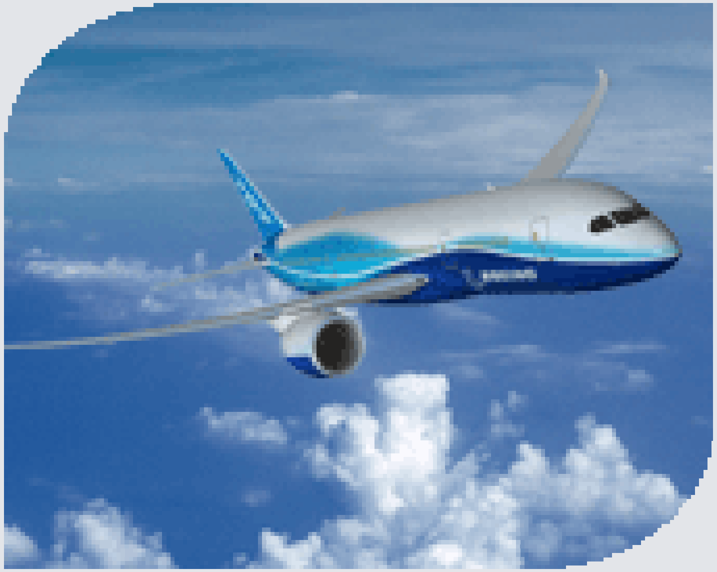
Formal Security Analysis of Software Distribution Systems

Monika Maidl[1], David von Oheimb[1],
Peter Hartmann[2], and Richard Robinson[3]

[1] Siemens Corporate Technology, Munich
[2] University of Applied Sciences, Landshut
[3] Boeing Phantom Works, Seattle, USA

FoMSESS Workshop 2008,
Darmstadt, Germany, 27 March 2008



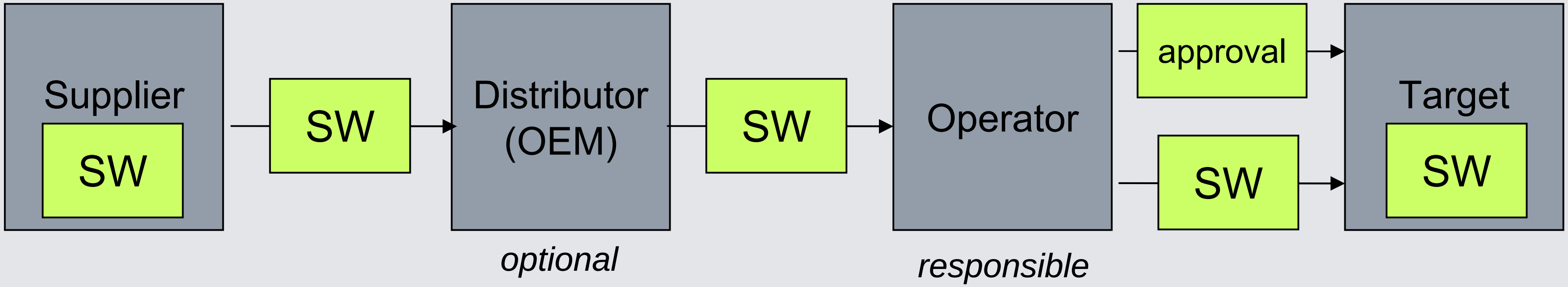
Overview

- **S**oftware **D**istribution **S**ystems
- Hybrid security assessment
- Alice-Bob protocol model
- Validation with AVISPA Tool
- Conclusion

Software Distribution System (SDS)

ICT systems with networked devices in the field performing safety-critical and/or security-critical tasks.
Field devices require secure software update.

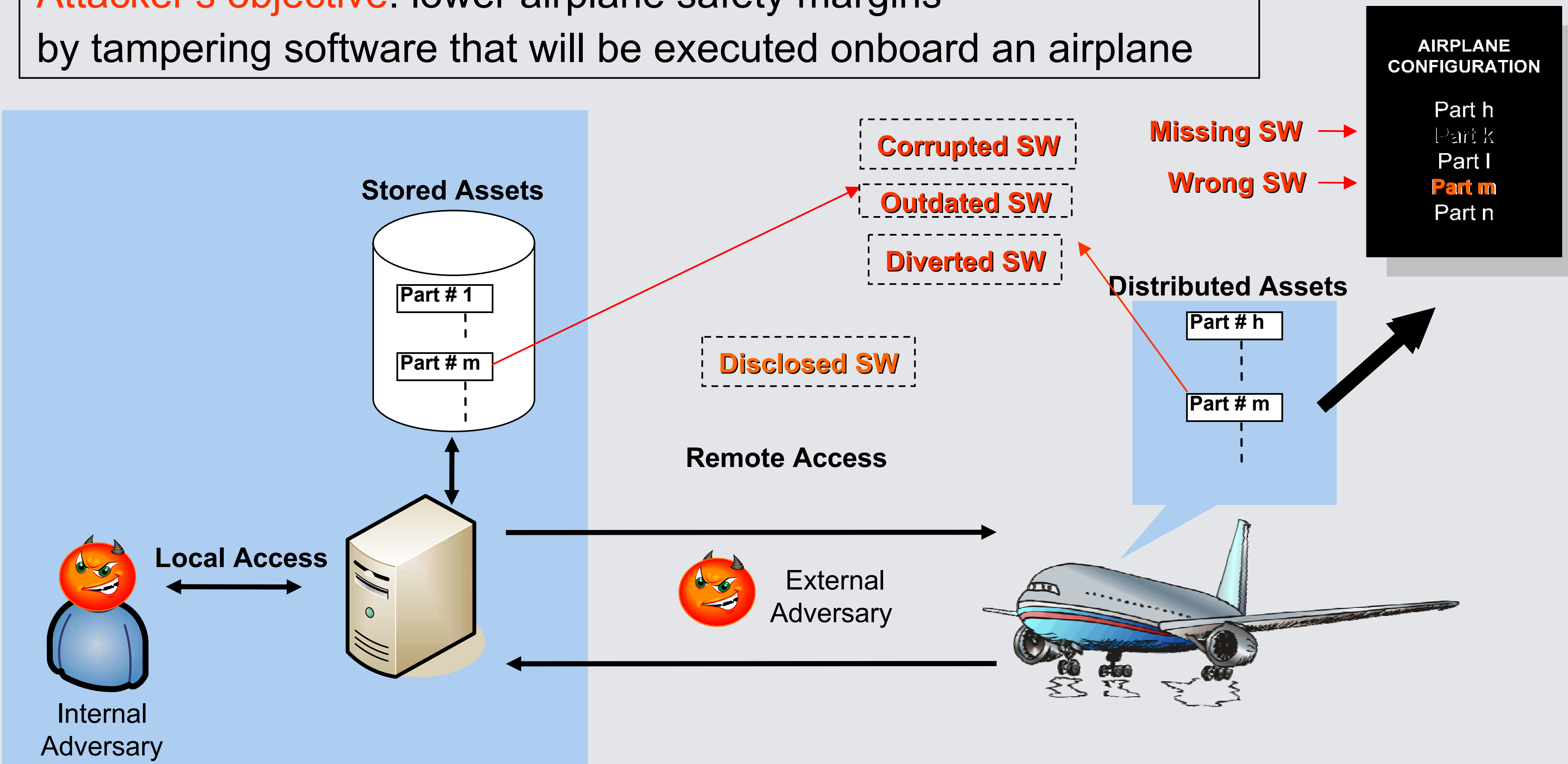
→ Software Distribution System (SDS):
System providing secure distribution of software (SW) from software supplier to target devices in the field



Transition from media-based (CD-ROMs etc.) to networked SW transport increases security risks due to transport over open, untrusted networks

Security threats at the airplane example

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



Corruption/Injection **Wrong Version** **Diversion** **Disclosure**

Overview

- **S**oftware **D**istribution **S**ystems
- Hybrid security assessment
- Alice-Bob protocol model
- Validation with AVISPA Tool
- Conclusion

Common Criteria (CC) for IT security evaluation

Common Criteria

product-oriented methodology

for IT security assessment

ISO/IEC **standard** 15408

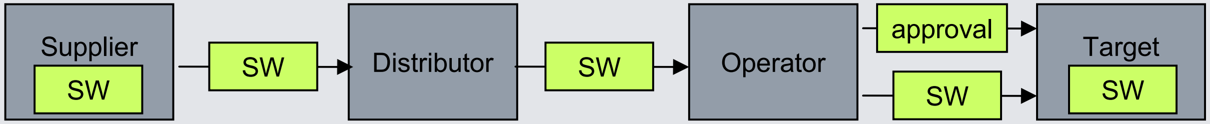
Current version: 3.1 of end-2006

Aim: gain confidence in the security of a system

- What are the objectives the system should achieve?
- Are the measures employed appropriate to achieve them?
- Are the measures implemented and deployed correctly?

Hybrid security assessment

- Highest CC evaluation assurance levels (EAL 6-7) require formal analysis
- SDS usually are complex distributed systems with many components



General problems:

- Highly critical system, but (complete) formal analysis too costly
- CC offer only limited support (“CAP”) for modular system evaluation

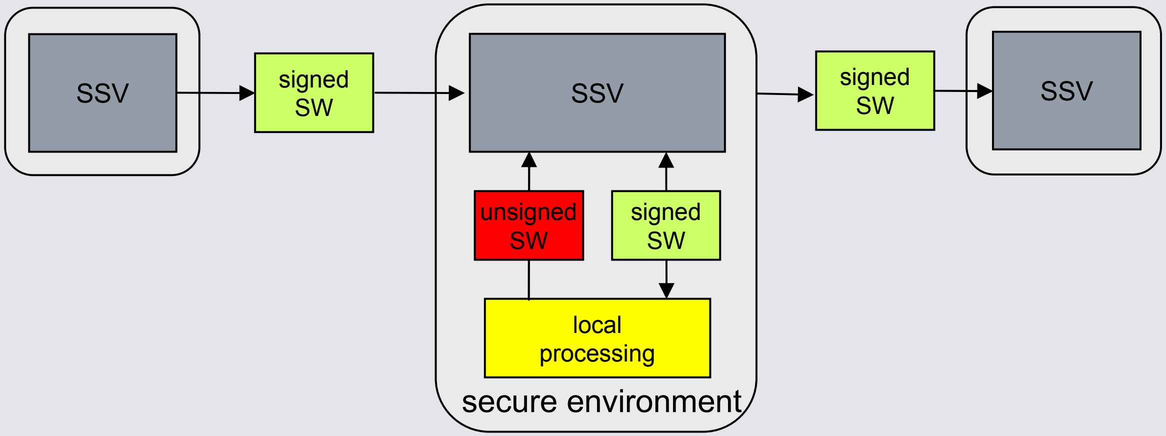
Pragmantic approach:

- Define confined security kernel with generic component: SSV
- Software Signer Verifier (SSV) handles digital signatures at each node
- Evaluate SSV according to Common Criteria EAL4 (non-formal)
- Analyze the interaction of SSVs in a formal way (→ crypto protocol)

Software Signer Verifier (SSV)

Each node in SDS runs an SSV instance, used for:

- Introducing unsigned software into the SDS,
  by digitally signing and optionally encrypting it
- Verifying the signature on software received from other SSVs,
  checking integrity, authenticity and authorization of the sender
- Approving software by adding an authorized signature
- Delivering software out of the SDS after successfully verifying it



- **S**oftware **D**istribution **S**ystems
- Hybrid security assessment
- Alice-Bob protocol model
- Validation with AVISPA Tool
- Conclusion

Formal modeling: Alice-Bob notation

```
SUP - {Asset.{h(Asset).DIS}_inv(KSUP).CertSUP}_KDIS -> DIS
DIS - {Asset.{h(Asset).DIS}_inv(KSUP).CertSUP
            .{h(Asset).OP }_inv(KDIS).CertDIS}_KOP  -> OP
OP  - {Asset.{h(Asset).DIS}_inv(KSUP).CertSUP
            .{h(Asset).OP }_inv(KDIS).CertDIS
            .{h(Asset).TD }_inv(KOP ).CertOP }_KTD  -> TD
```

`A - M -> B`	message M sent from A to B
`Asset`	a software item including its identity
`h(M)`	the hash value (i.e. crypto checksum) of content M
`M.N`	the concatenated contents of M and N
`{M}_inv(K)`	content M digitally signed with private key K
`{M}_K`	content M encrypted with public key K

Formal modeling: SDS node structure

```
SUP - {Asset.{h(Asset).DIS}_inv(KSUP).CertSUP}_KDIS -> DIS
DIS - {Asset.{h(Asset).DIS}_inv(KSUP).CertSUP
            .{h(Asset).OP }_inv(KDIS).CertDIS}_KOP  -> OP
OP  - {Asset.{h(Asset).DIS}_inv(KSUP).CertSUP
            .{h(Asset).OP }_inv(KDIS).CertDIS
            .{h(Asset).TD }_inv(KOP ).CertOP }_KTD  -> TD
```

SUP: software supplier with private key `inv(KSUP)`

DIS: software distributor with private key `inv(KDIS)`

OP : target operator with private key `inv(KOP)`

TD : target device with private key `inv(KTD)`

Signatures comprise hash value of asset and identity of intended receiver

Signatures are applied in parallel (rather than nested or discarded)

Formal modeling: approvals and certificates

```
SUP - {Asset.{h(Asset).DIS}_inv(KSUP).CertSUP}_KDIS -> DIS
DIS - {Asset.{h(Asset).DIS}_inv(KSUP).CertSUP
            .{h(Asset).OP }_inv(KDIS).CertDIS}_KOP  -> OP
OP  - {Asset.{h(Asset).DIS}_inv(KSUP).CertSUP
            .{h(Asset).OP }_inv(KDIS).CertDIS
            .{h(Asset).TD }_inv(KOP ).CertOP }_KTD  -> TD
```

- Approval information partially modelled: **operator** determines **target**
- Certificate of a node relates its identity with its public key,
  e.g. certificate of supplier SUP: **`CertSUP`** `= {SUP.KSUP}_inv(KCA)`
- Certificate authority (CA) with private key `inv(KCA)`
- Certificates are self-signed or signed by CA
- Locally stored sets of public keys of trusted SSVs and CAs

Overview

- **S**oftware **D**istribution **S**ystems
- Hybrid security assessment
- Alice-Bob protocol model
- Validation with AVISPA Tool
- Conclusion

Verification goals

Show asset authenticity, integrity and confidentiality*:*

- assets accepted by target have indeed been sent by the supplier
- assets accepted by target have not been modified during transport
- assets remain secret among the SSV instances
- asset authenticity and integrity also hop-by-hop

Correct destination covered*:*

- Name of the intended receiver in signed part, checked by target.
  Signature of the operator acts as installation approval statement

Correct version not modelled:

- Integrity of version info, *checks delegated* to SSV local environment

Formal Verification

- Alice-Bob notation not detailed and precise enough
- Use the specification language of the AVISPA Tool: HLPSL
- Software Signer Verifier (SSV) as parameterized role (node class)
- SDS as communication protocol linking different SSV instances
- Multiple protocol sessions describing individual SW transports

- Modelcheckers at their complexity limits, due to
  - parallel signatures, only the latest one being checked
  - multiple instances of central nodes (e.g. manufacturer)
  - …?

Overview

- **S**oftware **D**istribution **S**ystems
- Hybrid security assessment
- Alice-Bob protocol model
- Validation with AVISPA Tool
- Conclusion

Conclusion

- Challenges for SDS development
  - complex, heterogeneous, distributed system
  - security is critical for both safety and business

- Experience with SDS evaluation
  - Common Criteria most widely accepted methodology available
  - Problem of compositional security evaluation not solved
  - Use formal analysis where cost/benefit ratio is best
  - Highly precise design and documentation: assumptions, requirements
  - Shape system architecture to support security evaluation

- Future steps
  - Key management aspects: Public Key Infrastructure (PKI) components
  - Configuration management with installation instructions and reports