Formal security analysis and certification in industry, at the examples of an AADS\textsuperscript{1} and the AVANTSAR project

Dr. David von Oheimb
Siemens Corporate Technology, Security

Guest lecture on invitation by Dr. Ricarda Weber at the CS department of TU Munich, Germany, 04 June 2011

http://www.sec.in.tum.de/security-engineering-ss11/

\textsuperscript{1}Airplane Assets Distribution System
Overview

- IT Security at Siemens Corporate Technology
- Software distribution systems
- Common Criteria certification
- Formal security analysis
- Alice-Bob protocol model
- Validation with AVISPA Tool
- Conclusion on AADS
- Research project AVANTSSAR
Corporate Technology: Role within Siemens
Networking the integrated technology company

Customers

Chief Technology Officer (CTO)
- Review innovation strategies
- Drive technology based synergies
- Secure innovation power
- Technology assessments
- Governance and guidance

Sectors / Divisions

Industry

Energy

Healthcare

Corporate Technology (CT)

Siemens IT Solutions and Services (SIS) until June 2011
Siemens Financial Services (SFS)

Corporate Research and Technologies (CT T)
- Global Technology Fields with multiple impact
- Pictures of the Future
- Accelerators

Corporate Development Center (CT DC)
- Software development partner for the Sectors

Corporate Intellectual Property and Functions (CT I)
- Intellectual Property services & strategy
- Standardization, environmental affairs
- Global information research

Chief Technology Office (CT O)
- Direct support of CTO

Regions

Sectors / Divisions

Industry

Energy

Healthcare
Corporate Technology: around 3,000 R&D employees
Present in all leading markets and technology hot spots

- Munich
- Erlangen
- Berlin
- Vienna
- Bangalore
- Singapore
- Shanghai
- Tokyo
- Moscow
- St. Petersburg
- Romsey (RMR)
- Princeton
- Berkeley
- Brașov
GTF IT-Security – Competences ensure innovation for secure processes and protection of critical infrastructure

Competences Areas

Communication and Network Security
- Secure Communication Protocols and IP-based Architectures
- Sensor & Surveillance Security
- Security for Industrial Networks, Traffic Environments, and Building Technologies

Application Security & Methods
- Secure Service Oriented Architectures
- Enterprise Rights Management
- Trusted Computing
- Control Systems & SCADA Security
- Certification Support & Formal Methods

Cryptography
- Security for Embedded Systems
- RFID Security
- Anti-counterfeiting / anti-piracy
- Side Channel Attack Robustness
Overview

- IT Security at Siemens Corporate Technology
- **Software distribution systems**
- Common Criteria certification
- Formal security analysis
- Alice-Bob protocol model
- Validation with AVISPA Tool
- Conclusion on AADS
- Research project AVANTSSAR
Airplane Assets Distribution System (AADS)

AADS is a system for storage and distribution of airplane assets, including *Loadable Software Airplane Parts* (LSAP) and airplane health data.
Airplane Assets Distribution System architecture

A complex distributed store-and-forward middleware with OSS components

Figure is simplified and not up-to-date!
Security threats at the AADS example

Attacker’s objective: lower airplane safety margins by tampering software that will be executed on board an airplane.

- Stored Assets
  - Part #1
  - Part #m

- Local Access
  - Internal Adversary

- Remote Access
  - External Adversary
  - Distributed Assets
    - Part #h
    - Part #m

- Corrupted SW
- Outdated SW
- Diverted SW
- Disclosed SW
- Wrong SW
- Missing SW

Corruption/Injection        Wrong Version        Diversion        Disclosure
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.
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
Overview

- IT Security at Siemens Corporate Technology
- Software distribution systems
- **Common Criteria certification**
- Formal security analysis
- Alice-Bob protocol model
- Validation with AVISPA Tool
- Conclusion on AADS
- Research project AVANTSSAR
IT Security as a System Engineering Problem

- IT security aims at preventing, or at least detecting, unauthorized actions by agents in an IT system.

In the AADS context and others, security is a prerequisite of safety.

- Safety aims at the absence of accidents (→ airworthiness)

**Situation:** security loopholes in IT systems actively exploited

**Objective:** thwart attacks by eliminating vulnerabilities

**Difficulty:** IT systems are very complex. Security is interwoven with the whole system, so very hard to assess.

**Remedy:** evaluate system following the Common Criteria approach
  - address security systematically in all development phases
  - perform document & code reviews and tests
  - for maximal assurance, use formal modeling and analysis
Common Criteria (CC) for IT security evaluation

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?
CC General Approach

**Approach:** assessment of system + documents by neutral experts

- Gaining *understanding* of the system’s *security functionality*
- Checking *evidence* that the *functionality* is *correctly implemented*
- Checking *evidence* that the *system integrity* is *maintained*
Certification according to the Common Criteria is a rather complex, time consuming and expensive process.

A successful, approved evaluation is awarded a certificate.
CC: Security Targets

Security Target (ST): defines extent and depth of the evaluation for a specific product called Target of Evaluation (TOE)

Protection Profile (PP): defines extent and depth of the evaluation for a whole class of products, i.e. firewalls

STs and PPs may inherit (‘claim’) other PPs.

ST and PP specifications use generic “construction kit”:

- Building blocks for defining Security Functional Requirements (SFRs)
- Scalable in depth and rigor: Security Assurance Requirements (SARs)

layered as Evaluation Assurance Levels (EALs)
AADS Security Specification: CC Protection Profile (1)

1. Introduction
2. System Description - Target of Evaluation (TOE)
3. Security Environment
   - Assets and Related Actions
   - Threats
   - Security Assurance Requirements (EAL)
   - Assumptions
4. Security Objectives
   - ...
   - ...
Security Objectives for the AADS

- **Authenticity**: Software correctly intended destination?
- **Authorization**: Software configures properly?
- **Latest Version**: Software from correct source to intended destination?
- **Availability**: Software distributed and maintained by authorized entities?
- **Integrity**: Software altered in transit or during storage?
- **Nonrepudiation**: Security-related action on software traceable?
- **Freshness**: Can verify freshness of software?
- **01101010**: Authenticated stream: 01101010, or not?
AADS Security Specification: CC Protection Profile (1a)

1. Introduction

2. System Description - Target of Evaluation (TOE)

3. Security Environment
   - Assets and Related Actions
   - Threats
   - Security Assurance Requirements (EAL)
   - Assumptions

4. Security Objectives
   - ...
   - Rationale (Objectives and Assumptions cover Threats)
## Threats Addressed by the AADS Security Objectives

<table>
<thead>
<tr>
<th>Objectives</th>
<th>Threats</th>
<th>Safety-relevant</th>
<th>Business-relevant</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Corruption</td>
<td>Misconfiguration</td>
</tr>
<tr>
<td>Safety-relevant</td>
<td>Integrity</td>
<td>√</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Correct Destination</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Latest Version</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Authentication</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>Authorization</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>Timeliness</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Business-Relevant</td>
<td>Availability</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Early Detection</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Correct Status</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Traceability</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>Non-repudiation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Environment</td>
<td>Part_Coherence</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>Loading_Interlocks</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>Protective_Channels</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Network_Protection</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Host_Protection</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Assumptions</td>
<td>Adequate_Signing</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Configuration</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Development</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td></td>
<td>Management</td>
<td>√</td>
<td></td>
</tr>
</tbody>
</table>
AADS Security Specification: CC Protection Profile (2)

1. Introduction
2. System Description
3. Security Environment
   - Assets and Related Actions
   - Threats
   - Security Assurance Requirements (EAL)
   - Assumptions
4. Security Objectives
   - ...
   - Rationale
5. Security Functional Requirements
   - ...
   - ...
CC: Security Functional Requirements (SFRs) overview

FAU: Security audit
- Security audit automatic response (FAU_ARP)
- Security audit data generation (FAU_GEN)
- Security audit analysis (FAU_SAA)
- Security audit review (FAU_SAR)
- Security audit event selection (FAU_SEL)
- Security audit event storage (FAU_STG)

FCO: Communication

FCS: Cryptographic support

FDP: User data protection

FIA: Identification and authentication

FMT: Security management

FPR: Privacy

FPT: Protection of the TSF

FRU: Resource utilization

FTA: TOE access

FTP: Trusted path/channels
AADS Security Specification: CC Protection Profile (2)

1. Introduction

2. System Description

3. Security Environment
   - Assets and Related Actions
   - Threats
   - Security Assurance Requirements (EAL)
   - Assumptions

4. Security Objectives
   - ...
   - Rationale

5. Security Functional Requirements
   - ...
   - Rationale (omitted here)
AADS Security Specification: CC Protection Profile (3)

1. Introduction
2. System Description
3. Security Environment
   - Assets and Related Actions
   - Threats
   - Security Assurance Requirements: Evaluation Assurance Level
   - Assumptions
4. Security Objectives
   - ...
   - Rationale
5. Security Functional Requirements
   - ...
   - Rationale
<table>
<thead>
<tr>
<th>Assurance class</th>
<th>Assurance Family</th>
<th>Assurance Components by Evaluation Assurance Level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>EAL1</td>
</tr>
<tr>
<td>Development</td>
<td>ADV_ARC</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>ADV_FSP</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>ADV_IMP</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ADV_INT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ADV_SPM</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ADV_TDS</td>
<td>1</td>
</tr>
<tr>
<td>Guidance documents</td>
<td>AGD_OPE</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>AGD_PRE</td>
<td>1</td>
</tr>
<tr>
<td>Life-cycle support</td>
<td>ALC_CMC</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>ALC_CMS</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>ALC_DEL</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ALC_DVS</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ALC_FLR</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ALC_LCD</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ALC_TAT</td>
<td></td>
</tr>
<tr>
<td>Security Target evaluation</td>
<td>ASE_CCL</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>ASE_ECD</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>ASE_INT</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>ASE_OBJ</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>ASE_REQ</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>ASE_SPD</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>ASE_TSS</td>
<td>1</td>
</tr>
<tr>
<td>Tests</td>
<td>ATE_COV</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ATE_DPT</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ATE_FUN</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>ATE_IND</td>
<td>1</td>
</tr>
<tr>
<td>Vulnerability assessment</td>
<td>AVA_VAN</td>
<td>1</td>
</tr>
</tbody>
</table>
## CC: Evaluation Assurance Level 2

| Development | ADV_ARC.1 Security architecture description  |
|            | ADV_FSP.2 Security-enforcing functional specification  |
|            | ADV_TDS.1 Basic design  |

| Guidance documents | AGD_OPE.1 Operational user guidance  |
|                   | AGD_PRE.1 Preparative procedures  |

| Life-cycle support  | ALC_CMC.2 Use of a CM system  |
|                    | ALC_CMS.2 Parts of the TOE CM coverage  |
|                    | ALC_DEL.1 Delivery procedures  |

| Security Target Evaluation | ASE_XYZ (6 families of components) |

| Tests | ATE_COV.1 Evidence of coverage  |
|       | ATE_FUN.1 Functional testing  |
|       | ATE_IND.2 Independent testing - sample  |

<p>| Vulnerability analysis | AVA_VAN.2 Vulnerability analysis  |</p>
<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Development</td>
<td>ADV_FSP.4 Complete functional specification</td>
</tr>
<tr>
<td></td>
<td>ADV_IMP.1 Implementation representation of the TSF</td>
</tr>
<tr>
<td></td>
<td>ADV_TDS.3 Basic modular design</td>
</tr>
<tr>
<td>Guidance documents</td>
<td></td>
</tr>
<tr>
<td>Life-cycle support</td>
<td>ALC_CMC.4 Production support, acceptance procedures and automation</td>
</tr>
<tr>
<td></td>
<td>ALC_CMS.4 Problem tracking CM coverage</td>
</tr>
<tr>
<td></td>
<td>ALC_DVS.1 Identification of security measures</td>
</tr>
<tr>
<td></td>
<td>ALC_LCD.1 Developer defined life-cycle model</td>
</tr>
<tr>
<td></td>
<td>ALC_TAT.1 Well-defined development tools</td>
</tr>
<tr>
<td>Security Target Evaluation</td>
<td></td>
</tr>
<tr>
<td>Tests</td>
<td>ATE_COV.2 Analysis of coverage</td>
</tr>
<tr>
<td></td>
<td>ATE_DPT.2 Testing: security enforcing modules</td>
</tr>
<tr>
<td>Vulnerability analysis</td>
<td>AVA_VAN.3 Focused vulnerability analysis</td>
</tr>
</tbody>
</table>
# CC: Evaluation Assurance Level 6

**Development**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADV_FSP.5</td>
<td>Complete semi-formal functional spec. with additional error information</td>
</tr>
<tr>
<td>ADV_IMP.2</td>
<td>Implementation of the TSF</td>
</tr>
<tr>
<td>ADV_INT.3</td>
<td>Minimally complex internals</td>
</tr>
<tr>
<td>ADV_SPM.1</td>
<td>Formal TOE security policy model</td>
</tr>
<tr>
<td>ADV_TDS.5</td>
<td>Complete semi-formal modular design</td>
</tr>
</tbody>
</table>

**Guidance documents**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALC_CMC.5</td>
<td>Advanced support</td>
</tr>
<tr>
<td>ALC_CMS.5</td>
<td>Development tools CM coverage</td>
</tr>
<tr>
<td>ALC_DVS.2</td>
<td>Sufficiency of security measures</td>
</tr>
<tr>
<td>ALC_TAT.3</td>
<td>Compliance with implementation standards – all parts</td>
</tr>
</tbody>
</table>

**Life-cycle support**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATE_COV.3</td>
<td>Rigorous analysis of coverage</td>
</tr>
<tr>
<td>ATE_DPT.3</td>
<td>Testing: modular design</td>
</tr>
<tr>
<td>ATE_FUN.2</td>
<td>Ordered functional testing</td>
</tr>
</tbody>
</table>

**Security Target Evaluation**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AVA_VAN.5</td>
<td>Advanced methodical vulnerability analysis</td>
</tr>
</tbody>
</table>

© Siemens AG, Corporate Technology, Dr. David von Oheimb, 2011
CC: Factors determining the evaluation effort

- Boundary of TOE vs. TOE environment
- Definition of Threats and Security Objectives for the TOE
- Definition of Security Functional Requirements (SFRs)
- Selection of Evaluation Assurance Level (EAL)
## Selection of Evaluation Assurance Level (EAL) for AADS

<table>
<thead>
<tr>
<th>Threat Level</th>
<th>Flight safety</th>
<th>Airline business</th>
</tr>
</thead>
<tbody>
<tr>
<td>assume sophisticated adversary with moderate resources who is willing to take XXX risk</td>
<td>T5: XXX = significant e.g. intl. terrorists</td>
<td>T4: XXX = little e.g. organized crime, sophisticated hackers, intl. corporations</td>
</tr>
<tr>
<td>Information Value</td>
<td></td>
<td></td>
</tr>
<tr>
<td>violation of the protection policy would cause YYY damage to the security, safety, financial posture, or infrastructure of the organization</td>
<td>V5: YYY = exceptionally grave Risk: loss of lives</td>
<td>V4: YYY = serious Risk: airplanes out of service, or damage airline reputation</td>
</tr>
<tr>
<td>Evaluation Assurance Level</td>
<td>EAL 6: semi-formally verified design and tested</td>
<td>EAL 4: methodically designed, tested, and reviewed</td>
</tr>
</tbody>
</table>

Evaluating the whole AADS at EAL 6 would be extremely costly. Currently available Public Key Infrastructure (PKI) certified only at EAL 4. Two-level approach: evaluate only LSAP integrity & authenticity at EAL 6.
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

Pragmatic 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)
Overview

- IT Security at Siemens Corporate Technology
- Software distribution systems
- Common Criteria certification
- **Formal security analysis**
- Alice-Bob protocol model
- Validation with AVISPA Tool
- Conclusion on AADS
- Research project AVANTSSAR
Formal Security Analysis: Approach and Benefits

Mission: security analysis with maximal precision

Approach: formal modeling and verification

Improving the quality of the system specification
Checking for the existence of security loopholes

High-Level Protocol Spec. Language
Model checkers (AVISPA tools)

Interacting State Machines
Interactive theorem prover (Isabelle)
Formal Security Models

- A security policy defines what is allowed (actions, data flow, ...) typically by a relationship between subjects and objects.
- A security model is a (+/- formal) description of a policy and enforcing mechanisms, usually in terms of system states or state sequences (traces).
- Security verification proves that mechanisms enforce policy.
- Models focus on specific characteristics of the reality (policies).
- Types of formal security models
  - Automata models
  - Access Control models
  - Information Flow models
  - Cryptoprotocol models
Interacting State Machines (ISMs)

Automata with (nondeterministic) state transitions + buffered I/O, simultaneously on multiple connections.

Transitions definable in executable and/or axiomatic style. An ISM system may have changing global state. Applicable to a large variety of reactive systems. By now, not much verification support (theory, tools).
Formal model of Infineon SLE 66 Smart Card Processor

System Structure Diagram:

State Transition Diagram (abstracted):

First higher-level (EAL5) certification for a smart card processor!
Formal RBAC model of Complex Information System

Is the security design (with emergency access etc.) sound?

Privileges:
- \( \text{roles} \subseteq \text{user} \times \text{role} \)
- \( \text{subroles} \subseteq \text{role} \times \text{role} \)
- \( \text{privs} \subseteq \text{role} \times \text{privilege} \)

Permissions:
- \( \text{groups} \subseteq \text{user} \times \text{group} \)
- \( \text{subgroups} \subseteq \text{group} \times \text{group} \)
- \( \text{gperms} \subseteq \text{group} \times \text{permission} \)
- \( \text{uperms} \subseteq \text{user} \times \text{permission} \)

\[ (u, p) \in \text{roles} \circ \text{subroles}^* \circ \text{privs} \]

\[ (u, p) \in (\text{groups} \circ \text{subgroups}^* \circ \text{gperms}(e)) \cup \text{uperms}(e) \]

“nagging questions” \( \rightsquigarrow \) clarifications improving specification quality.

Open issue: relation between model and implementation (\( \rightsquigarrow \) testing).
Information Flow Models

- Identify knowledge/information domains
- Specify allowed flow between domains
- Check the observations that can be made about state and/or actions
- Consider also indirect and partial flow

- Classical model: Noninterference (Goguen & Meseguer)
- Many variants: Non-deducability, Restrictiveness, Non-leakage, ...

Very strong, but rarely used in practice

Available: connection with ISMs
Language-based Information Flow Security

Policy: no assignments of high-values
to low-variables, enforced by type system

Semantically: take \((x, y)\) as elements of the state space
with high-level data (on left) and low-level data (on right).

Step function \(S(x, y) = (S_H(x, y), S_L(x, y))\)
does not leak information from high to low
if \(S_L(x_1, y) = S_L(x_2, y)\) (functional independence).

Observational equivalence \((x, y) \simeq (x', y')\) :\(\Longleftrightarrow\) \(y = y'\)
allows reformulation:

\[
\frac{s \underset{L}{\sim} t}{S(s) \underset{L}{\sim} S(t)} \quad \text{(preservation of \(\sim\))}
\]

Generalization to action sequences \(\alpha\) and arbitrary policies \(\rightsquigarrow\)
Cryptoprotocol models

- Describe message exchange between processes or principals
- Take cryptographic operations as perfect primitives
- Describe system with specialized modeling languages
- State secrecy, authentication, ... goals
- Verify (mostly) automatically using model-checkers

EU project AVISPA, ...
Example: H.530 Mobile Roaming Authentication

Two vulnerabilities found and corrected. Solution standardized.
Overview

- IT Security at Siemens Corporate Technology
- Software distribution systems
- Common Criteria certification
- Formal security analysis
- **Alice-Bob protocol model**
- Validation with AVISPA Tool
- Conclusion on AADS
- Research project AVANTSSAR
Formal modeling: Alice-Bob notation

\[ \text{SUP} \rightarrow \{ \text{Asset}.\{h(\text{Asset}).DIS\}_\text{inv(KSUP)}.\text{CertSUP}\}_\text{KDIS} \rightarrow \text{DIS} \]
\[ \text{DIS} \rightarrow \{ \text{Asset}.\{h(\text{Asset}).DIS\}_\text{inv(KSUP)}.\text{CertSUP} \rightarrow \text{OP} \}
\]
\[ \text{OP} \rightarrow \{ \text{Asset}.\{h(\text{Asset}).DIS\}_\text{inv(KSUP)}.\text{CertSUP} \rightarrow \text{TD} \}
\]

**SUP** - a software item including its identity

**DIS** - the hash value (i.e. cryptographic checksum) of content \( M \)

**OP** - the concatenated contents of \( M \) and \( N \)

**TD** - content \( M \) digitally signed with private key \( K \)

**A** - message \( M \) sent from \( A \) to \( B \)

**Asset** - a software item including its identity

**h(M)** - the hash value (i.e. cryptographic checksum) of content \( M \)

**M.N** - the concatenated contents of \( M \) and \( N \)

**\{M\}_\text{inv}(K)** - content \( M \) digitally signed with private key \( K \)

**\{M\}_K** - content \( M \) encrypted with public key \( K \)
Formal modeling: SDS protocol structure

SUP - {Asset \cdot \{h(Asset) \cdot DIS\}_inv(KSUP) \cdot CertSUP\}_KDIS \rightarrow DIS
DIS - \{Asset \cdot \{h(Asset) \cdot DIS\}_inv(KSUP) \cdot CertSUP \cdot \{h(Asset) \cdot OP \}_inv(KDIS) \cdot CertDIS\}_KOP \rightarrow OP
OP - \{Asset \cdot \{h(Asset) \cdot DIS\}_inv(KSUP) \cdot CertSUP \cdot \{h(Asset) \cdot OP \}_inv(KDIS) \cdot CertDIS \cdot \{h(Asset) \cdot TD \}_inv(KOP) \cdot CertOP \}_KTD \rightarrow 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 linearly)
Formal modeling: SDS approvals and certificates

- Approval information partially modelled: **operator** determines **target**

- **Certificate** of a node relates its identity with its public key, e.g. certificate of supplier **SUP**: 
  \[ \text{CertSUP} = \{\text{SUP.KSUP}\}_{\text{inv(KCA)}} \]

- **Certificate authority (CA)** with private key \( \text{inv(KCA)} \)

- **Certificates** are **self-signed** or signed by **CA**

- **Locally stored sets** of public keys of trusted SSVs and CAs
Overview

- IT Security at Siemens Corporate Technology
- Software distribution systems
- Common Criteria certification
- Formal security analysis
- Alice-Bob protocol model
- **Validation with AVISPA Tool**
- Conclusion on AADS
- Research project AVANTSSAR
 Verification goals

Show asset **authenticity & integrity (end-to-end) 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 & 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:
- Version info is integrity protected, but
  *checks delegated* to SSV local environment
The AVISPA model

- **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

Detailed model omitted here
Results of the AVISPA tools

Details on use of the tools omitted here

Verification successful for small number of protocol sessions

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

- IT Security at Siemens Corporate Technology
- Software distribution systems
- Common Criteria certification
- Formal security analysis
- Alice-Bob protocol model
- Validation with AVISPA Tool
- **Conclusion on AADS**
- Research project AVANTSSAR
Conclusion (1) on AADS

- Challenges for AADS development
  - pioneering system design and architecture
  - complex, heterogeneous, distributed system
  - security is critical for both safety and business

- Common Criteria offer adequate methodology for assessment, at least for small components/systems

- Systematic approach, in particular formal analysis, enhances
  - understanding of the security issues
  - quality of specifications and documentation
  - confidence (of Boeing, customers, FAA, etc.) in the security solutions
Conclusion (2) on AADS

- Experience with SDS evaluation
  - Common Criteria most widely accepted methodology
  - 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 etc.
  - Configuration management
    - with installation instructions and status/completion reports
Overview

- IT Security at Siemens Corporate Technology
- Software distribution systems
- Common Criteria certification
- Formal security analysis
- Alice-Bob protocol model
- Validation with AVISPA Tool
- Conclusion on AADS
- **Research project AVANTSSAR**
Automated VAlidation of Trust and Security of Service-oriented ARchitectures

EU FP7-2007-ICT-1, ICT-1.1.4, Strep project no. 216471
Jan 2008 - Dec 2010, 590 PMs, 6M€ budget, 3.8M€ EC contribution
AVANTSSAR project motivation

ICT paradigm shift: from components to services, composed and reconfigured dynamically in a demand-driven way.

Trustworthy service may interact with others causing novel trust and security problems.

For the composition of individual services into service-oriented architectures, validation is dramatically needed.
Example 1: Google SAML-based Single Sign-On (SSO)

A malicious service provider can access the data of the physician located at all other services connected via Google SSO.
Example 1: Google SAML SSO protocol flaw

Fig. 1. SP-Initiated SSO with Redirect/POST Bindings
AVANTSSAR consortium

Industry
- SAP Research France, Sophia Antipolis
- Siemens Corporate Technology, München
- IBM Zürich Research Labs (part time)
- OpenTrust, Paris

Academia
- Università di Verona
- Università di Genova
- ETH Zürich
- INRIA Lorraine
- UPS-IRIT Toulouse
- IEAT Timisoara

Expertise
- Service-oriented enterprise architectures
- Security solutions
- Standardization and industry migration
- Security engineering
- Formal methods
- Automated security validation
AVANTSSAR main objectives and principles

AVANTSSAR product: Platform for formal specification and automated validation of trust and security of SOAs

- Formal language for specifying trust and security properties of services, their policies, and their composition into service-oriented architectures
- Automated toolset supporting the above
- Library of validated industry-relevant case studies

Migration of platform to industry and standardization organizations

- Speed up development of new service infrastructures
- Enhance their security and robustness
- Increase public acceptance of SOA-based systems
AVANTSSAR project results and innovation
Example 2: Electronic Car Registration policies

ACL
- anybody, get empty forms
- RegOffHead, write
- RegOffEmpl, read
- RegOffEmpl, write, if his RegOffHead says so

local policy
- RegOffCA can say who is RegOffHead who is RegOffEmpl

question:
May Peter write to CentrRep?

certificate
- Peter is RegOffEmpl of CarRegOffice (signed by RegOffCA)

certificate
- Melinda is RegOffHead of CarRegOffice (signed by RegOffCA)

certificate
- Peter can write CentrRep (signed by Melinda)

References:
- Question: May Peter write to CentrRep?
Authorization and trust management via token passing
There are three roles in the protocol (C, A, TS) and potentially several instances for each role.
The client C (or user) uses the system for SSO, authorization and trust management.
Each application A is in one domain, each domain has exactly one active token server TS.
A1 uses the system to pass to A2 some Order and an ADT (Authorization Decision Token).

- **Order** contains:
  - workflow task information
  - application data
  - information about the client C and his current activity to be delivered securely (integrity and confidentiality)

- **ADT** is mainly authorization attributes and decisions
  - sent via TS1 and TS2, who may weaken it
  - must remain unaltered, apart from weakening by TS
  - must remain confidential among intended parties C, A1, and A2 must be authenticated among each other

Security prerequisites:
PKI is used for A and TS, username & pwd for C
TS enforces a strict time-out
Example 3: ASLan++ model of A2

entity A2 (Actor: agent, TS2: agent) { % Application2, connected with TokenServer2
symbols
  C0,C,A1: agent;
  CryptedOrder, Order, Order0, Details, Results, TaskHandle, ADT, HMAC: message;
  SKey: symmetric_key;
body { while (true) {
  select {
    % A2 receives (via some C0) a package from some A1. This package includes encrypted
    % and hashed information. A2 needs the corresponding key and the Authorization Decision Token.
    on (?C0 -> Actor: (?A1.Actor.?TaskHandle.?CryptedOrder).?HMAC): {
    % A2 contacts its own ticket server (TS2) and requests the secret key SKey and the ADT.
    Actor *->* TS2: TaskHandle;
    }
    % A2 receives from A1 the SKey and checks if the decrypted data corresponds to the hashed data
    on (TS2 *->* Actor: (?ADT.?SKey).TaskHandle & CryptedOrder = scrypt(SKey,?Order0,?Details.?C)
    & HMAC = hmac(SKey, A1.Actor.TaskHandle.CryptedOrder)): {
    % A2 does the task requested by A1, then sends to A1 via C the results encrypted with the secret key.
    Results := fresh(); % in general, the result depends on Details etc.
    Actor -> C: Actor.C.A1. scrypt(SKey,Results);
    }
  }
} }

goals
  authentic_C_A2_Details: C *-> Actor: Details;
  secret_Order: secret (Order0,Details.C, {Actor, A1});
AVANTSSAR final status

WP2: ASLan++ supports the formal specification of trust and security related aspects of SOAs, and of static service and policy composition

WP3: Techniques for: satisfiability check of policies, model checking of SOAs w.r.t. policies, different attacker models, compositional reasoning, abstraction

WP4: Deploy second prototype of AVANTSSAR Platform

WP5: Formalization of industry-relevant problem cases as ASLan++ specifications and their validation

WP6: Ongoing dissemination and migration into scientific community and industry
AVANTSSR demo

- Needham-Schroeder Public Key Protocol
- TLS client and server
Formality Level: should be adequate:
  ▶ the more formal, the more precise,
  ▶ but requires deeper mastering of formal methods

Choice of Formalism: dependent on ...
  ▶ application domain, modeler’s experience, tool availability, ...
  ▶ formalism should be simple, expressive, flexible, mature

Abstraction Level: should be ...
  ▶ high enough to achieve clarity and limit the effort
  ▶ low enough not to loose important detail

refinement allows for both high-level and detailed description
Formal Security Analysis: Information Required

- **Overview**: system architecture (components and interfaces), e.g. databases, authentication services, connections, ...
- **Security-related concepts**: actors, assets, states, messages, ...
- **Threats**: which attacks have to be expected.
- **Assumptions**: what does the environment fulfill.
- **Security objectives**: what the system should achieve. Described in detail such that concrete verification goals can be set up e.g. integrity: which contents shall be modifiable by whom, at which times, by which operations (and no changes otherwise!)
- **Security mechanisms**: relation to objectives and how they are achieved. e.g. who signs where which contents, and where is the signature checked Described precisely but at high level (no implementation details required), e.g. abstract message contents/format but not concrete syntax
Development Phases and the Benefits of Formal Analysis

Requirements analysis:

- understanding the security issues
  - abstraction: concentration on essentials, to keep overview
  - genericity: standardized patterns simplify the analysis

Design, documentation:

- quality of specifications
  - enforces preciseness and completeness

Implementation:

- effectiveness of security functionality
  - formal model as precise reference for testing and verification
AVANTSSAR impact: industry migration

Services need to be securely combined according to evolving trust and security requirements and policies.

A rigorous demonstration that a composed SOA meets the security requirements and enforces the application policy will:

- significantly increase customers’ confidence
- enable customers to fully exploit the benefits of service orientation

Integration of AVANTSSAR Platform in industrial development environment

The AVANTSSAR Platform will advance the security of industrial vendors’ service offerings: validated, provable, traceable.

AVANTSSAR thus strengthens the competitive advantage of the products of the industrial partners.