Agenda

- Why do we need secure SW engineering?
- The Generic Application Security Framework
- Status of the GASF Project
The Mission Operations Infrastructure (MOI) comprises assets and services supporting the ESA multi-mission model in all phases: development, launch and operations.

The Information Security Management System (ISMS) is the implementation of the security directives resulting in requirements (SSRS) and procedures (SECOPS).
The MOI ISMS risk assessment has identified software engineering and applications as an area that requires improvement.

The new MOI SSRS and SECOPS include requirements and procedures for secure software engineering.

- This outcome is consistent with the general trend in SW security.
- **Application-Layer attacks are the Nr. 1 cyber attack type in the Internet nowadays.**
- A successful attack on the MOI has the potential to deal significant damage to the Agency and its assets!
MOI Applications
Properties & Weaknesses

• The core of each ground segment infrastructure are software components
  ➢ Complex, distributed, and interconnected systems
  ➢ Multiple languages and technologies
  ➢ Multiple generations of (long-lived) software systems

• Such software is subject to an increasing number of attacks
  ➢ Malware (Viruses, Worms, Trojan Horses)
  ➢ Direct Hacker Attacks
    ▪ Targeted Attacks
    ▪ Automated Attacks

• These attacks exploit weaknesses in the software (application-layer)
  ➢ Implementation flaws (Exploits)
  ➢ Unclean programming (e.g. not using best practices)
  ➢ Problems with third-party COTS
MOI Application-Layer Threats

- Problems with application-layer attacks
  - They are not likely to be detected by firewalls
    - They are executed using protocols that pass through network layer protection
  - They can be automated easily
    - Between 60% and 70% of attacks are automated
    - “Script Kiddies” do not need in-depth knowledge
  - Each attack is technology specific → huge variety
  - They can never be completely eliminated
    - Full formal validation of SW is impossible

→ Secure software engineering reduces the risk and minimizes the impact of an attack
  - Increased reliability, availability, and robustness
  - Problem: Doing it right is a complex endeavor
The Secure Software Development Lifecycle

- Secure Software Development is not trivial and not “one-off”
- It affects nearly every process in the Software Development Lifecycle

Requirements Analysis
  - Security Functional and Assurance Requirements

Evolution
  - Reacting to new attacks

Design
  - Secure Design Policies

Testing
  - Validation & Verification Certification

Implementation
  - Secure Coding Best Practices

Information Risk Assessment

European Space Agency
SSDLC → The overhead

### Activities

1. High Level Architecture Review
2. Classify Application and/or Data
3. Define High Level Security Controls
4. Identify Security Requirements
5. Initial Security Risk Assessment
6. Design Security Architecture
7. Code Review
8. Develop Detailed Security Design
9. Update Security Risk Assessment
10. Build Application and Security Controls
11. Initial Scan and Vulnerabilities Mitigation
12. Code Review
13. Develop Security Test Plan and Test Scripts
14. Update Security Risk Assessment
15. Complete Vulnerability Scanning
16. Execute Security Test Scripts
17. Update Security Risk Assessment
18. Mitigate / Accept Risks and Vulnerabilities
19. Deploy Security Controls
20. Update RA with Accepted Risks

### Deliverable / Work Product

A. Documented Data Classification
B. High Level Security Architecture
C. Initial Security Risk Assessment
D. Conceptual Security Design
E. Detailed Security Requirements
F. Detailed Security Design
G. Updated Security Risk Assessment
H. Build Application and Security Controls
I. Update Risk Assessment
J. Security Test Plan and Test Scripts
K. Results of Code Review
L. Results from Security Test Scripts
M. Document Risk Acceptance
N. Deployed Security Controls
O. Document Security Risk
ESAs GENERIC APPLICATION SECURITY FRAMEWORK (GASF)
ESOC started the development of **GASF - Generic Application Security Framework**

**GASF Main Objectives**

- Provide means to comply with ISMS secure software engineering requirements
- Address the complete range of ESA Software
- Limit the security process-related overhead for technical officers/ DSMs and developers
- Be based on and complement ESA software engineering standards and their processes

**GASF Outputs**

- GASF Governance and Processes Documentation
- Security Requirements and Best Practises Database
- GASF Toolset
Preparation of the software development

- Preparation of the security aspects of the development process
  - Identification & attribution of security roles
  - Preliminary global risk assessment
  - Checking need for certification
Specifications of high-level system security requirements

- Differentiation of requirement areas
- Definition of preliminary requirement filters

Selection of the supplier and establishment of a security relationship
Two categories of security requirements

- Functional → Verification
- Assurance → Validation

Various areas where security requirements are needed

- Contract with developer
- Statement of Work/ Work Description Document
- Software Documentation
- Software Requirements Specification (SRS)

Various factors influence the choice of security requirements

- Confidentiality/Integrity/Availability (CIA) rating
- Corporate security policy
- Target deployment environment
- Program/Mission specific constraints and requirements
• GASF maintains security requirements in a hierarchical database
  ➢ Areas as highest level
  ➢ Lower Hierarchy based on ISO 27001
  ➢ Types and categories as attributes
  ➢ Various requirement sources e.g. NIST
Requirements Analysis
Specification of Templates

• Requirements selection using **templates**
• Templates are filters that are applied to the requirements base
  ➢ CIA Templates
  ➢ Environment Templates
  ➢ Project Templates
• Templates are cumulative
• Requirement selections can be saved as new templates
  ➢ Easy re-use for future similar software developments
  ➢ Reduction of risk assessment overhead
Selection of final set of security requirements and architecture

- Specification of low-level security requirements
  - Further breakdown of previous step
  - Involvement of the supplier
- Definition of security software integration strategy
GASF Security Design Tools

- **Security Design Patterns**
  - Standard design blueprints for typical security requirements
    - Access Control, Secure File Transfer etc.
  - Integration into the general software design
  - Reduction of security design related development overhead

- **Data Policy Design Restrictions**
  - Design rules implied by applicable corporate or project security policies (can be related to assurance requirements)
GASF requirement base links security best practices when possible

- Best practices source is Common Weakness Enumeration (CWE)
  - CWE is built and maintained from multiple well known sources e.g OWASP
  - Each CWE entry explains how to mitigate the weakness → concrete help for developers

- Example: Buffer Overflow
  - Requirement: All buffer operations shall check input sizes
  - CWE-120: Buffer Copy without Checking Size of Input

- This helps developers implementing security requirements fast and in a standard way
  - No need for proprietary approach
  - Lends to later software verification
• GASF strongly supports security validation & acceptance
  ➢ Validation is specified in the assurance security requirements
  ➢ GASF specifies validation procedures and guidelines
    • Static Source Code Analysis
    • Penetration Tests
    • Vulnerability Scanning
• GASF supports certification
  ➢ Using assurance requirements the software owner can use GASF to aid certification e.g. NIST 140-2/3
The GASF Governance Framework

• GASF includes a governance framework that addresses
  ➢ Maintenance and evolution of the requirements base
  ➢ Maps GASF into corporate roles & responsibilities
  ➢ Integrates into the ISMS lifecycle
  ➢ Continuous improvement and adaptation

• Monitoring of CWE and other references ensures that newly discovered weaknesses are directly integrated into GASF
• Information Risk Assessment is a resource-intense process and
  ➢ Required for each software development
• GASF reduces overhead resulting from risk assessment
  ➢ For standard developments, information risk assessment will be a one-off exercise
  ➢ Requirement templates capture results of the risk assessment and can be re-used
STATUS OF THE GASF PROJECT & FUTURE ACTIVITIES
GASF Project Status (June 2013)

- GASF High Level Requirements for SOW and STC – **Available**
- Q3 2013
  - GASF Tool SDD
  - GASF specification + DSM/TO procedures (1st issue)
  - Complete top-down set of security requirements (1st issue)
- Q1 2014
  - GASF Tool + complete documentation set
  - GASF specification (final)
  - Final version of the complete set of security requirements
  - GASF Security Governance Strategy (DSM/TO and development team procedures in applying GASF)
  - Result of pilot project: software security analysis of existing system based on code review and GASF tool recommendations
The GASF project is feeding into a standardisation activity to complement ECSS software engineering standards with the secure SW engineering methodology.

Standardisation will benefit all SW development in ESA:

- Implementation of the ESA Security Directives in the area of Software Engineering should be **uniform**.
- Discharge of programmes from developing their own approach and methodologies.
- Contribution to the increase of knowledge of security implications in software development and implementation of a consistent and common mitigation approach.
Thank You! Any Questions?

- **Sources**

- **References**
  - European Cooperation for Space Standardisation - [http://ecss.nl](http://ecss.nl)
  - ESA SSA Application Security Framework, July 2011
    - A secure SW engineering framework to vaccinate operational software, Spaceops 2012
THANK YOU FOR YOUR ATTENTION

ANY QUESTIONS?

The last line of defense – Secure Software Engineering for Space Data Systems

Daniel Fischer
European Space Agency
• Security has emerged as a **Strategic Objective** for the Agency, since
  - ESA is responsible for assets of very high tangible and intangible value
  - Multiple ESA/EU Programmes have stringent security requirements: Galileo, Copernicus, SSA

• Thus, ESA needs to address security to
  - Protect its assets
  - Meet the general security requirements of new programmes
  - Stay competitive and acquire new business
  - Meet the specific needs of programs and missions