Streamlining Security Testing and Security Risk Management as part of a Secure System Engineering Framework at ESA

Marcus Wallum

Ground System Architectures Workshop 2019
26/02/2019
A recap..

- Back in 2017..

  Effective Security Engineering for the Ground Segment

  GSAW 2017
  Daniel Fischer, Mariella Spada
  14/03/2017

- Cyber security emerging as a **strategic objective** for ESA

- Proposed a Framework for **Secure Software and Systems Engineering** for the Ground Segment

- What were we talking about? Where are we now?
A recap.. SSE Standard: What are the key processes?

- Risk Assessment
- Requirements Engineering
- Design and implementation
- Verification and validation
- Operations, maintenance, disposal
- Authorisation (including accreditation & certification processes)
A recap.. What are the key support tools?

- Supporting methodologies and tools, aimed at **streamlining** implementation:

  -> Security requirements engineering (**GASF**)  
  Status: **Mature, Operational in ESA**

  -> Security Verification and Validation - Automated penetration testing (PenBox)  
  Status: Under Development

  -> Security Risk Management (**SEST**)  
  Status: Mature, Operational
A recap.. What are the key support tools? (2)

- Supporting methodologies and tools, aimed at **streamlining** implementation:

  -> Security requirements engineering (**GASF**)  
      Status: Mature, Operational in ESA

  -> Security Verification and Validation - Automated penetration testing (**PenBox**)  
      Status: **Under Development**

  -> Security Risk Management (**SEST**)  
      Status: Mature, Operational
A recap. What are the key support tools? (3)

- Supporting methodologies and tools, aimed at streamlining implementation:
  - Security requirements engineering (GASF)
    Status: Mature, Operational in ESA
  - Security Risk Management (SEST)
    Status: Mature, Operational
  - Security Verification and Validation - Automated penetration testing (PenBox)
    Status: Under Development
A recap.. What are the key support tools? (4)

• Supporting methodologies and tools, aimed at **streamlining** implementation:

  -> Security requirements engineering (GASF)
     Status: Mature, Operational in ESA

  -> Security Verification and Validation - Automated penetration testing (**PenBox**)  
     Status: **Under Development**

  -> Security Risk Management (**SEST**)  
     Status: **Mature, Operational**
PenBox: Why automate?

- Comprehensive security testing of software and systems is **complex and expensive**
- Penetration testing in particular is high-cost and **effort-intensive** - not scalable
  - Dedicated **experts** and specialised tools
  - Broad attack surface at multiple layers, however actions are **repeatable** (cf. IKC)
  - Results not easily interpretable for non-experts
- Need to remain aware of **limitations**
PenBox: Global Approach

1. Define Macro Threat Scenarios and Attack trees

Generic phases and “hack asset” attack tree applicable for majority of ground systems

- Hack asset
- Fingerprint asset (MCS)
- Verify commanding capability
PenBox: Global Approach

2. Define security **requirements profile** for ground systems
Assign as security **controls with traceability to attack nodes**

Example attack node:
**Gain access to application:**
**Brute force**

**Requirement:**
The system shall detect and lock-out repeated unsuccessful authentication attempts
PenBox: Global Approach

3. Define automation architecture for chaining penetration testing tools with modes, configuration, execution and logic 🔄🔧

- Tools require valid inputs for valid runs – parameter types and execution modes
- Valid runs execute and results are extracted to a results tree
- Pilot.py process computes valid runs for execution from results tree (chaining)
3. Define automation architecture for **chaining penetration testing tools** with modes, configuration, execution and logic.

- Tools will not run unless there are new results (no repetition)
- Tools may be “on-demand” or “continuous”
- Tool input handlers:
  - Shell (most common)
  - GUI (Xvfb, PyAutoGUI)
  - Docker (rare)
PenBox: Global Approach

3. Define automation architecture for **chaining penetration testing tools** with modes, configuration, execution and logic.

- Over 50 open source penetration testing tools benchmarked and integrated!
PenBox: Global Approach

4. Allocate tools to attack nodes to **execute** a Scenario’s attack tree on a System Under Test

<table>
<thead>
<tr>
<th>Attack tree</th>
<th>Tools</th>
<th>SUT</th>
<th>Results tree</th>
</tr>
</thead>
<tbody>
<tr>
<td>Network discovery</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port/Service discovery</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Break authentication</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gain access to network</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

4. Allocate tools to attack nodes to **execute** a Scenario’s attack tree on a System Under Test.
PenBox: Global Approach

4. Allocate tools to attack nodes to **execute** a Scenario’s attack tree on a System Under Test

- **Attack tree**
  - Network discovery
  - Port/Service discovery
  - Break authentication
  - Gain access to network

- **Tools**
  - Nmap:scan mode1

- **SUT**
  - 148.253.246.19

- **Results tree**
PenBox: Global Approach

4. Allocate tools to attack nodes to **execute** a Scenario’s attack tree on a System Under Test

**Attack tree**

- Network discovery
- Port/Service discovery
- Break authentication
- Gain access to network

**Tools**

- Nmap: scan mode2

**SUT**

- 148.253.246.19
- Port 22
- Port 80
- Port 443
- SSH
- HTTP
- HTTP-S

**Results tree**
4. Allocate tools to attack nodes to **execute** a Scenario’s attack tree on a System Under Test

**Attack tree**

**Tools**

**SUT**

**Results tree**

- Network discovery
- Port/Service discovery
- Break authentication
- Gain access to network

**Hydra:** Bruteforce SSH

- SSH
- Port 22
- Port 443
- Password: pass123
- HTTP-S
- Port 80
- HTTP
PenBox: Global Approach

5. Implement logic from results to prove scenario success and test and verify security requirements

- Trigger/success nodes defined per requirement
- Executes requirement assessment logic
- Assessment result pushed to result tree
PenBox: Global Approach

6. Display results and generate reports

| Domain | IP range | IP | Port | Transport protocol | Application protocol | Application | Version |
|--------|----------|----|------|--------------------|----------------------|-------------|---------|---------|
PenBox: Global Approach

6. Display results and generate reports

- All log data is recorded
- Full **traceability**:
  scenario-attack node-tool-result-SUT-requirement
- Enables **detailed report generation**
PenBox: Proof of Concept successful!

- **Automated** security testing for a SDLC!

- SUT a (segregated clone of) virtual ESOC **Ground Segment Reference Facility**
SEST: Brief Overview

- Security Risk assessment: Another typically complex and cumbersome process

- SEST Tool (web-based) enables a guided and semi-automated implementation of a risk assessment methodology (MEHARI)

- For use at earliest phase of the system engineering lifecycle
SEST: Brief Overview – Data Model

- Multiple Projects and associated profiles for re-use
- Risk Assessment procedure uses various data models:
  - Asset model
  - Threat model
  - Vulnerability model
  - Safeguard model (Requirements)
- Computed risk scenario, residual risk and risk treatment models
SEST: Brief Overview - conclusions

- Guided methodology and **easy UI**
- **Increased efficiency** (with limitations)
- Automatic computations (malfunctions<->business processes, seriousness<->risk scenarios, etc.) for **faster iterations**
- Exportable results and project templates for **re-use. Support to Audits**
- Asset model candidate for future **MBSE link**
- **Import/mapping of requirements** (GASF)
Secure Systems Engineering for Space: Future

- Follow-up activity “SSE4Space”: consolidate and integrate to form a streamlined **framework for secure systems development**

- Integration with **MBSE** framework
- Space **Data Link** Security
- **Space Segment** Security

- Assuance: Integrated **Certification concept**
- Additional testing tools
Thankyou for your attention!

Questions?
Secure Systems and Software Engineering

- ESOC has lead a gap analysis and subsequent definition of an ESA-internal Secure Software Engineering (SSE) standard (released 2016).
  -> Panel and WG composed of representatives from all programmes and directorates

- ESSB-ST-E-008: Secure Software Engineering Standard (normative)
  -> Standardizing secure SW engineering processes identified by the gap analysis

- ESSB-ST-E-007: Secure Software Engineering Handbook (non-normative)
  -> Complementing the standard: guidelines and recommendations

- ESSB-HB-E-009: Glossary of Secure Software Engineering Terms

- Applicable standard for the ESOC ISMS and all in-house SW developments
- Full adoption at ECSS level is planned
GASF: General Application Security Framework

• Security requirements specification and management is a **complex subject**

• The General Application Security Framework tool (GASF) is an **easy-to-use** tool:
  - **Simplifies** the application of a complex subject matter for non-experts **whilst not diminishing** the suitability and effectiveness of security controls
  - Permits the **efficient** definition of security requirements for a mission, system or software development
  - Supports **approval workflows** and informed decision making
  - Supports **document generation** (SRS)

The framework consists of 3 pillars:
  1. Security requirement **catalogues**
  2. Context-specific **profiles** which specify **needs**
  3. Requirements engineering **tool**
GASF: Security Requirement Catalogues

• **Structured** set of security requirements that may be *used as reference* for composition of a selection / profile

• Derived from **well-known sources**:

  - **GASF Evolution Requirement Catalogue** compiles lessons learned and existing security requirements catalogues from across the agency:
    - ESA SSE
    - ESRIN
    - ESTEC

  - Merged catalogue is a candidate for use as a *reference throughout the Agency*
GASF: Tool

- Web Application
- Requirements can be amended or commented
- Projects can be compared
- Ad-hoc discussions and comments support collaboration
- Search engine
- Capture of details and history (versioning, ID, verification method etc)
- Generation of reports (PSO can review deltas verses profile recommendations) and SRS
GASF: Workflow

- Extensive and comprehensive catalogue of requirements to apply
  - Profiles ensure requirements are adapted to the context
    - Declare already implemented controls
    - Require controls to be implemented
  - Recommendations computed to guide final selection
- Export function:
  - PSO or audit report
  - .csv file (e.g. for DOORS import)
  - Security Requirements Specification document