Combining Risk Assessment and Security Testing

Jürgen Großmann, Martin Schneider, Johannes Viehmann, Marc-Florian Wendland

Fraunhofer FOKUS

Overview

RACOMAT

Risk Assessment COMbined with Automated Testing

Table of content

- Introduction
- Problems and challenges
- State of the art
- The RACOMAT method
- The RACOMAT tool
- Case studies
- Conclusion and future work

Introduction
Importance of Risk Management

Why just identifying risks is not enough

- Example: 2013 global surveillance disclosures
  - German government justified NSA spying till September 2013 because "security is a super basic right"
    (Hans-Peter Friedrich, German minister of the interior, own translation)
  - Their opinion changed dramatically as soon as they learned that the mobile phone of German chancellor Angela Merkel was obviously observed, too
  - ➢ Which risk is higher? Living in an Orwellian surveillance for sure or being eventually not able to prevent some act of terrorism?

Introduction
Importance of Risk Management for ICT-Systems

Basic observations

- Heterogeneous cross linked ICT-Systems of growing complexity are a key factor in modern industries and societies
- Security is crucial in various market sectors, including IT, health, aviation and aerospace.

Why Risk Management is required

- In the real world, perfect security often cannot be achieved
  - There are residual risks for any complex ICT-System
- Risk assessment und risk treatment can help to create trust by:
  - Communicating residual risks
  - Help to implement safeguards and treatments for to high risks in order to reduce the risks

Problems and Challenges

Risk Assessment and Security Testing

Risk assessment might be difficult and expensive

- Hard for large scale systems
- Is highly dependent on the skills and estimates of analysts

→ Make risk analysis more objective with testing

Security testing might be difficult and expensive, too

- Testing for unwanted behavior – there is no specification what to expect
- Even highly insecure system can produce lots of correct test verdicts if the “wrong” test cases have been created and executed
- Manual testing is error prone and infeasible for large scale systems

→ Automate security testing using risk assessment

State of the Art
Risk Assessment, TBRA, RBST

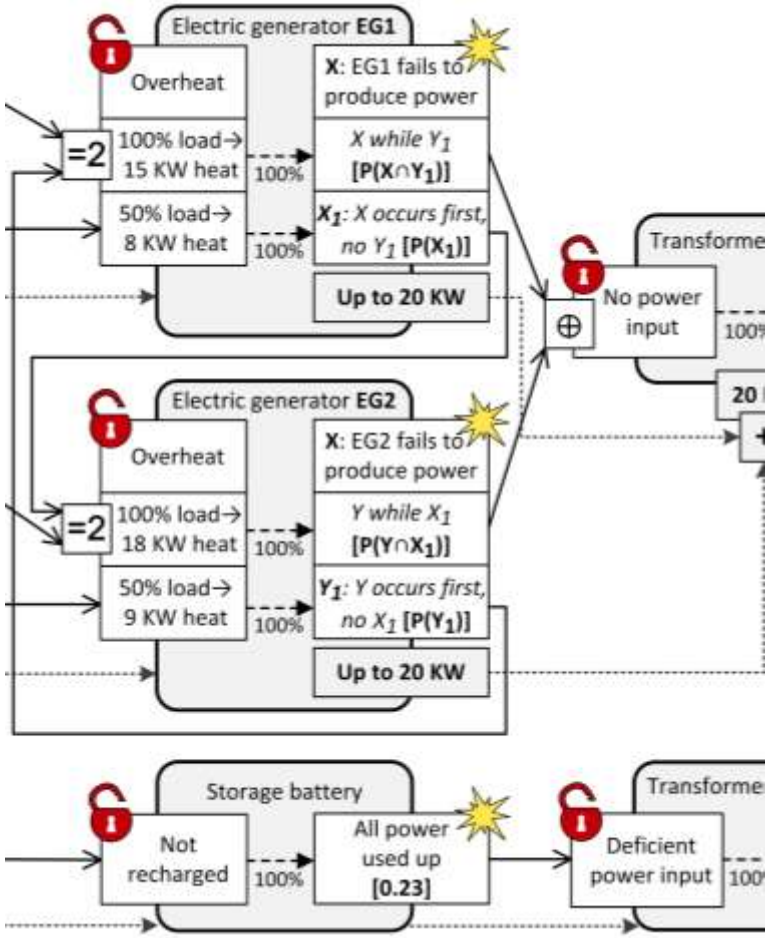
Methods for Risk Assessment

- FMEA/FMECA, FTA, ETA, CORAS …
- Compositional Risk Analysis
- Standard: ISO 31000

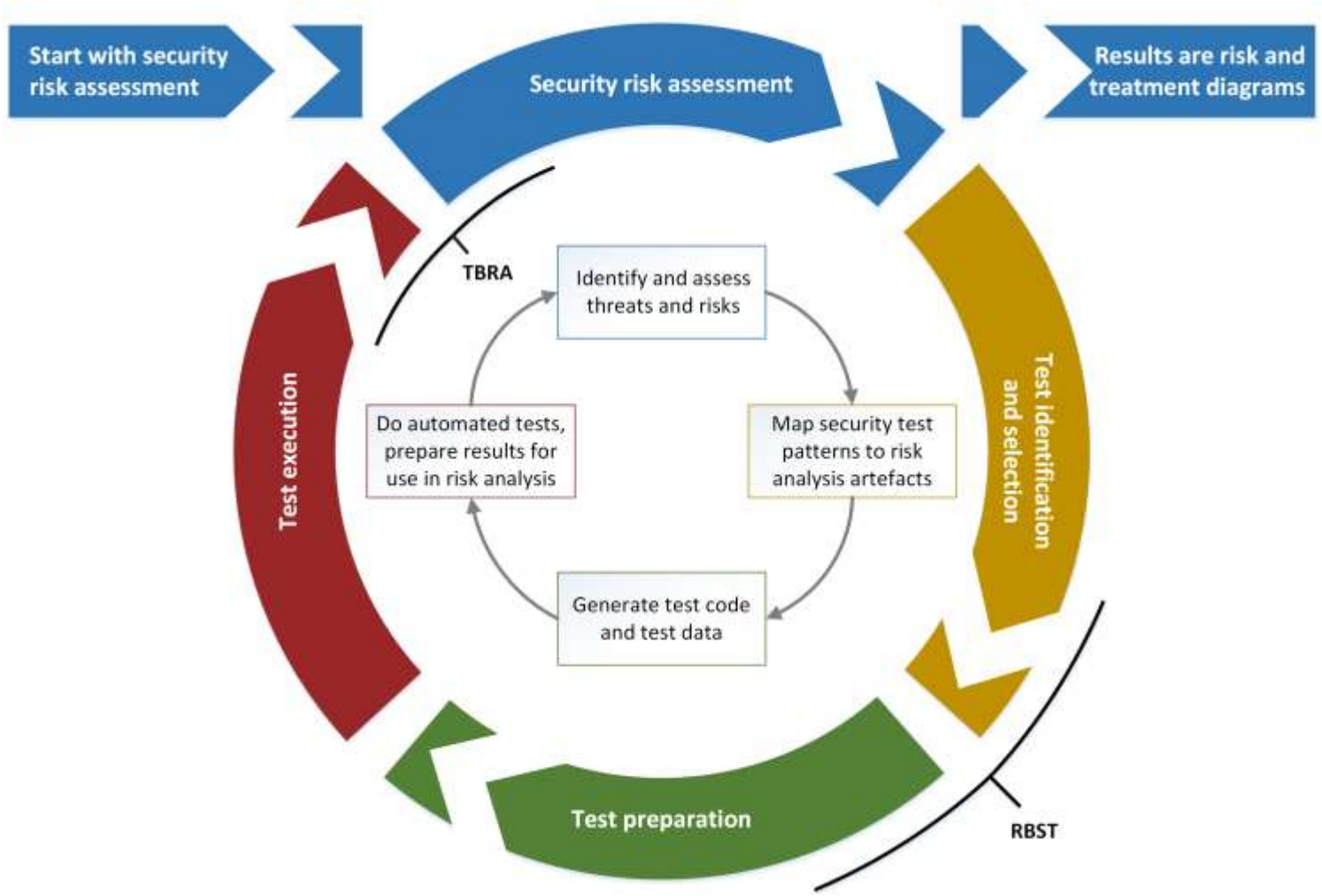
Combination of risk assessment und security testing

- Test-Based Risk Assessment (TBRA)
  - Improve risk assessment with results of security tests
- Risk-Based Security Testing (RBST)
  - Optimize security testing with results of risk assessment
- Combination of TBRA and RBST
  - No specific method established

→ The RACOMAT Method should close the gap



The RACOMAT Method
Iterative Process



The RACOMAT Method

Levels of Interaction Between Risk Assessment and Security Testing

1. Identification

What should be tested?

2. Prioritization

Spend how much effort for which tests?

3. Generation

Which test cases should be created?

4. Execution

How to stimulate and observe? Where to stimulate and observe?

5. Feedback

What do the test results mean for the overall risk picture?

The RACOMAT Method
Reusability and RBST Automatization

- Component based, low level risk assessment
  - Reusable risk assessment artifacts
  - Compositional risk analysis
  - Connection with system components
- Security testing is a part of the risk analysis
  - Automated risk-based security testing with the help of Security Test Pattern

  Security test pattern contain:
  - Strategies, models und code snippets for test case generation, test execution and test observation
  - Generic links between test pattern, risk analysis artifacts and system components
  - Information about testability and test effort, user feedback
  - Metrics or links to metrics and information how to use them with the test pattern

The RACOMAT Method
Reusability and TBRA Automatization

What do raw test results mean?

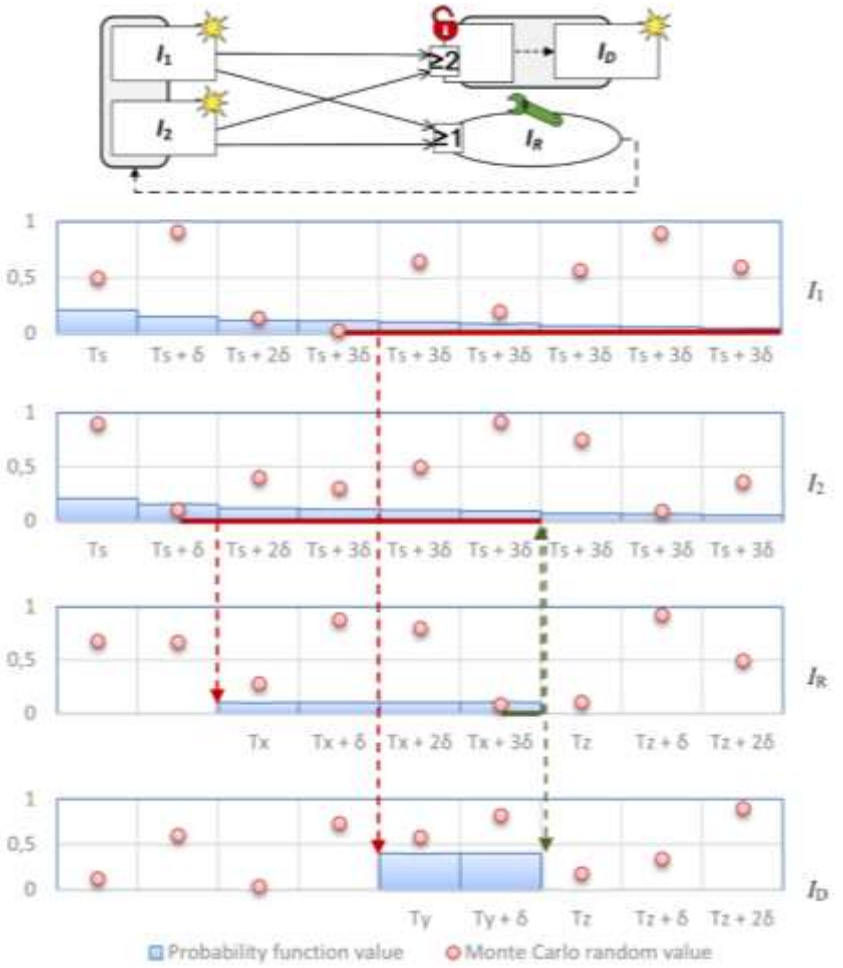
- Proper interpretation is not trivial, especially if nothing unwanted has been triggered
  - ➢ Try to offer reusable artifacts that help
- Security Testing Metrics provide generic functions for evaluating results from security testing
  - Within RACOMAT, such metrics are used for the TBRA step (i.e. results are risk artifacts)

    Security testing metrics contain:
    - Category (e.g. list up metrics, coverage metrics, efficiency metrics, technical impact metrics)
    - Machine readable interface description
      - Parameters, return value
    - Executable or machine interpretable functions
      - Enabling complete automatization
    - Human readable description
    - User feedback (e.g. ratings, comments, results)

But how to create sound security testing metrics?

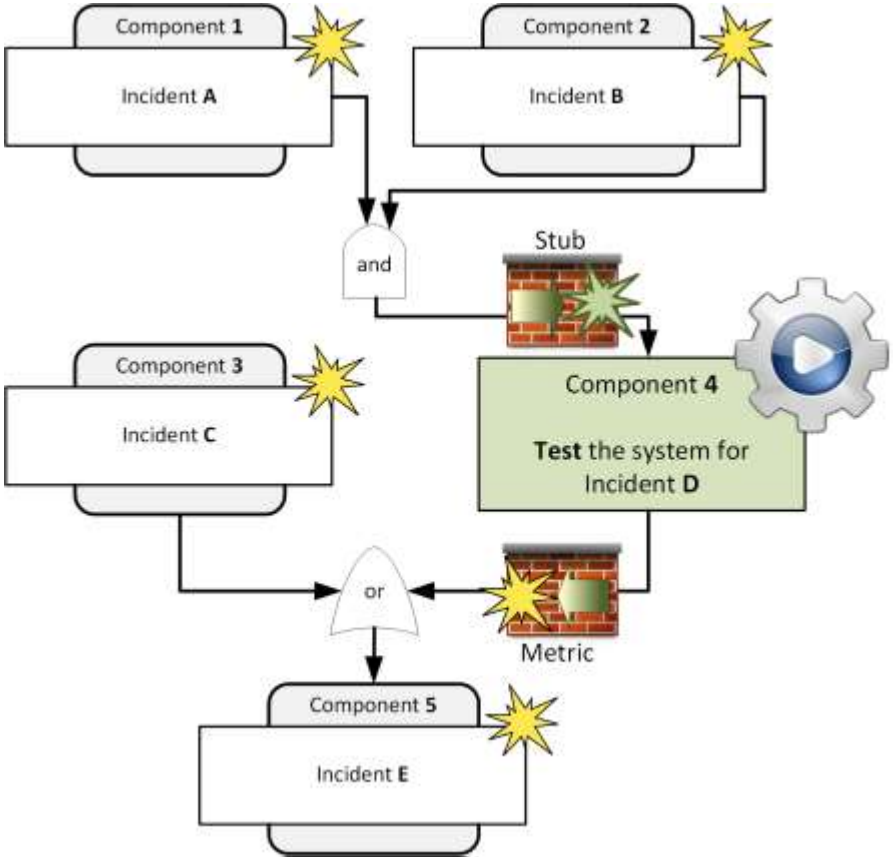
The RACOMAT Method
Security testing metrics and stubs

- Starting position for complex, large scale systems:
  - Testing the entire system is expensive – simulate it!
    - Create an event graph (e.g. a fault tree) containing the relevant incidents
    - Model dependencies using relations, gates
    - Estimate likelihoods for the base incidents and relations
    - Simulation (e.g. Monte Carlo Simulation) can then be used to approximate likelihoods for dependent events



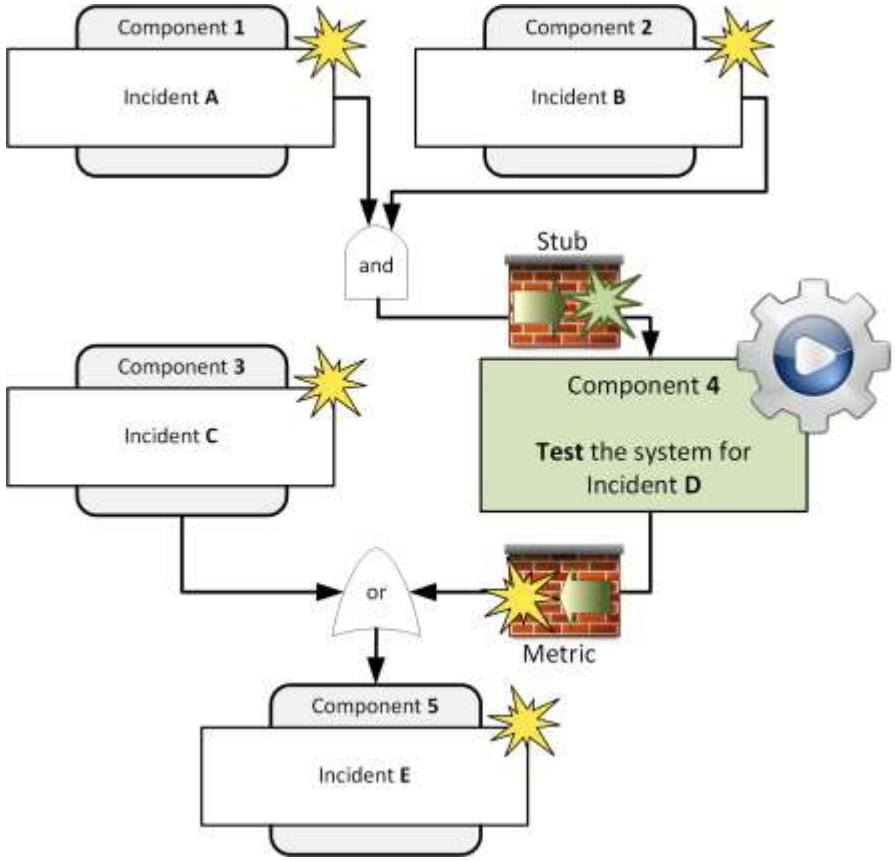
The RACOMAT Method
Security testing metrics and stubs

- Starting position for complex, large scale systems:
  - Testing the entire system is expensive – simulate it!
    - Create an event graph (e.g. a fault tree) containing the relevant incidents
    - Model dependencies using relations, gates
    - Estimate likelihoods for the base incidents and relations
    - Simulation (e.g. Monte Carlo Simulation) can then be used to approximate likelihoods for dependent events
- Replace most critical / most uncertain estimated component with the real system and test it
  - Base incidents may be created with stubs



The RACOMAT Method
Security testing metrics and stubs

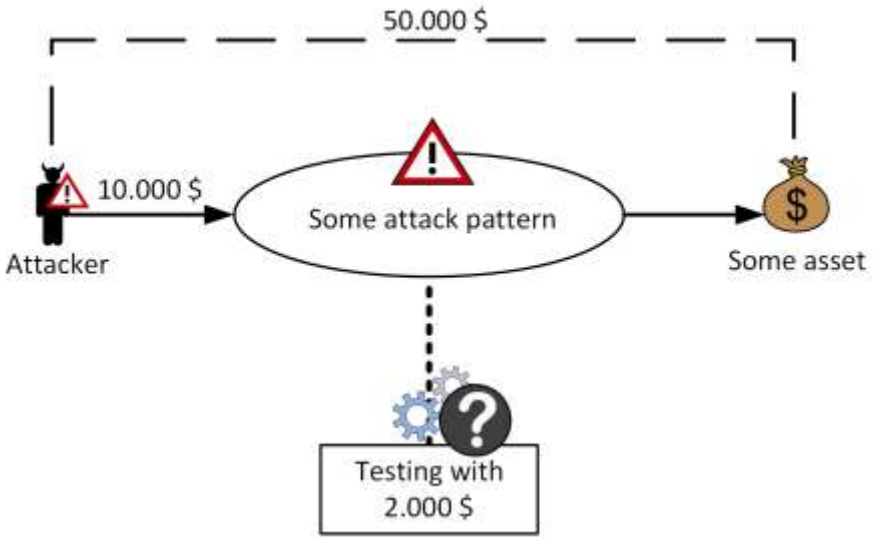
- Starting position for complex, large scale systems:
  - Testing the entire system is expensive – simulate it!
    - Create an event graph (e.g. a fault tree) containing the relevant incidents
    - Model dependencies using relations, gates
    - Estimate likelihoods for the base incidents and relations
    - Simulation (e.g. Monte Carlo Simulation) can then be used to approximate likelihoods for dependent events
- Replace most critical / most uncertain estimated component with the real system and test it
  - Base incidents may be created with stubs
- With a simple list up metric it is possible to
  - use expected incidents as triggers in the simulation – updates dependent likelihoods
  - extend the risk graph with unexpected incidents



The RACOMAT Method
Security testing metrics and economics

A more advanced efficiency metric:

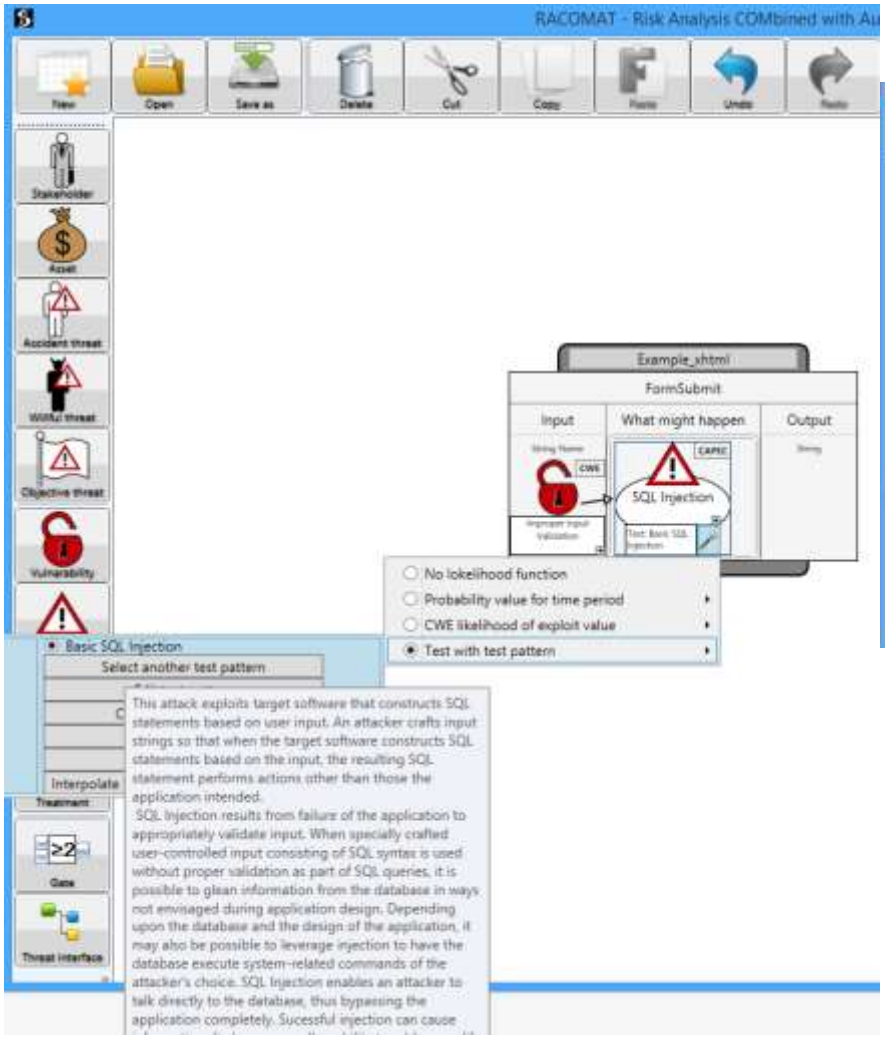
- Idea: Try to figure out *P* indicating how likely it is that an attacker will apply the attack pattern that was used for testing successfully?
  - In future simulations, that likelihood *P* will be used instead of testing the component again
- Input:
  - *R*: testing results: number of times unwanted incident was triggered
  - *T*: how much budget was spend for testing
  - *A*: estimated budget of deliberate human threats for such an attack
- A metric could define a function to calculate a probability value like that the attack will occur, e.g.:
  - $P = \left(1 - \frac{1}{(\sqrt{2})^{A*(1+R)/T}}\right)$



R \ T	2000	4000	10000	20000
0	0,82	0,58	0,29	0,16
1	0,97	0,82	0,50	0,29
2	0,99	0,92	0,65	0,41

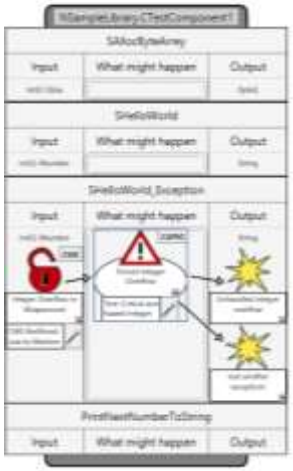
The RACOMAT Tool
Features and Workflow 1/2

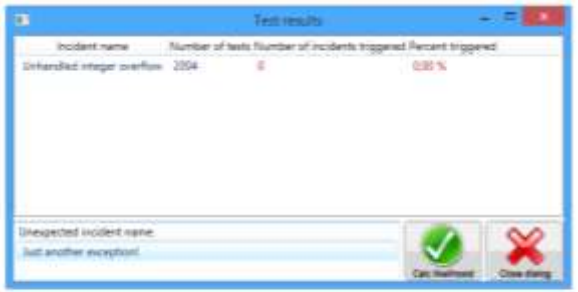
- System analysis and risk assessment
  - Automatically creates interface models for programs, APIs, components, Web-Pages or Web-Services
  - Generates semi automatically initial fault trees or CORAS risk graphs
    - Uses risk catalogues (Mitre CWE / CAPEC, BSI IT-Grundschutz …)
  - Edit and compose per Drag and Drop
  - Calculates likelihoods for dependent incidents automatically
- Security Test Pattern instantiation
  - Suggests associations with identified threat scenarios and system components
  - Calculates, how much test effort should be spend



The RACOMAT Tool
Features and Workflow 2/2

- Execution of tests
  - Once a test pattern is instantiated, generating, executing and evaluating tests woks at least semi automatically
    - Often no manual work is required at all, e. g. for overflows or (SQL-) Injections
- Updates the risk picture based upon the test results semi automatically
  - Makes suggestions using the metrics of the security test pattern
    - More precise likelihood values
    - Allows to add unexpected observations as new faults or unwanted incidents by dragging them to the risk graph



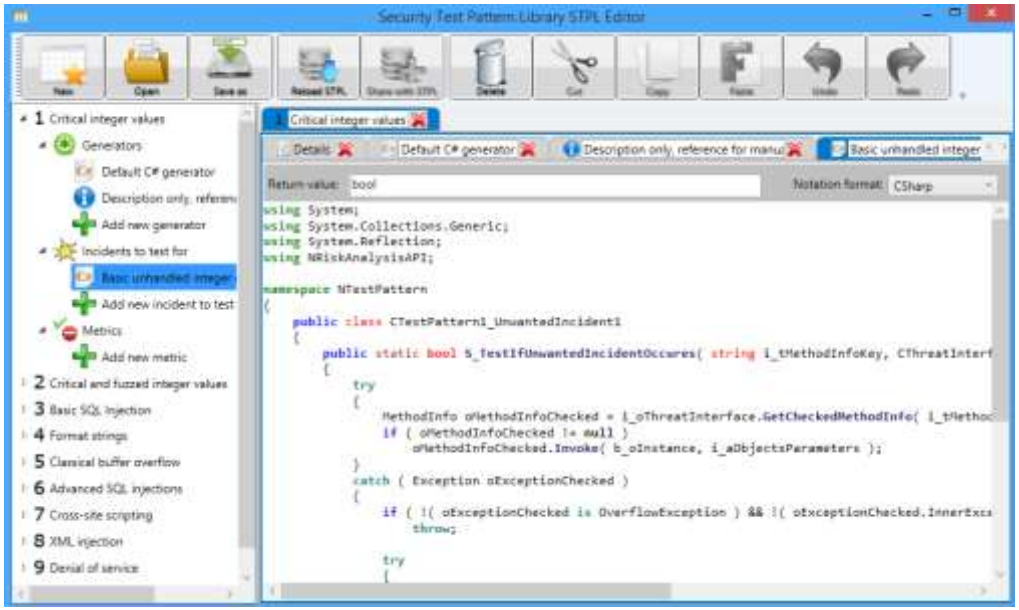


The RACOMAT Tool
Security Libraries

Security Test Pattern Library STPL: a catalogue of security test pattern for most common attacks

- If there are no fitting test patterns, new test pattern can be created using the RACOMAT Tool
- User can contribute feedback and they can suggest extensions for the open STPL
  - Quality management with ratings / comments of the users

Security Testing Metric Library STML: a catalogue of security testing metrics



The RACOMAT Tool – Demo

Case Studies
First experiences from praxis

- RACOMAT method and tool are tested in two case-studies for modular large scale systems
  - S-Network (Fraunhofer, H-C3 TU Berlin, http://surn.net)
  - Command Central (Software AG, EU-FP7 funded project RASEN, http://www.rasenproject.eu)

Positive experiences

- The assistants and the libraries of predefined artifacts help to avoid that the analysts miss important aspects
  - Negative risk assessment: remove not relevant threats instead of looking for the relevant threats
- Reusing artifacts helps to reduce the need to reinvent the wheel each and every time – hence, it reduces the potential for analysts and testers to make errors

Problems

- There are currently only a few useable security test pattern and security testing metrics
  - It is difficult to make sound estimates for the test quality, test effort and especially for generic test evaluation and metric functions

Conclusion and Future Work

- RACOMAT method and tool already combine risk assessment with security tests tightly
  - Other analysis methods: Simulation, monitoring, verification, review …

  - Basic threat simulation (Monte Carlo simulation) already implemented into RACOMAT
- Assistance for analysis of external cloud services (outsourcing)
- Vision: Open Risk Assessment – Community Driven Risk Analysis

Questions, Remarks?

Thanks a lot for the attention!

Johannes Viehmann 2014

Contact

Fraunhofer Institute for Open Communication Systems FOKUS

Kaiserin-Augusta-Allee 31
10589 Berlin, Germany

www.fokus.fraunhofer.de

Johannes Viehmann
johannes.viehmann@fokus.fraunhofer.de

System Quality Center SQC

http://s.fhg.de/sqc

Dr. Tom Ritter
Head of competence center SQC
tom.ritter@fokus.fraunhofer.de

Friedrich Schön
Head of competence center SQC
friedrich.schoen@fokus.fraunhofer.de