1 The RACOMAT tool

General information	
Name	RACOMAT (Risk Analysis COMbined with Automated Testing) Tool
Provider	Fraunhofer FOKUS
Topic addressed	The entire combined test-based risk assessment (TBRA) and risk-based security testing (RBST) process.
Description	Utilizing static and dynamic analysis techniques, the RACOMAT tool generates initial editable fault trees or CORAS risk graphs with linked system models for programs, libraries, components and web interfaces. Users decide which elements from existing risk related libraries like Mitre CAPEC and CWE should be added. The RACOMAT tool supports compositional risk analysis with simple drag and drop for existing artefacts and it calculates likelihoods for dependent incidents by performing Monte Carlo simulations. Security test patterns are automatically associated with risk analysis artefacts as well as system model components (e.g. input and output ports) and their priority is calculated. If no appropriate test patterns exist in the library, the tool allows its users to create new test patterns within the tool and to upload them to the library for sharing. Given an appropriate test pattern, test generation, execution and result aggregation are at least semi-automated. Indeed, there is no need for additional manual work at all for testing many common issues like overflows or SQL injections. Security testing metrics suggested by the test patterns can be used to analyze and evaluate test results. New security testing metrics can be created and edited in the RACOMAT tool. With appropriate security testing metrics, it is possible to update the risk graphs automatically with more precise likelihood estimates or new faults / incidents based on the test results. Besides using the RACOMAT tool as a stand-alone tool, it is possible to use the RACOMAT tool as an integration platform and to utilize other eventually more specialized tools for some steps in the combined TBRA and RBST process.
License	RASEN project partner can obtain a license for the project duration. The final license model is not yet decided.
Website	http://www.rasenproject.eu/the-racomat-tool-2/
Technical information	
Download site	Public beta planned for fourths quarter 2015
OS	Windows Vista or newer Windows version
Technology environment	.Net 4.5, WPF
Other dependencies	none
Additional information	
Known issues/risks	RACOMAT security testing is meant to attack SUTs – use with caution!
Additional useful information	Since there is not yet a public security test pattern library server online, only a local test pattern library can be used in the delivered RACOMAT tool version.

2 Description of the Prototype Tools

Risk assessment might be difficult and expensive, it often depends on the skills and estimates of the analysts. Testing is one analysis method that might yield more objective results, but security testing itself might be difficult and expensive, too, because security testing means to test for unwanted behavior and there is usually no specification what to expect. Besides that manual testing is itself error prone and infeasible for large scale systems, even highly insecure system can produce lots of correct test verdicts if the “wrong” test cases have been created and executed. Therefore, it makes sense to do Risk Assessment COMbined with Automated Testing, i.e. to use RACOMAT Tool. The RACOMAT tool is intended to cover the entire process of combined test-based risk assessment (TBRA) and risk-based security testing (RBST) within a single standalone tool. It allows to apply and evaluate the RASEN Method without troubling about interoperation and interaction between different tools.

However, more specialized tools might be more powerful than the RACOMAT Tool for some tasks. Eventually, use case partners do already use some existing tools for some steps of the risk assessment and security testing process. Therefore, we try to build bridges from the RACOMAT tool to other tools, especially to those developed or used by our project partners. The idea is to make the RACOMAT Tool the central integration platform which can be used to control the overall combined TBRA and RBST process while for some tasks within that process, other tools are used. Since the RACOMAT Tool itself implements the entire process, there will be no gaps, for sure, even if for some steps within the process there were no other tools than the RACOMAT Tool available.

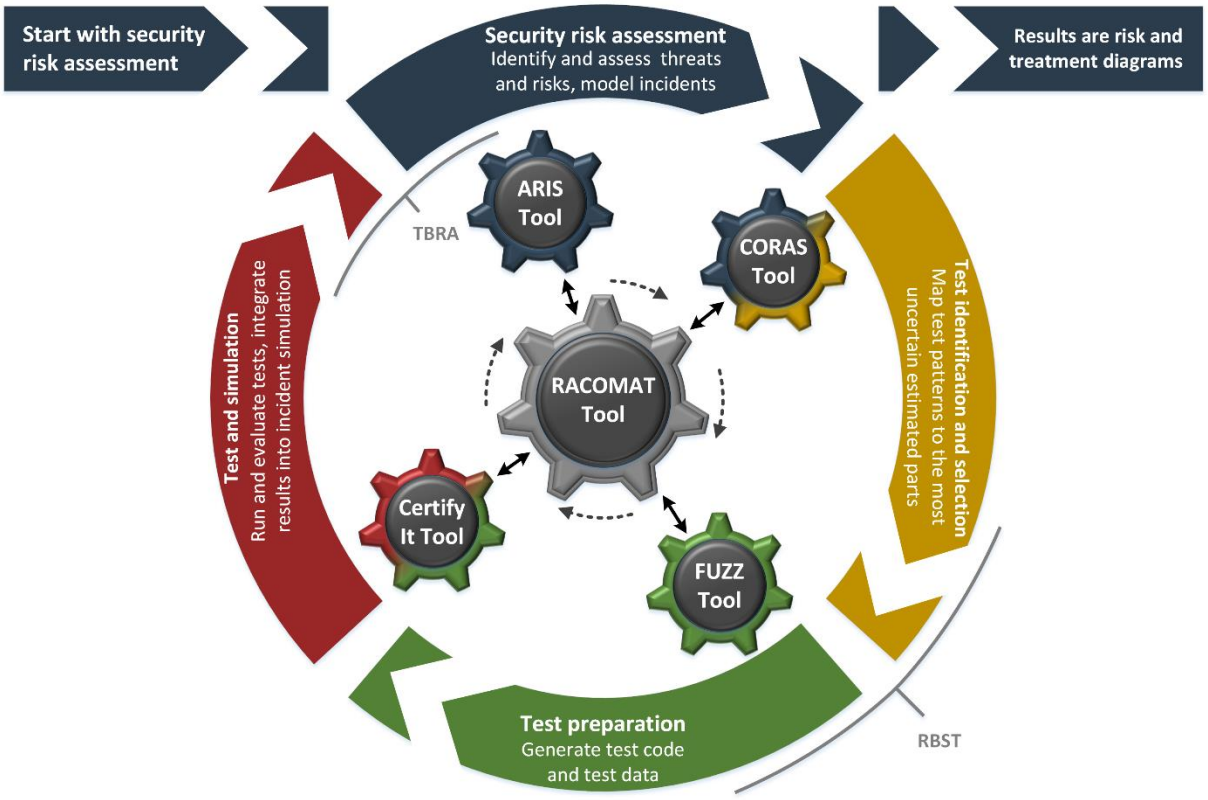

Figure 1 – The process of the RACOMAT Method with the RACOMAT tool as a platform integrating various other tools

2.1 Features of the RACOMAT Tool

Current Status

- Component based, low level system analysis and risk assessment
    - Automatically creates interface models for programs, APIs, components, Web-Pages or Web-Services
    - Generates semi automatically initial fault trees (FTA), event trees (ETA) or CORAS risk graphs
    - Reusable risk assessment artifacts – uses existing risk catalogues (Mitre CWE / CAPEC, BSI IT-Grundschutz …)
    - Compositional risk analysis,edit and compose per Drag and Drop
    - Calculates likelihoods for dependent incidents automatically
        - Supports timing issues – likelihoods might change over time
    - Identifies and prioritizes elements worth further investigation
    - Allows to model relations between risk artefacts and with system components
- Supports security testing as a part of the risk analysis process
    - Security Test Pattern instantiation
        - Suggests test patterns for identified threat scenarios
        - Assisted association with risk artefacts and system components by drag and drop
            - Indicates where to stimulate and what to observe
    - Execution of tests, observation
        - 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
        - Automatic observation and basic aggregation of raw test results
- Updates the risk picture based upon the test results semi automatically
    - Makes suggestions using the metrics associated with 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
        - Automatically creates relations to related threat scenarios that were tested
- Create, edit and share reusable artefacts
    - Threat interfaces
    - Security test patterns
    - Security testing metrics
- Flexible dashboard showing final risk assessment results for the management
- Interoperability
    - Import and export in XML RASEN exchange formats
    - Import from and export to ARIS Business Architect (Software AG case study)

Planned Features

The most relevant upcoming features expected within the first public beta release are the following:

- Calculate how much test effort should be spend for which tests (monetary value)
- Further import / export functionality

- Integration into Microsoft Visual Studio IDE
    - Plug-in
    - Use powerful Visual Studio code editor, debugging tools etc. for creating and editing test patterns and testing metrics directly within Visual Studio
- Public RACOM server to share threat interfaces, test pattern and testing metrics

2.2 Installation of the RACOMAT Tool

To install the RACOMAT tool, just execute the setup executable contained in the RACOMAT.zip file on a computer running Microsoft Windows Vista or a newer Microsoft Windows version. For best compatibility, we recommend using the latest Microsoft Windows 10 version.

The RACOMAT tool requires Microsoft .NET 4.5 or a later, compatible version. The RACOMAT setup program should automatically download and install the .NET framework if no appropriate version has been installed before. Eventually the user has to confirm the installation of .NET and accept the related Microsoft license terms. The latest version of .NET can also be obtained manually from https://www.microsoft.com. Note that an internet connection will be required anyway to download the .NET setup program since it is not included in the RACOMAT.zip file.

2.3 Documentation of the RACOMAT tool

RACOMAT is still a prototype. The online documentation is currently under construction. However, there are already some tool tips in the program and there is a short Video Demo / Tutorial showing the basic workflow included in the tool deliverable ZIP file.

Here we also give a short tutorial introducing the most important concepts. For demonstration, a web application called Damn Vulnerable Web Application (DVWA) will be used here, because it has many weaknesses. DVWA can be obtained from here: http://www.dvwa.co.uk/.

After starting RACOMAT, users will see an empty risk graph. By dragging elements from the toolbar on the left to the risk graph, it is possible to manually create a risk model. Drag one threat interface to the graph. Choose “HTTP interface” from the small dialog that pops up.

The http interface assistant will be opened, which is most appropriate to create the initial system model for an application having an http based web interface like DVWA.

The http assistant basically contains a web browser. Just enter an URI in the address field and hit return or click the “Go” button. While browsing the web, RACOMAT automatically records messages that could eventually be used to test the interface. These recorded messages will be displayed in the list view at the bottom of the http assistant window. Drag and drop one or more of the recorded messages to the risk graph.

In the example shown here, we just log in to DVWA, choose SQL Injection from the menu and submit the value “1”. We drag the only the last message to the graph.

Close the http assistant window when done by clicking the cross in the upper right corner of that window.

So far, the risk graph only contains a system model. In order to add risk related information, click on one of the “String” buttons in the input column of some added threat interface instance. A window with a list of Mitre CWE based weaknesses typically related to that kind of input parameters will be shown. If some weakness in the list is not applicable, right click it. Else, drag it into the input field for which it might be relevant. Thereby, a new vulnerability will be added to the risk graph.

In the example, we drag the SQL injection weakness to the input field “ID”.

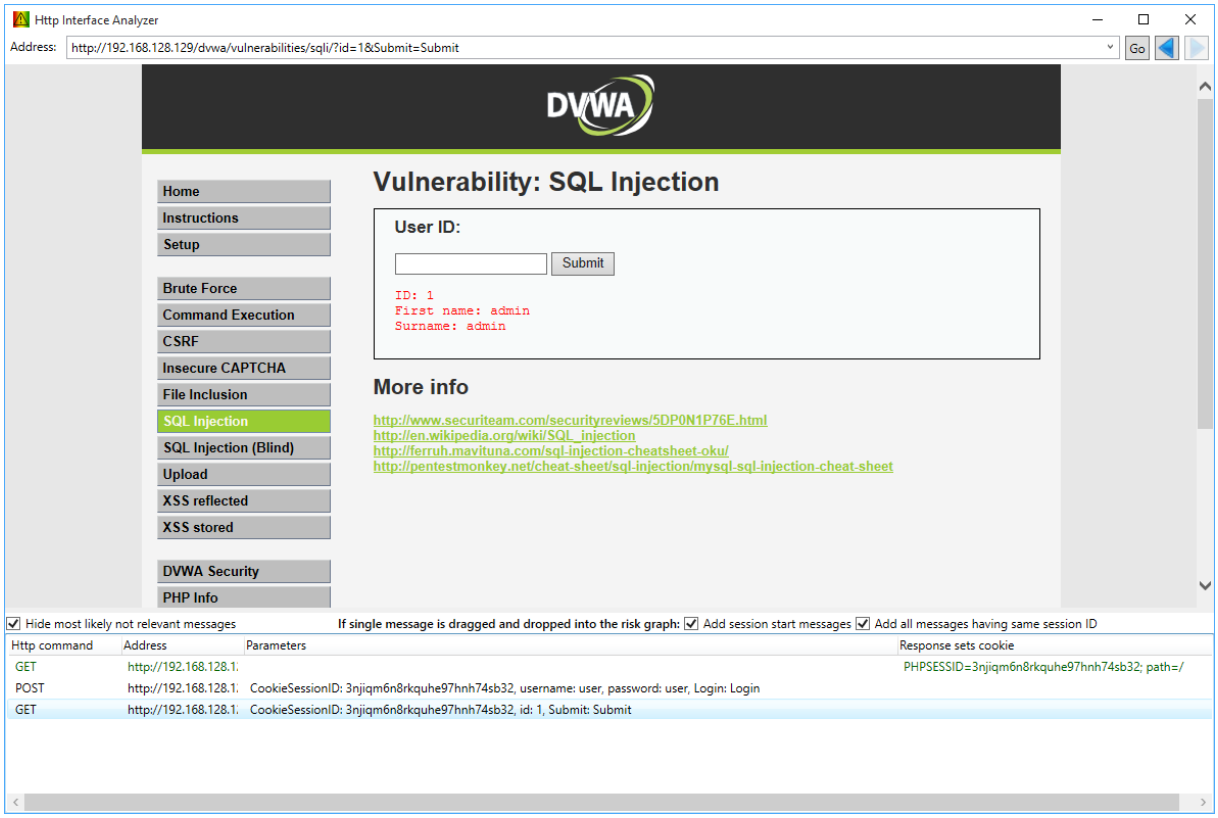

Figure 2 – The http assistant

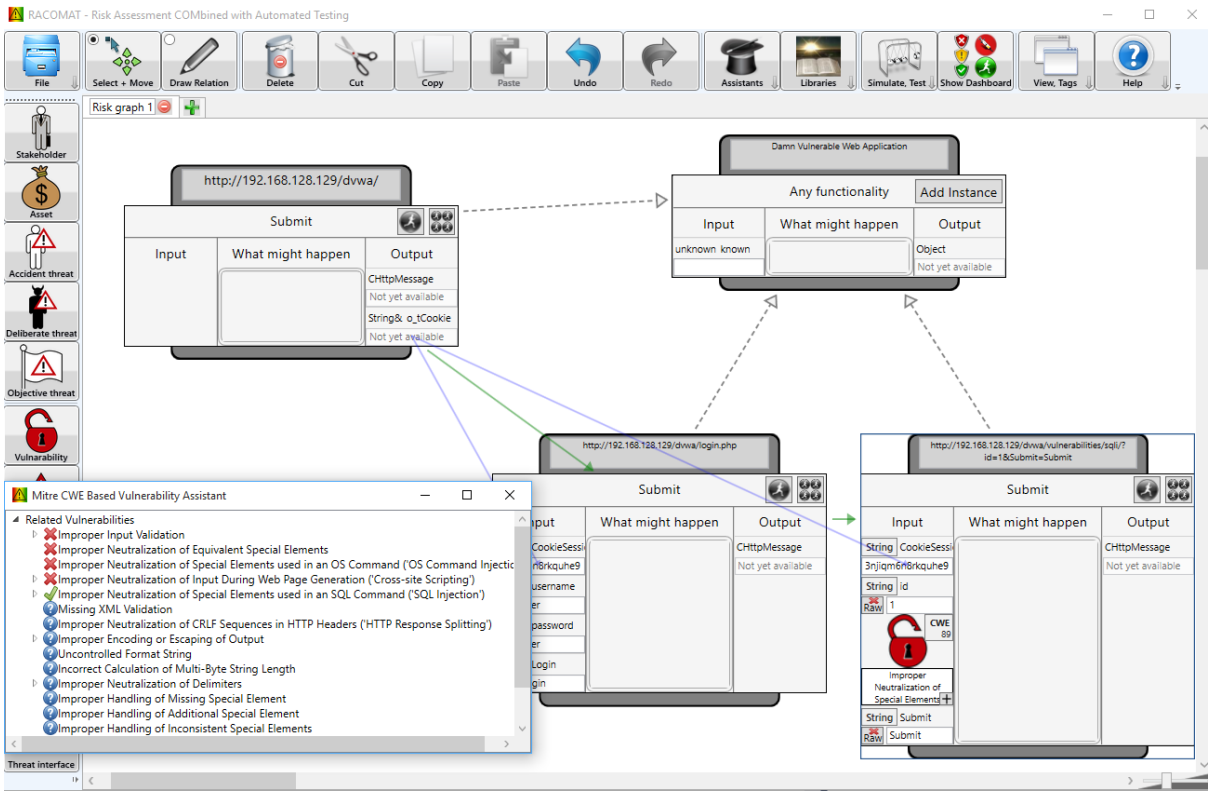

Figure 3 – Adding risk information with the Mitre CWE based assistant

Close the Mitre CWA assistant and click on one of the “CWE” button of some vulnerability. In the menu that opens choose CAPEC based threat scenarios and drag any relevant threat scenarios to the “What might happen” column of the threat interface. This will add threat scenarios to the risk graph and create

relations to the weaknesses from which the threat scenarios were created. In the example we add the attack pattern “SQL Injection”.

The threat scenarios have a “CAPEC” button. Clicking such a “CAPEC” button opens another menu, which can be used to add unwanted incidents. Add one or more of these unwanted incidents by dragging them to the risk graph, e.g. to some field in the output column of the threat scenario. In the example we select the unwanted incident “SQL injection most likely occurred”, which is actually defined by some test pattern.

Each vulnerability, threat scenario and unwanted incident has a small button with a plus sign in its lower right corner. Clicking on that button will show more detailed information. To start security testing, open the detail area for some threat scenario by clicking on its “+” button. From the combo box on the very top of that detail area, it is possible to choose a test pattern that should be applied. If there is already at least one fitting test pattern existing for that threat scenario and associated with the CAPEC ID, then by default some test pattern will already be selected. Otherwise, a test pattern has to be selected manually. Eventually, even a new test pattern has to be created. In the example, the Basic SQL Injection test pattern is already pre-selected. So there is nothing that needs to be done manually.

Just hit the execute button (Figure 4) in order to start the testing process.

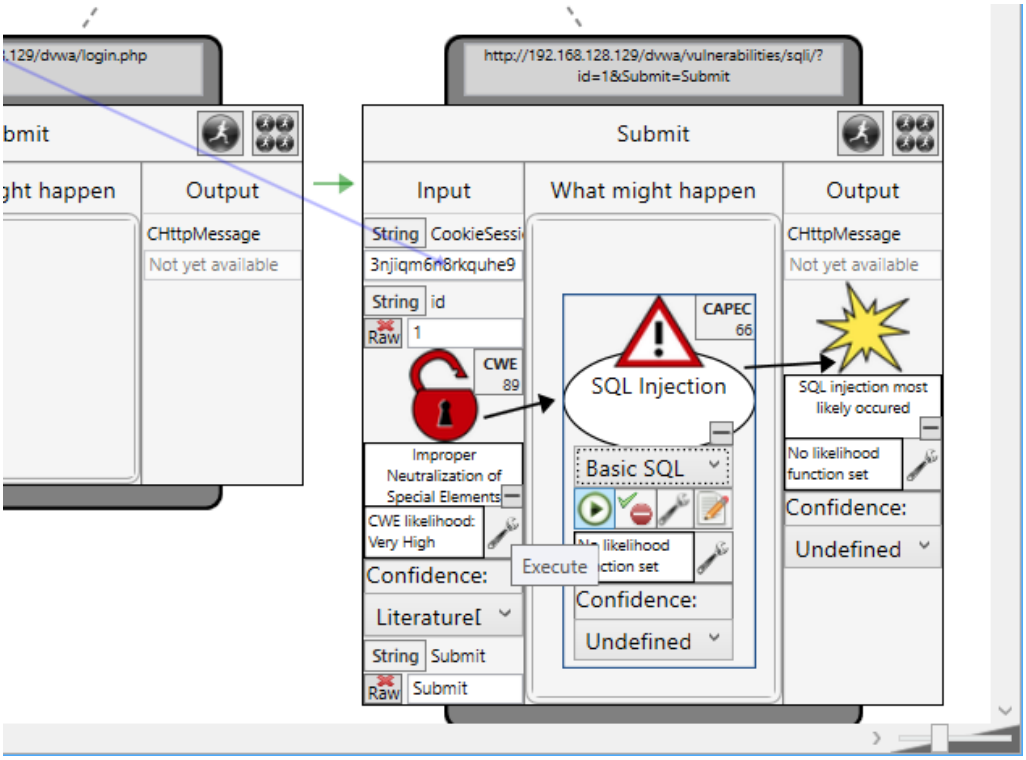

Figure 4 – Executing tests

The test results will be shown in a separate window. Hit the “Calc likelihoods” button to update likelihood values in the risk graph. If there are any unexpected incidents detected, then these should be dragged to the risk graph.

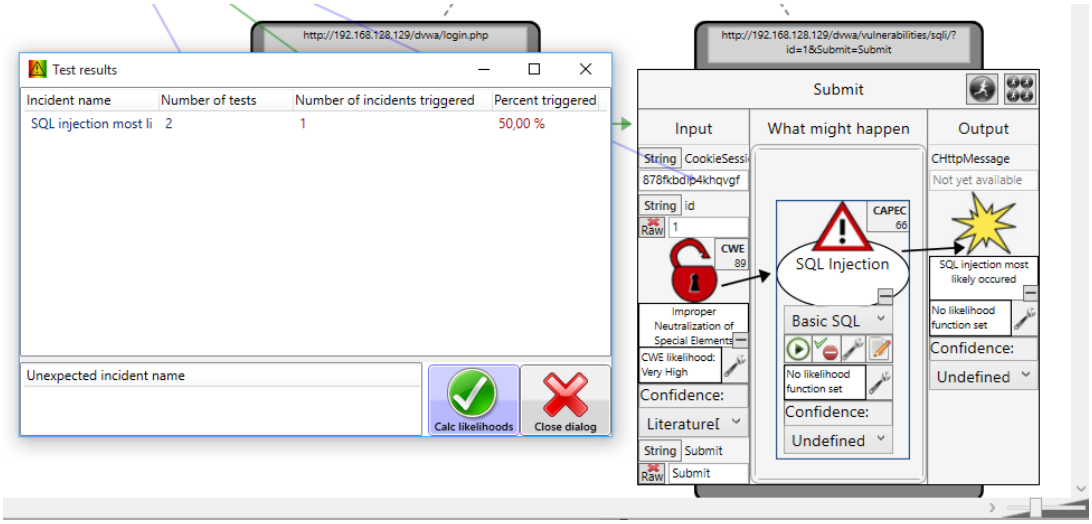

Figure 5–Interpreting test results in the terms of risk assessment