Rapid Security Framework (RSF)

Taxonomie, Bewertungsparameter und Marktanalyse zur Auswahl von Fuzzing-Tools
Motivation

Attacks
- Industrial espionage
- Sabotage
- Data privacy problems

Software is unsecure
bugs: especially **security related bugs**!

→ Detected and **undetected** – (less-than-) **Zero Day** – Vulnerabilities
→ Vulnerability detection := **Secure Software + Costreduction**
Known and unknown vulnerabilities

Intended application

Functional errors

Known

Unknown

Actual application

Vulnerabilities
Methodical Penetration Testing: RiDA

- **Reactive**
  - (Network) Forensics
  - Penetration Testing

- **Proactive**
  - Rapid in-Depth Analysis (RiDA)
    - Threat Modeling
    - Static Analysis
    - Fuzzing

- **Creative Methodical**
  - Damage Potential
    - White Risks (known)
    - Black Risks (unknown)
Standard software development

- Requirements
- Design
- Implementation
- Verification

**Functional Error Correction**

Zero-Day Vulnerabilities

- Design
- Code
- Runtime
Secure software development

- Requirements
- Design
- Implementation
- Verification

Functional Error Correction

Zero-Day Vulnerabilities

- Design
- Code
- Runtime

Threat Modeling
- Static Analysis
- Fuzzing

- Continuous Tool Identification & Evaluation
- >100 Evaluation Criteria
- >300 Tools (> 200k EUR)
Benefit: cost of bug elimination

Costs

Phase

Part of graph:

- Design: 1x
- Implementation: 6x
- Verification: 15x
- Release: 100x

[NIST 2002]
Threat Modeling techniques

- Data Flow Diagrams
- Attack Trees
- ...
Data Flow Diagrams

- **User**
- **Admin**
- **Authn Engine**
- **Audit Engine**
- **Service**
- **Creds**
- **Data Files**
- **Audit Data**
- **Mnmgt Tool**
- **User Data**
- **Set/Get Creds**
- **Set**
- **Request**
- **Response**
- **Authn Info**
- **Authn Request**
- **Requested File(s)**
- **Audit Write**
- **Audit Read**
- **Audit Requests**
- **Audit Info**

[Baier 2006]
Data Flow Diagrams

User

Request

Response

Authn Engine

Get Creds

Authn Info

Requested File(s)

Data Files

Audit Data

Audit Engine

Audit Write

Audit Read

Audit Requests

Audit Info

Company Secrets

Decrypt

Service

Authn Request

Audit Data

Admin

Set User Data

Mnmgt Tool

Verify User Data

Set/Get Creds

Credentials

Set Get Creds

[Baier 2006]
Attack Trees

Threat #1
Decrypt Company Secrets

1.1
Obtain Encrypted File

1.1.1
Bribe the Sysadmin

1.1.2
Hack the Database

1.2
Obtain Password/Key

1.2.1
Brute Force Password/Key

1.2.2
Use Keylogger
Static Analysis Process

Establish Goals
- Enhance Code Understanding
- Find Bugs
- Find Vulnerabilities
- Identify distinctions to specification

Run Tools
- Static Analyser

Review Code

Make Fixes

Reporting
- Generate Report

Target Applications Source Code
Fuzzing process

Identify Input Interfaces
- e.g. Sniffer

Generate Fuzz
- Send Fuzz

Monitoring:
- Recognize Failures
  - Monitor

Reporting
- Generate Report

Vulnerabilities Analysis
- Identification & Analysis

Target Application

State: {Fail, Working}
> 300 Fuzzer

Which are the right for your application(s)?
Rapid in-Depth Security Framework (RSF) activities

1. Tool identification: search strategy- keywords
2. Screening: Taxonomies
3. Tool evaluation: objective, customisable evaluation criteria
Fuzzer taxonomy
classified by purpose
Fuzzer taxonomy
classified by local interface
Fuzzer taxonomy
classified by remote interface

Remote

Network-Protocol

802.11x
UDP
TCP
HTTP(S)
SIP
H.323
[...]

WiFi

Web-Application

VoIP
Market analysis Fuzzer: protocol specific

Supported interfaces

- Network-Protocol: 34%
- Web-Application: 25%
- File-Format: 15%
- Web-Browser: 10%
- API: 7%
- Database: 4%
- Command-Line: 2%
- Environment-Variable: 2%
- In-Memory: 1%

Market analysis Fuzzer: protocol specific

Supported interfaces

- Network-Protocol: 34%
- Web-Application: 25%
- File-Format: 15%
- Web-Browser: 10%
- API: 7%
- Database: 4%
- Command-Line: 2%
- Environment-Variable: 2%
- In-Memory: 1%

© softScheck
One-off Tools: Protocol Specific Fuzzer
# Evaluation criteria & description parameter

<table>
<thead>
<tr>
<th>Product description</th>
<th>Feature comprehensiveness</th>
<th>Documentation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product name Version Release date Developer Distribution Source Available languages Supported Operating Systems Protocol modeling Fuzz-data generation Intended use Interfaces</td>
<td>Feature comprehensiveness</td>
<td>Documentation</td>
</tr>
<tr>
<td>Target monitoring Resetting of the target Exception analysis Reporting Ability to parameterize Error reproduction Scheduled tasks Support for parallel fuzzing Ability to interrupt fuzzing Software ergonomics Effectively Efficiency Functional criteria Dialog criteria In- and output criteria</td>
<td>License type and name Costs for license Costs for Updates Costs for support Additional paid features Costs for purchase of additional components</td>
<td>User manual Technical documentation Quality of integrated help system Quality of the FAQ Quality of video documentation Quality of Tutorials and How-to’s Quality of public support forum or similar system Quality of customer support Volume of 3rd party documentation Case studies and Whitepapers Webinars / Training</td>
</tr>
<tr>
<td>Costs and license</td>
<td>Development status Development activity Interfaces for further development Developer tools Used programming language</td>
<td></td>
</tr>
</tbody>
</table>

© softScheck
Tool-composition: 1 Tool

Case Study: Fuzzing Frameworks

Fuzzing Framework criteria

Model-Inference and Dynamic Generation-Based fuzzing criteria

Generation-Based fuzzing criteria

Mutation-Based and Model-Inference fuzzing criteria

Features

Software ergonomics

Documentation

Extendibility

Costs and License

General
Tool-composition: many Tools

Case study: Fuzzing Frameworks
Test automation
SANS TOP 25 Vulnerabilities

1. Cross-Site Scripting (XSS)
2. SQL Injection
3. Buffer Overflow
7. Path Traversal
8. Unrestricted Upload of Dangerous File Type
9. OS Command Injection
11. Hardcoded Credentials
12. Buffer Access with Incorrect Length Value
13. PHP File Inclusion
14. Improper Validation of Array Index
15. Improper Check for Unusual or Exceptional Conditions
18. Incorrect Calculation of Buffer Size (AW)
19. Missing Authentication for Critical Function
21. Incorrect Permission Assignment for Critical Response
22. Allocation of Resources Without Limits or Throttling
23. Open Redirect
24. Use of a Broken or Risky Cryptographic Algorithm
25. Race Conditions

→ 80% automated identifiable: Blackbox + Whitebox!
→ 20% manuel identifiable: e.g. ‘10. Missing Encryption of Sensitive Data’
Finding all the vulnerabilities: RiDA
Technology- and tool-composition

- Threat Modeling
- Static Analysis
- Fuzzing

# Tools
- # identified vulnerabilities

© softScheck
Future prospects

Rapid in-Depth Analysis Security Suite (RIASS)

- Technology integration: Threat Modeling, Static Analysis, Fuzzing
- Tool integration: up to 300 Tools
- People integration: whole development & product lifecycle

→ Software development industry
→ Consulting industry: QA, (security)testing industry, Compliance
Prof. Dr. Hartmut Pohl
Hartmut.Pohl@softSScheck.com  +49 (2241) 9558 - 881

Peter Sakal, B.Sc. M.Sc.
Peter.Sakal@softSScheck.com  +49 (2241) 9558 - 882

www.softSScheck.com