Software assurance with OpenSAMM

Jacco van Tuijl

jacco@owasp.org
Speaker BIO

• Hack in the Box - Core Crew NL
• Red team tester / Pen tester / security consultant for 7 year
• Software engineering background
• Software security architect @ RES Software
Why a software assurance program?

- Preventing security issues from occurring
- Finding security issues in early stage of development is much cheaper than after release
- Less vulnerabilities in software releases
- Better prepared for when security issues occur
- Keeps your product out of the “Hall of shame”
- Customer demand
Traditional security testing

• A team of developers can make more vulnerabilities in a day than a tester can find in a day

• A tester can find more vulnerabilities in a day than that a team of developers can fix in a month

• Results in an ever-expanding list of known vulnerabilities
OpenSAMM

- OpenSAMM v1.0 released 2009
- OpenSAMM v1.1 (2016 = current)

Work in progress:
- OpenSAMM v1.2 & v2
  - More tools and materials
  - Implementation guidance dev ops & agile
  - Privacy?
Other methodologies

- OWASP CLASP - obsolete
- BSIMM - Proprietary Cigital fork of OpenSAMM alfa
- MS SDL
- SAMATE - Software Assurance Metrics And Tool Evaluation (NIST)
- SSE-CMM
- Grip op SSD – CIP (Dutch government requirement)
# Microsoft SDL

<table>
<thead>
<tr>
<th>Training</th>
<th>Requirements</th>
<th>Design</th>
<th>Implementation</th>
<th>Verification</th>
<th>Release</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>Security</td>
<td>Design</td>
<td>Tools</td>
<td>Dynamic</td>
<td>Incident Response Plan</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Requirements</td>
<td></td>
<td>Analysis</td>
<td>Analysis</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Quality</td>
<td>Attack</td>
<td>Unsafe</td>
<td>Fuzz Testing</td>
<td>Security Review</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Gates/Bug</td>
<td>Analysis/</td>
<td>Functions</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Bars</td>
<td>Reduction</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4. Perform</td>
<td>Threat</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Security</td>
<td>Modeling</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>and Privacy</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Risk Assessments</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

OWASP
Open Web Application Security Project
<table>
<thead>
<tr>
<th>MS SDL</th>
<th>OpenSAMM</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Core Security Training</td>
<td>Education &amp; guidance</td>
</tr>
<tr>
<td>2. Establish Security Requirements</td>
<td>Security requirements</td>
</tr>
<tr>
<td>3. Create Quality Gates/Bug Bars</td>
<td>Code review and security test baseline</td>
</tr>
<tr>
<td>4. Perform Security and Privacy Risk Assessments</td>
<td>Threat Assessment</td>
</tr>
<tr>
<td>5. Establish Design Requirements</td>
<td>Security Requirements</td>
</tr>
<tr>
<td>6. Perform Attack Surface Analysis/Reduction</td>
<td>Threat assessment (ML1) &amp; Design review (One of the security practices)</td>
</tr>
<tr>
<td>7. Use Threat modeling</td>
<td>Threat assessment (ML1)</td>
</tr>
<tr>
<td>8. Use Approved Tools</td>
<td>Secure architecture (ML1)</td>
</tr>
<tr>
<td>9. Deprecate Unsafe Functions</td>
<td>Code review</td>
</tr>
<tr>
<td>10. Perform Static Analysis</td>
<td>Code review</td>
</tr>
<tr>
<td>11. Perform Dynamic Analysis</td>
<td>Security testing</td>
</tr>
<tr>
<td>12. Perform Fuzz Testing</td>
<td>Security testing</td>
</tr>
<tr>
<td>13. Conduct Attack Surface Review</td>
<td>Design review &amp; security testing</td>
</tr>
<tr>
<td>14. Create an Incident Response Plan</td>
<td>Vulnerability management</td>
</tr>
<tr>
<td>15. Conduct Final Security Review</td>
<td>Verification</td>
</tr>
<tr>
<td>17. Execute Incident Response Plan</td>
<td>Incident response plan &amp; team and vulnerability management</td>
</tr>
</tbody>
</table>
SAMM Business functions

Business Functions

- Governance
- Construction
- Verification
- Operations
12 focus areas

Governance
- Strategy & Metrics
- Education & Guidance
- Policy & Compliance

Construction
- Security Requirements
- Secure Architecture
- Threat Assessment

Verification
- Design Review
- Security Testing
- Implementation Review

Operations
- Environment Hardening
- Operational Enablement
- Issue Management
### 3 maturity levels for each focus area

<table>
<thead>
<tr>
<th>Education &amp; Guidance</th>
<th>...more on page 42</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>EG 1</strong></td>
<td></td>
</tr>
<tr>
<td><strong>EG 2</strong></td>
<td></td>
</tr>
<tr>
<td><strong>EG 3</strong></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Objective</strong></th>
<th><strong>Activity</strong></th>
<th><strong>Activity</strong></th>
<th><strong>Activity</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Offer development staff access to resources around the topics of secure programming and deployment</td>
<td>A. Conduct technical security awareness training</td>
<td>A. Conduct role-specific application security training</td>
<td>A. Create formal application security support portal</td>
</tr>
<tr>
<td></td>
<td>B. Build and maintain technical guidelines</td>
<td>B. Utilize security coaches to enhance project teams</td>
<td>B. Establish role-based examination/certification</td>
</tr>
<tr>
<td>Educate all personnel in the software life-cycle with role-specific guidance on secure development</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mandate comprehensive security training and certify personnel for baseline knowledge</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Baseline Assessment

To see what is already being done:

<table>
<thead>
<tr>
<th>Functions</th>
<th>Security Practices</th>
<th>Current</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Governance</td>
<td>Strategy &amp; Metrics</td>
<td>0+</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Governance</td>
<td>Policy &amp; Compliance</td>
<td>0+</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Governance</td>
<td>Education &amp; Guidance</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Construction</td>
<td>Threat Assessment</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Construction</td>
<td>Security Requirements</td>
<td>0+</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Construction</td>
<td>Secure Architecture</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Verification</td>
<td>Design Analysis</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Verification</td>
<td>Implementation Review</td>
<td>0+</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Verification</td>
<td>Security Testing</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Operations</td>
<td>Issue Management</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Operations</td>
<td>Environment Hardening</td>
<td>0+</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Operations</td>
<td>Operational Enablement</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Interview Details

- **Organization**: AppSec Demo
- **Project**: Mobile App
- **Interview Date**: 9-19-2016
- **Interviewer**: Jacco van Tuijl
- **Persons Interviewed**: Henk de Vries - CTO

---

**OWASP**

Open Web Application Security Project
Ready made roadmaps

Roadmap templates for:
• Independent software vendor
• Online service provider
• Financial services organization
• Government organization
Roadmap
Governance

Strategy & Metrics

- Baseline assessment
- SSDLC Roadmap
- Application risk profile
- Register security spend

Education & Guidance
Application Risk profile

• Classify each Application based on financial impact of worst-case scenario
  – Critical: the end of the organization
  – High: big losses
  – Medium: medium losses
  – Low: almost no impact

Quality Gates based on risk: education, compliance, design review, implementation review and security test
Governance

Strategy & Metrics

Policy & Compliance
• Identify external compliance drivers
• Monitor changes
• Checklist and audit
• Release gates
Identify compliancy, regulations and standards

- Law & Regulation
  - US (SOx, HIPAA, Technology Management Reform Act, Security Act)
  - EU (ECHR)
  - International
  - Canada (PIPEDA)
- Contracts & licenses
  - Customer contracts / EULA / bewerkers overeenkomst
  - Partner contracts
  - 3th party components
  - Suppliers contracts
- Company goals and values
- Industry standards
  - PCI-DSS
  - FIPS
  - ISO 27001, ISO 27035
  - OpenSAMM, MS SDL, BSIMM
  - CIP – Grip op SSD
  - Common Criteria for Information Technology Security Evaluation
  - OWASP Application Security Verification Standard
  - CMMI
  - OWASP top 10
  - SANS top 20
Governance

- Strategy & Metrics
- Policy & Compliance
- Education & Guidance
  - High-over security training
  - Role-based training
  - Role-based examination & certification
Maturity level 1: High-over training

SSDLC & Security Awareness

- Microsoft Security Development Lifecycle Core Training classes
  - Introduction to Security Development Lifecycle
  - Basics of Secure Design, Development & Test
  - Introduction to Threat Modeling
  - Privacy in Software Development

- OWASP TOP 10
Maturity level 2: Role specific training

<table>
<thead>
<tr>
<th>Role</th>
<th>Training and/or workshop</th>
</tr>
</thead>
<tbody>
<tr>
<td>Architect</td>
<td>Security principles &amp; threat modelling</td>
</tr>
<tr>
<td>Developer</td>
<td>Secure programming</td>
</tr>
<tr>
<td>Tester</td>
<td>Security testing</td>
</tr>
<tr>
<td>Requirements Engineer</td>
<td>Abuse-cases &amp; Security requirements</td>
</tr>
</tbody>
</table>
How & where do we get security requirements?

- Customer agreements
- Compliance / industry standards
- Access control matrix
- Misuse-cases / abuser stories
- Threat model
- Security testing
- Security practices
### Security Requirements

#### Access control matrix

<table>
<thead>
<tr>
<th>Feature: Service store runbook</th>
<th>Create</th>
<th>Modify</th>
<th>Execute</th>
<th>Read</th>
<th>Delete</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unauthenticated users</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Authenticated user</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Administrators</td>
<td>Yes</td>
<td>Dynamic</td>
<td>Dynamic</td>
<td>Dynamic</td>
<td>Dynamic</td>
</tr>
</tbody>
</table>
Threat modeling

• Microsoft Threat Modeling Tool 2016
  – Spoofing
  – Tampering
  – Repudiation
  – Information disclosure
  – Denial of service
  – Elevation of privilege
### Threat Modeling

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
<th>Justification</th>
<th>Interaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tampering</td>
<td>SQL injection is an attack in which malicious code is inserted into strings that are later passed to an instance of SQL Server for processing.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Denial Of Service</td>
<td>Does Web Service or SQL Database take explicit steps to control resource consumption? Resource consumption attacks can be prevented.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spoofing</td>
<td>SQL Database may be spoofed by an attacker and this may lead to incorrect data delivered to Web Service. Consider using a secure connection.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Information Disclosure</td>
<td>Improper data protection of SQL Database can allow an attacker to read information not intended for disclosure. Review authorization policies.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tampering</td>
<td>If Web Application is given access to memory, such as shared memory or pointers, or is given the ability to control what Web Server does then it is vulnerable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elevation Of Privilege</td>
<td>Web Service may be able to impersonate the context of Web Application in order to gain additional privilege.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tampering</td>
<td>If Web Application is given access to memory, such as shared memory or pointers, or is given the ability to control what Web Server does then it is vulnerable.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elevation Of Privilege</td>
<td>Web Service may be able to impersonate the context of Web Application in order to gain additional privilege.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spoofing</td>
<td>OAuth Provider may be spoofed by an attacker and this may lead to data being sent to the attacker's target instead of the intended recipient.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Repudiation</td>
<td>OAuth Provider claims that it did not receive data from a process on the other side of the trust boundary. Consider using logging to detect this.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Denial Of Service</td>
<td>An external agent interrupts data flowing across a trust boundary in either direction.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elevation Of Privilege</td>
<td>Common SSO implementations such as OAuth2 and OAuth3 are vulnerable to MitM attacks.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Spoofing</td>
<td>OAuth Provider may be spoofed by an attacker and this may lead to unauthorized access to Web Service. Consider using a secure connection.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Repudation</td>
<td>Web Service claims that it did not receive data from a source outside the trust boundary. Consider using logging to detect this.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Denial Of Service</td>
<td>Web Service crashes, hails, stops or runs slowly; in all cases violating an availability metric.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Denial Of Service</td>
<td>An external agent interrupts data flowing across a trust boundary in either direction.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elevation Of Privilege</td>
<td>Web Service may be able to impersonate the context of OAuth Provider in order to gain additional privilege.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elevation Of Privilege</td>
<td>OAuth Provider may be able to remotely execute code for Web Service.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**40 Threats Displayed, 40 Total**

**ID:** 23  **Diagram:** Diagram 1  **Status:** Not Started  **Last Modified:**

**Title:** Spoofing of Source Data Store SQL Database
Threat modeling

• Classify control priority: High, medium or low

Quality gate example:
• All high risk controls in high or critical risk applications should be code reviewed.
• Existence of all controls in high or critical risk applications should be validated.
• The working of all medium and high risk controls should be tested.
Misuse and Abuse-cases

- Negative testing / unhappy flow / abuser story

Diagram:
- User
  - Enter username and password
    - Includes
    - User authentication
      - Includes
      - Show generic error message
        - Includes
        - Lock account after N failed login attempts
          - Includes
          - Validate password minimum length and complexity
            - Includes

- Application / Server
  - Brute force authentication
    - Includes
    - Harvest (guess) valid user accounts
      - Includes
      - Dictionary attacks
        - Includes

- Hacker / Malicious User
Construction

Security Requirements

Threat Assessment

Secure Architecture

- Review architecture for security principles
- List of recommended technologies
- Validate usage of recommended technologies
Secure Architecture

Security principles

• Attack surface reduction
• Defense in depth
• Least privilege
• Secure defaults
  • Securing the weakest link
  • Simplicity in design
  • Fail securely
  • Avoid security by obscurity
  • Detect intrusions and log attacks
  • Don’t trust infrastructure/services/people
  • Input Validation & output encoding
  • Avoid single points of failure
  • Data in transit & rest protection
  • Data loss prevention
  • Audit trail
  • Promote Privacy
  • Never assume that your secrets are safe
  • Complete Mediation
  • Psychological acceptability (security VS usability)
Secure Architecture

Defense in depth

Defense in Depth Layers

- Data
- Application
- Host
- Internal Network
- Perimeter
- Physical
- Policies, Procedures, Awareness
Defense in depth examples

- WAF + Urlscan + Input validation + Parameterized queries + data at rest encryption + output encoding
- Network firewall + IDS + Host based firewall
- Email antivirus and spam filter + strip possible harmful file formats + Host based anti-virus
- HTTPS over IPSEC over a private network
Least privilege Windows processes

1. Local Service (best)
2. Network Service
3. Unique user account
4. Local System
5. Local administrator account
6. Domain administrator account (worst)
Verification

Design Review
- Identify software attack surface
- Analyze design against security requirements
- Release gates

Security Testing
Attack surface analysis

• Look at all of your entry points: Channels, Methods and data
  – Network i/o
  – File i/o
  – Process i/o

• Rank them
  – Authenticated vs Anonymous
  – Administrator only vs regular user
  – Network vs local
  – UDP vs TCP
Also look at sub-features

- File formats
  - Image: JPG, FLA, BMP, PNG or GIF
  - Data: csv, excel or SQL
- HTTP verbs
  - GET, POST, PUT and DELETE
- SMTP
  - Helo, EHLO, MAIL, RCPT, VRFY and EXPN
- HTTPS
  - SSLv1, SSLv2, SSLv3, TLS1.0, TLS1.1 and TLS1.2
## Service Information

### Running Processes

<table>
<thead>
<tr>
<th>New</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>120</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Image Name (PID)</th>
<th>Command Line</th>
<th>Account</th>
<th>Process Flags</th>
</tr>
</thead>
<tbody>
<tr>
<td>splwow64.exe (7836)</td>
<td>C:\Windows\splwow64.exe</td>
<td></td>
<td>(Linker Version: 9.0.1) (ASLR)</td>
</tr>
</tbody>
</table>

## Network Information

### Ports

<table>
<thead>
<tr>
<th>Type</th>
<th>TCP</th>
<th>UDP</th>
</tr>
</thead>
<tbody>
<tr>
<td>All New Ports (142 total)</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>Running as System</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Running as Local Service</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Running as Network Service</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Running as Other</td>
<td>10</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Port Name</th>
<th>State</th>
<th>Process</th>
<th>Account</th>
</tr>
</thead>
<tbody>
<tr>
<td>53367/TCP -- Unknown Protocol</td>
<td>Established</td>
<td>Ssms.exe (PID 6352)</td>
<td></td>
</tr>
<tr>
<td>53399/TCP -- Unknown Protocol</td>
<td>TimeWait</td>
<td>(PID )</td>
<td></td>
</tr>
<tr>
<td>53400/TCP -- Unknown Protocol</td>
<td>TimeWait</td>
<td>(PID )</td>
<td></td>
</tr>
<tr>
<td>53401/TCP -- Unknown Protocol</td>
<td>TimeWait</td>
<td>(PID )</td>
<td></td>
</tr>
<tr>
<td>53402/TCP -- Unknown Protocol</td>
<td>Established</td>
<td>System (PID 4)</td>
<td></td>
</tr>
<tr>
<td>53403/TCP -- Unknown Protocol</td>
<td>TimeWait</td>
<td>(PID )</td>
<td></td>
</tr>
<tr>
<td>53407/TCP -- Unknown Protocol</td>
<td>TimeWait</td>
<td>(PID )</td>
<td></td>
</tr>
</tbody>
</table>
Attack surface reduction examples

• Windows
  – Authenticated RPC
  – Firewall on by default
• SQL Server
  – Xp_cmdshell off by default
  – CLR and COM off by default
• IIS
  – Off by default
  – Static files by default
• Visual Studio
  – Web service listen on localhost only
  – SQL Server Express listen on localhost only
It is not just about turning stuff off

<table>
<thead>
<tr>
<th>Higher Attack Surface</th>
<th>Lower Attack Surface</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execute by default</td>
<td>Off by default</td>
</tr>
<tr>
<td>Open Socket</td>
<td>Closed socket</td>
</tr>
<tr>
<td>UDP</td>
<td>TCP</td>
</tr>
<tr>
<td>Anonymous access</td>
<td>Authenticated access</td>
</tr>
<tr>
<td>Admin access</td>
<td>User access</td>
</tr>
<tr>
<td>Internet access</td>
<td>Local subnet access</td>
</tr>
<tr>
<td>System</td>
<td>Not system</td>
</tr>
<tr>
<td>Uniform defaults</td>
<td>User-chosen settings</td>
</tr>
<tr>
<td>Large code</td>
<td>Small code</td>
</tr>
<tr>
<td>Weak/flexible ACLs</td>
<td>Strong/strict ACLs</td>
</tr>
</tbody>
</table>
Verification

Design Review

Code Review
- Checklist
- Review of high-risk code
- Automated code analysis
- Derive test cases from security requirements
- Conduct penetration testing
- Automated security testing
- Release gates for security testing
OWASP - Application Security verification Standard

- Provides 3 levels of application verification
Deployment

Operational Enablement

- Document procedures for typical application security alerts
- Change management
- Operational security guide
- Secure Operational environment specifications
- Install security updates
- Create security response team
- Incident response process
- Responsible disclosure
- Root cause analysis for incidents
Responsible disclosure

• Responsible disclosure policy
• Facilitate security researchers that want to report security issues (without service contract or legal consequences)
• Prioritize issues
• Security bulletin - mailing list
  – application specific
  – no advertisements
• 60 day max fix time

Responsible disclosure ≠ Full disclosure
Prioritize issues

Common Vulnerability Scoring System Version 3.0 Calculator

Hover over metric group names, metric names and metric values for a summary of the information in the official CVSS v3.0 Specification Document. The Specification is available in the list of links on the left, along with a User Guide providing additional scoring guidance, an Examples document of scored vulnerabilities, and notes on using this calculator (including its design and an XML representation for CVSS v3.0).

<table>
<thead>
<tr>
<th>Base Score</th>
<th>7.1 (High)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Attack Vector (AV)</strong></td>
<td></td>
</tr>
<tr>
<td>Network (N)</td>
<td>Adjacent (A)</td>
</tr>
<tr>
<td><strong>Attack Complexity (AC)</strong></td>
<td></td>
</tr>
<tr>
<td>Low (L)</td>
<td>High (H)</td>
</tr>
<tr>
<td><strong>Privileges Required (PR)</strong></td>
<td></td>
</tr>
<tr>
<td>None (N)</td>
<td>Low (L)</td>
</tr>
<tr>
<td><strong>User Interaction (UI)</strong></td>
<td></td>
</tr>
<tr>
<td>None (N)</td>
<td>Required (R)</td>
</tr>
<tr>
<td><strong>Scope (S)</strong></td>
<td></td>
</tr>
<tr>
<td>Unchanged (U)</td>
<td>Changed (C)</td>
</tr>
<tr>
<td><strong>Confidentiality (C)</strong></td>
<td></td>
</tr>
<tr>
<td>None (N)</td>
<td>Low (L)</td>
</tr>
<tr>
<td><strong>Integrity (I)</strong></td>
<td></td>
</tr>
<tr>
<td>None (N)</td>
<td>Low (L)</td>
</tr>
<tr>
<td><strong>Availability (A)</strong></td>
<td></td>
</tr>
<tr>
<td>None (N)</td>
<td>Low (L)</td>
</tr>
</tbody>
</table>
Responsible disclosure policy

• Clear rules
  – What is allowed and what not
  – What can be expected from the organization

• Bug Bounty program
  – Big reward will get you a lot reports: most false
  – Lot of work to analyze reports
Prioritize issues

• CVSS v3

Issues reported externally or published on the internet should get a higher priority

The higher the application risk rating the higher the priority

60 day fix time is common practice
Release date: 10-06-2016
Vulnerability ID: 16-0132
Severity: Medium
CVE number: N/A

Affected software
Application X version 1.x
Application Y version 2.3 Build 1941 and older

Summary
When using Application X or Application Y in with configuration Z a rare race condition could occur that could result in a temporary bypass of security control Q

Solution
Application X version 1.x
Upgrade Application X version 1.x to version 2.0 or newer
Application Y version 2.3 Build 1941 and older
Upgrade Application Y to version 2.3 build 2133 or newer

Workaround
Limit access to ... using group policy
Privacy

• Privacy impact assessment
  – NIST Privacy Impact Assessments
  – MS Application Privacy Assessment
• Avoid handling PII where possible
• Define where PII will be used for in privacy statement
• Don’t keep PII longer than required
• Data processing agreement
TIPS

• Tools available on the OpenSAMM Wiki
• Use tools & materials from MS SDL
• Join OpenSAMM Mailing list and Monthly call
• Add me on LinkedIn: Jacco van Tuijl