Cyber Security Body of Knowledge: Secure Software Lifecycle
When you use this information under the Open Government Licence, you should include the following attribution: CyBOK Secure Software Lifecycle Knowledge Area Issue 1.0 © Crown Copyright, The National Cyber Security Centre 2019, licensed under the Open Government Licence http://www.nationalarchives.gov.uk/doc/open-government-licence/.

The CyBOK project would like to understand how the CyBOK is being used and its uptake. The project would like organisations using, or intending to use, CyBOK for the purposes of education, training, course development, professional development etc. to contact it at contact@cybok.org to let the project know how they are using CyBOK.
Security Software Lifecycle

The components of a comprehensive software development process to prevent and detect all classes of security defects and to respond in the event of an exploit.
Agenda

• History

• Prescriptive Secure Software Lifecycle Processes
  – Microsoft Security Development Lifecycle (12 practices)
  – Touchpoints (7 practices)
  – SAFECode (8 practices)

• Adaptations of the Secure Software Lifecycle
  – Agile, Mobile, Cloud, Internet of Things (IoT), Road Vehicles, eCommerce/Payment card Industry

• Assessing the Secure Development Lifecycle
  – SAMM
  – BSIMM
  – Common Criteria
History (and Current 😞): Penetrate and Patch

- Security is assessed when “complete”; discovered vulnerabilities then fixed/patched
  - Costly
  - Attackers can find and exploit vulnerabilities without being noticed
  - (Urgent) Patches can introduce new vulnerabilities
  - Patches go unapplied by customers
1998: Gary McGraw DARPA funded project on the application of software engineering to assessment of security vulnerabilities

McGraw (and with John Viega) advocated for proactive and rigorous software analysis to assess and prevent vulnerabilities

- Book: Building Secure Software
2002: Bill Gates announces the Trustworthy Computing Initiative

2004: Turned into a structured process, the SDL (http://microsoft.com/sdl)

- Evolved to Version 5.2 in 2012, Version 6.0 in 2013
- Microsoft offers many (free) tools and templates to support SDL

“Trustworthy Computing is the highest priority for all the work we are doing. We must lead the industry to a whole new level of Trustworthiness in computing.”
Agenda

• History
• Prescriptive Secure Software Lifecycle Processes
  – Microsoft Security Development Lifecycle (12 practices)
  – Touchpoints (7 practices)
  – SAFECode (8 practices)
• Adaptations of the Secure Software Lifecycle
  – Agile, Mobile, Cloud, Internet of Things (IoT), Road Vehicles, eCommece/Payment card Industry
• Assessing the Secure Development Lifecycle
  – SAMM
  – BSIMM
  – Common Criteria
1: Provide Training

Assess organizational knowledge on security and privacy – establish training program as necessary

- Establish training criteria
  - Content covering secure design, development, test and privacy
  - (e.g. 80% of all technical personnel trained)
- Establish minimum training frequency – attackers are a moving target
  - Employees must attend n classes per year
- Need to cover all the topics in the SDL
Consider security at the outset of a project

- Development team identifies security requirements
- Factor in security implications of functional requirements, legal and industry, compliance, standards, known threats, previous security incidents.
- Techniques for systematically developing: examples: SQUARE, abuse cases, i*, and KAOS
3: Define Metrics and Compliance Reporting

- Define and document a bug bar for security
  - Classification of what are moderate, important, or critical security and privacy bug types
  - User to set priority for fixing and to determine if the product can ship
- Ensure that bug reporting tools can track security and issues and that a database can be queried dynamically for all security bugs
- Understand requirement compliance and associate reporting requirements
Threat modeling is a process to understand (potential) security threats to a system, determine risks from those threats, and establish appropriate mitigations.
• Consider design principles, such as:
  – Economy of mechanism
  – Fail-safe defaults
  – Complete mediation
  – Separation of privilege
  – Least privilege
  – Least common mechanism
  – Secure the weakest link
  – Defense in depth
  – Give or earn but never assume trust
  – Always consider the users
Through the proper use of cryptography, one can protect the confidentiality of data, protect data from unauthorized modification, and authenticate the source of data.
7: Manage the security risk of using 3rd party components

Open Source Vulnerabilities Published Per Year

53.8% increase in the number of open source application library security vulnerabilities published in 2016

https://snyk.io/stateofossecurity/
8: Use approved tools

- Specification of approved build tools and options
  - Compilers and code generators
  - Static analysis
  - Debuggers
  - Dynamic analysis
  - Test verification
  - IDE

- Decide on settings and ensure settings are correct

- Security isn’t static
  - Update your tools because they change with dynamic threat environment

http://www.ultraedit.com/support/tutorials_power_tips/uestudio/integrated_debugger.html
9: Perform static analysis security testing (SAST)
10: Perform dynamics analysis security testing (DAST)
11: Perform penetration testing
12: Establish a standard incident response practice

Creation of a clearly defined support policy

- Prepare Cyber Security Incident Response Plan (CSIRP)
  - Identify contact for Cyber Security Council and resources to respond to incidents
  - 24x7x365 contact information for engineering, marketing and management individuals with decision-making authority
- Ensure ability to service all code “emergency” releases and all licensed 3rd party code.
Agenda

• History

• Prescriptive Secure Software Lifecycle Processes
  – Microsoft Security Development Lifecycle (12 practices)
  – Touchpoints (7 practices)
  – SAFECode (8 practices)

• Adaptations of the Secure Software Lifecycle
  – Agile, Mobile, Cloud, Internet of Things (IoT), Road Vehicles, eCommece/Payment card Industry

• Assessing the Secure Development Lifecycle
  – SAMM
  – BSIMM
  – Common Criteria
Touchpoints

- Code review (tools)
- Architecture risk
- Penetration Testing
- Risk-based testing
- Abuse cases
- Security requirements
- Security ops

Diagram showing the flow of security processes:

1. Security requirements
2. Abuse cases
3. Risk analysis
4. Risk-based security tests
5. Code review (tools)
6. Penetration testing
7. Risk analysis
8. Security operations

Steps include:

- Requirements and use cases
- Architecture and design
- Test plans
- Code
- Tests and test results
- Feedback from the field

Image of a book cover titled "Building Secure Software".
4. Risk-based security testing

- Code review (tools)
- Architecture risk
- Penetration Testing
- Risk-based testing
- Abuse cases
- Security requirements
- Security ops

Diagramming

Validation

Threat enumeration

Mitigation

SOFTWARE TESTING
Figure 4. Use and misuse cases for Web portal security.

7. Security operations

- Code review (tools)
- Architecture risk
- Penetration Testing
- Risk-based testing
- Abuse cases
- Security requirements
- Security ops

Diagram:
- Security requirements
- External review
- Code review (tools)
- Penetration testing
- Abuse cases
- Security operations

- Requirements and Use Cases
- Architecture and Design
- Test Plans
- Code
- Tests and Test Results
- Feedback from the Field
Agenda

• History
• Prescriptive Secure Software Lifecycle Processes
  – Microsoft Security Development Lifecycle (12 practices)
  – Touchpoints (7 practices)
  – SAFECode (8 practices)
• Adaptations of the Secure Software Lifecycle
  – Agile, Mobile, Cloud, Internet of Things (IoT), Road Vehicles, eCommece/Payment card Industry
• Assessing the Secure Development Lifecycle
  – SAMM
  – BSIMM
  – Common Criteria
SAFECODE

Security control  Design  Secure coding  3rd party  Testing  Security findings  Vulnerability response  Deploy SDL

SAFECODE Charter Members

[Logos of various companies]
3. Secure coding practices

Security control → Design → Secure coding → 3rd party → Testing → Security findings → Vulnerability response → Deploy SDL

**USE OF SECURE CODING STANDARDS**
8. Planning the implementation and deployment of secure development
Agenda

• History

• Prescriptive Secure Software Lifecycle Processes
  – Microsoft Security Development Lifecycle (12 practices)
  – Touchpoints (7 practices)
  – SAFECode (8 practices)

• Adaptations of the Secure Software Lifecycle
  – Agile, Mobile, Cloud, Internet of Things (IoT), Road Vehicles, eCommerce/Payment card Industry

• Assessing the Secure Development Lifecycle
  – SAMM
  – BSIMM
  – Common Criteria
Adaptations

• Agile:
  – SAFECODE provides security user stories
  – Microsoft Secure DevOps
    • Keeping credentials safe
    • Continuous learning and monitoring

• Mobile
  – OWASP resources for secure testing and threat modeling

• Cloud
  – SAFECODE practices for cloud
    • Cloud-based security threats
Adaptations - 2

- **Internet of Things (IoT)**
  - NIST practices: RFID, not allowing default passwords, use of Manufacturer Usage Description (MUD) specifications, secure update process

- **Road Vehicles**
  - US Highway/Traffic Safety Administration guidelines, particularly related to the systems engineering

- **eCommerce/ Payment Card Industry (PCI)**
  - Data Security Standard
Agenda

• **History**

• **Prescriptive Secure Software Lifecycle Processes**
  – Microsoft Security Development Lifecycle (12 practices)
  – Touchpoints (7 practices)
  – SAFECODE (8 practices)

• **Adaptations of the Secure Software Lifecycle**
  – Agile, Mobile, Cloud, Internet of Things (IoT), Road Vehicles, eComerce/Payment card Industry

• **Assessing the Secure Development Lifecycle**
  – SAMM
  – BSIMM
  – Common Criteria
Open SAMM Framework

Software Development

Business Functions:
- Governance
- Construction
- Verification
- Operations

Security Practices:
- Strategy & Metrics
- Education & Guidance
- Security Requirements
- Design Review
- Security Testing
- Environment Hardening
- Policy & Compliance
- Threat Assessment
- Secure Architecture
- Implementation Review
- Issue Management
- Operational Enablement
# BSIMM Framework

<table>
<thead>
<tr>
<th>Governance</th>
<th>Intelligence</th>
<th>SSDL Touchpoints</th>
<th>Deployment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Strategy &amp; Metrics (SM)</td>
<td>Attack Models (AM)</td>
<td>Architecture Analysis (AA)</td>
<td>Penetration Testing (PT)</td>
</tr>
<tr>
<td>Training (T)</td>
<td>Standards &amp; Requirements (SR)</td>
<td>Security Testing (ST)</td>
<td>Configuration Management &amp; Vulnerability Management (CMVM)</td>
</tr>
</tbody>
</table>

**CyBOK**
Objectives of the Common Criteria

- Permits comparability between the results of independent security evaluations by providing common set of requirements for secure functionality of IT products;
- Establishes level of confidence in the security functionality of IT products; and
- May help consumers determine whether the IT product fulfills their security needs

http://www.commoncriteriaportal.org/thecкра.html
CC Assurance* Levels

- EAL1: Functionally Tested
- EAL2: Structurally Tested
- EAL3: Methodically Tested and Checked
- EAL4: Methodically Designed, Tested, and Reviewed
- EAL5: Semiformally Designed and Tested
- EAL6: Semiformally Verified Design and Tested
- EAL7: Formally Verified Design and Tested

*Assurance: grounds for confidence that an entity meets its security objectives - http://www.commoncriteriaportal.org/files/ccfiles/ccpart1v2.3.pdf
Summary

• Important for an organization to have a comprehensive secure software lifecycle

• Organizations usually customize their own secure software lifecycle (rather than take a prescriptive approach)

• Successful adoption requires cultural change (in addition to adopting the technical practices)
Images

- https://online.stanford.edu/courses/xacs101-software-security-foundations
- http://www.greatpriceshere.com/2008/06/30/bill-gates-dethroned/
- https://news.virginia.edu/content/qa-technology-expert-and-uva-grad-gary-mcgraw-talks-cybersecurity
- http://www.webdesigncompany.net/website-optimization-moving-target/
- https://www.ioausa.com/employee-benefits/compliance/
- https://towardsdatascience.com/the-basics-of-cryptography-80c7906ba2f7
- http://www.ultraedit.com/support/tutorials_power_tips/uestudio/integrated_debugger.html
- https://medium.com/@dieswaytoofast/fuzzing-and-deep-learning-5aae84c20303
- https://www.opsfolio.com/risk-center/use-secure-coding-standards/