Your Dynamic Software Security Journey with OWASP SAMM2

Bart De Win

Bart.DeWin@owasp.org

OWASP
Open Web Application
Security Project

Bart?

Bart De Win, Ph.D.

- 20+ years experience in secure software development
- Belgian OWASP chapter co-leader
- OWASP SAMM contributor, evangelist and co-leader
- Author of >60 publications
- Director & security consultant @PwC BE
- Bart.de.win@pwc.com

This training ?

- Goal is to discuss how to apply OWASP SAMM in practice
- Looking into different parts from a practical perspective
- Based on the case of your own company
- Discussing some of the challenges that you might face
- Open interaction session
- SAMM2 – WIP

Timing

9h00 – 10h30: Training

10h30 – 11h00: coffee break

11h00 – 12h30 : Training

12h30 – 13h30: lunch

13h30 – 15h00: Training

15h00 – 15h30: coffee break

15h30 – 17h00: Training

Rules of the House

- Turn off mobile phones
- Interactive training
- Chatham house rules

Preparation

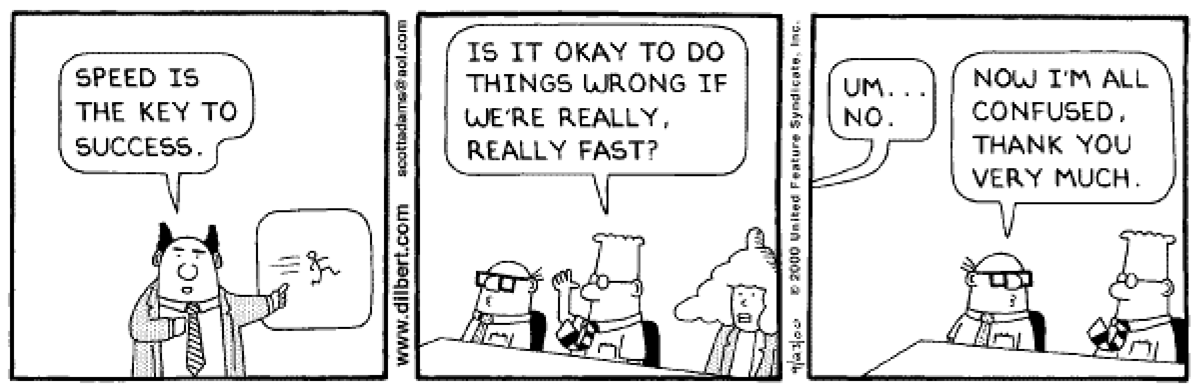
- Get the OWASP SAMM2 Toolbox at:

https://github.com/OWASP/samm/tree/master/Supporting%20Resources/v2.0/toolbox/

Today’s Agenda

1. **Introduction to SDLC and OWASP SAMM**
2. Applying OWASP SAMM
   - Methodology
   - Assessment Governance
   - Assessment Design
   - Assessment Implementation
   - Assessment Verification
   - Assessment Operations
   - Setting Improvement Targets
3. OWASP SAMM Tools
4. OWASP SAMM Best Practices

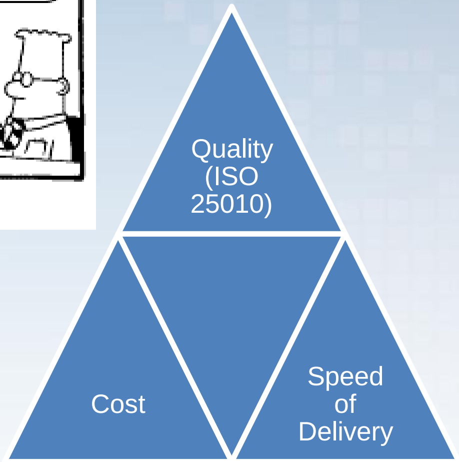
Application Security Problem


SPEED IS THE KEY TO SUCCESS.
www.dilbert.com
scottadams@aol.com
IS IT OKAY TO DO THINGS WRONG IF WE'RE REALLY, REALLY FAST?
© 2000 United Feature Syndicate, Inc.
UM... NO.
NOW I'M ALL CONFUSED. THANK YOU VERY MUCH.

Software complexity

Technology stacks

Requirements?



75% of vulnerabilities are application related

Mobile

Connected

Multi-platform

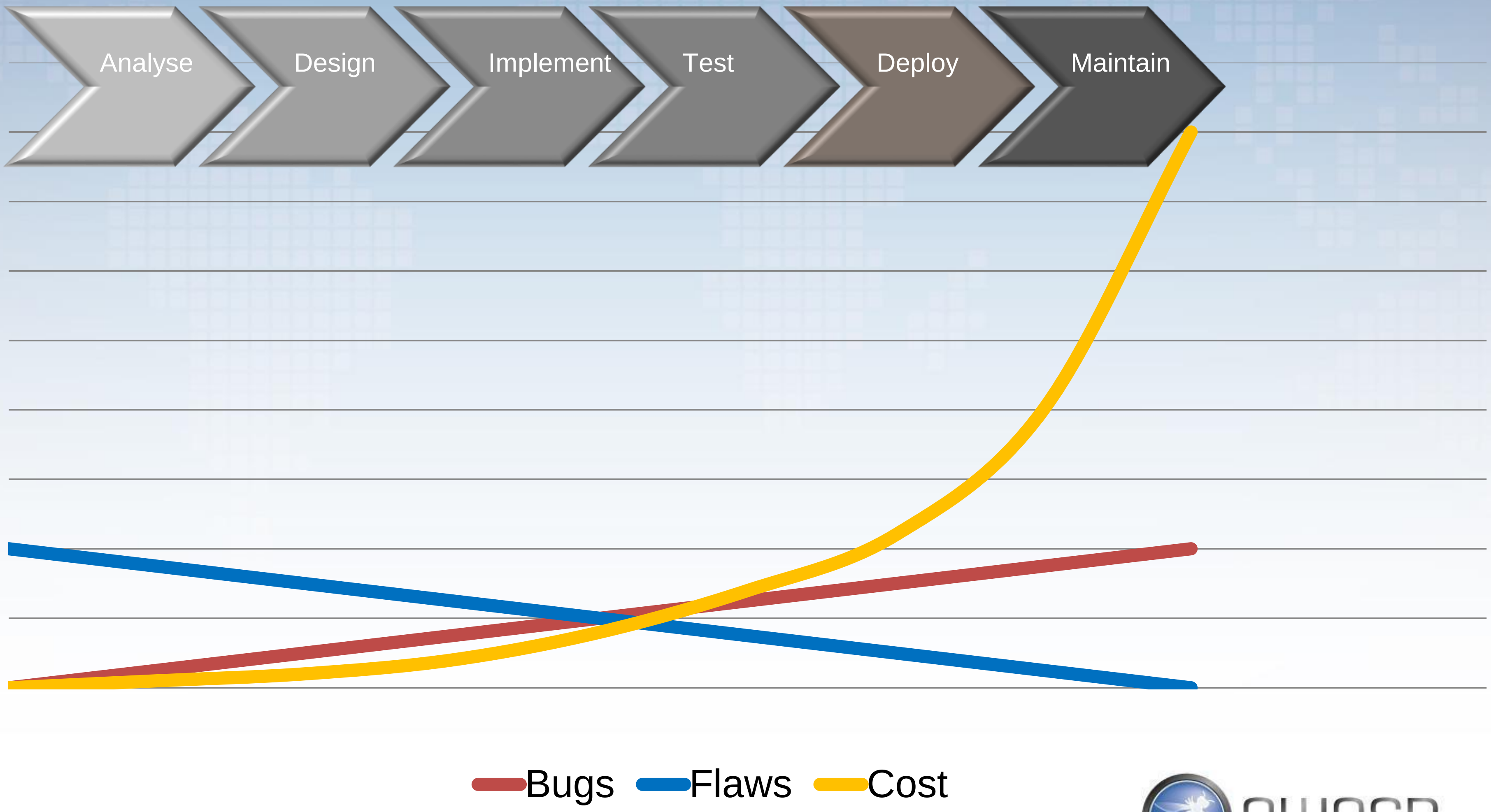
Cloud

Responsive Design

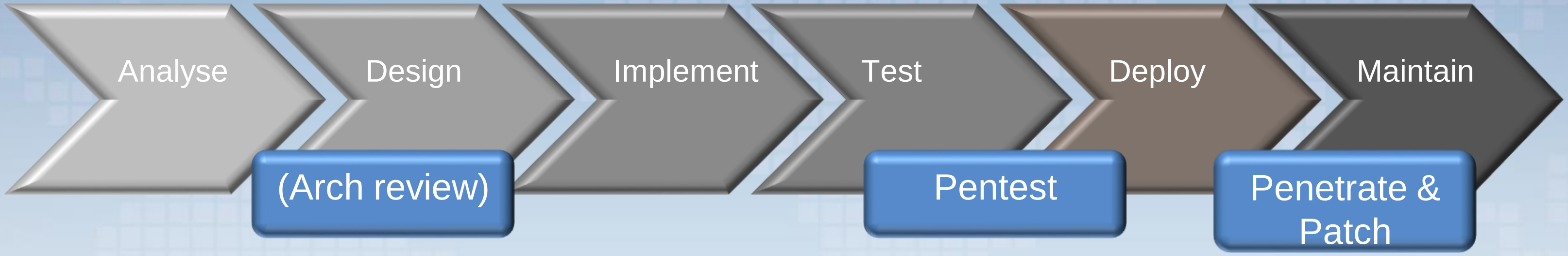
Application Security Symbiosis

Open Web Application Security Project

Application Security during Software Development



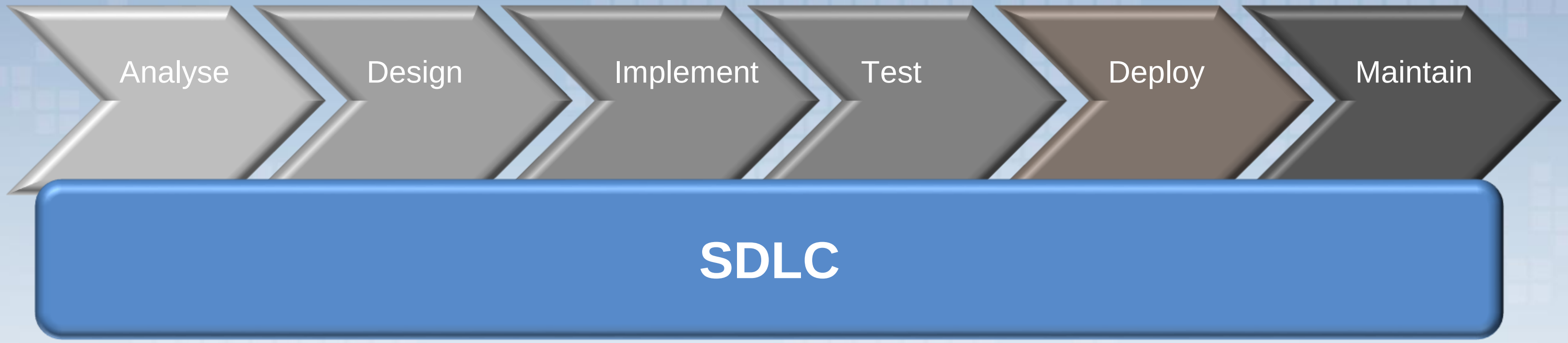
The State-of-Practice in Secure Software Development



Problematic, since:

- Focus on bugs, not flaws
- Penetration can cause major harm
- Not cost efficient
- No security assurance
  - All bugs found ?
  - Bug fix fixes all occurences ? (also future ?)
  - Bug fix might introduce new security vulnerabilities

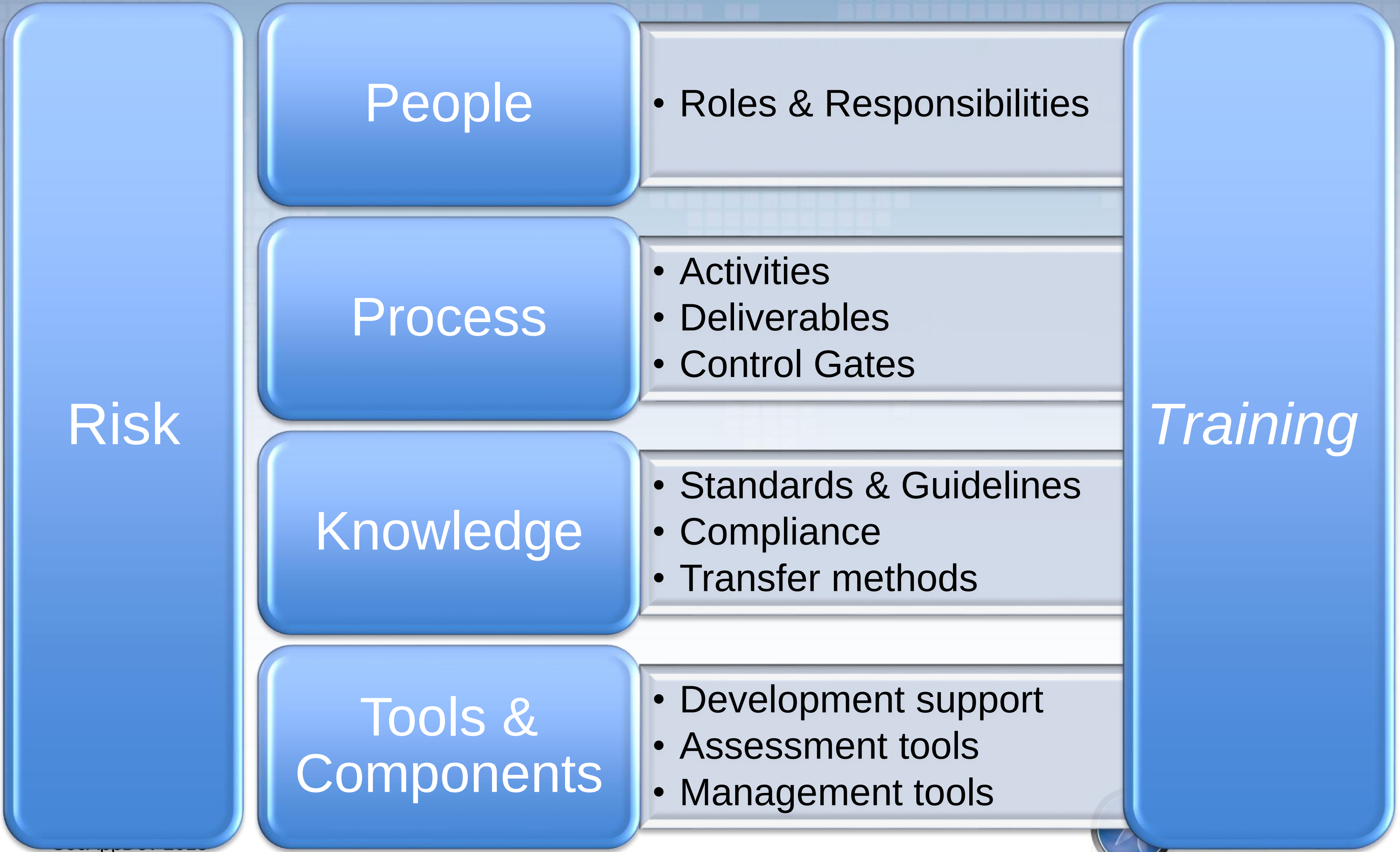
SDLC ?



Enterprise-wide software security improvement program

- Strategic approach to assure software quality
- Goal is to increase systematicity
- Focus on security functionality and security hygiene

SDLC Cornerstones



Risk

People
- Roles & Responsibilities

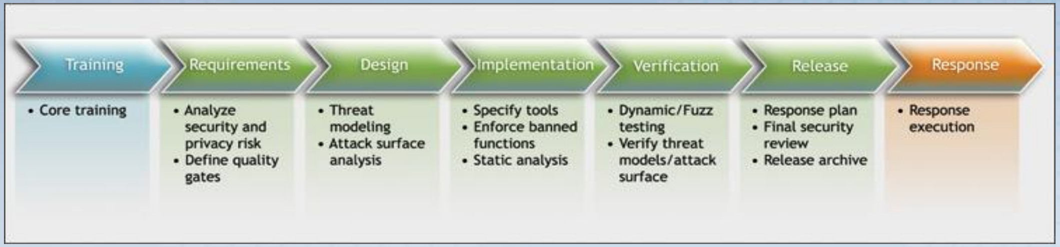
Process
- Activities
- Deliverables
- Control Gates

Knowledge
- Standards & Guidelines
- Compliance
- Transfer methods

Tools & Components
- Development support
- Assessment tools
- Management tools

Training

SDLC-related initiatives



Training
- Core training

Requirements
- Analyze security and privacy risk
- Define quality gates

Design
- Threat modeling
- Attack surface analysis

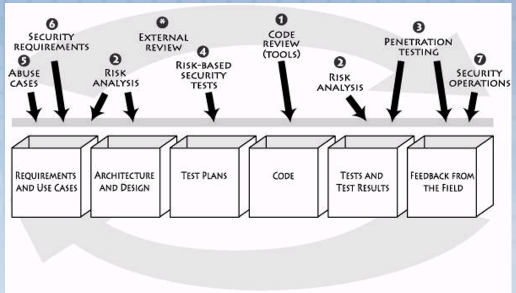
Implementation
- Specify tools
- Enforce banned functions
- Static analysis

Verification
- Dynamic/Fuzz testing
- Verify threat models/attack surface

Release
- Response plan
- Final security review
- Release archive

Response
- Response execution

Microsoft SDL

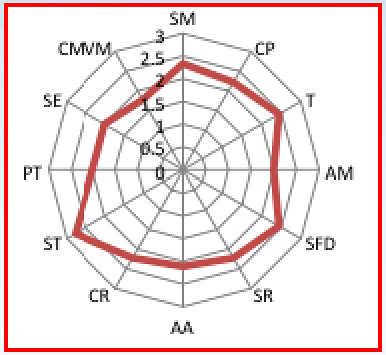


TouchPoints

CLASP

NIST National Institute of Standards and Technology

SP800-64

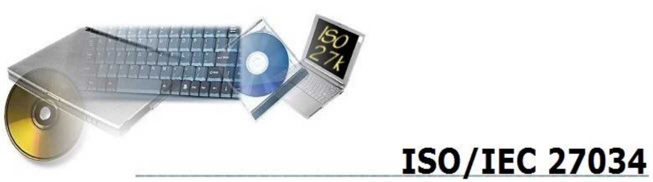


BSIMM

SAFECode Software Assurance Forum for Excellence in Code Driving Security and Integrity

SSE CMM

SSE-CMM



ISO/IEC 27034

Gartner.

Software Engineering Institute | Carnegie Mellon

TSP-Secure

GASSP

Software Assurance Maturity Model

A guide to building security into software development

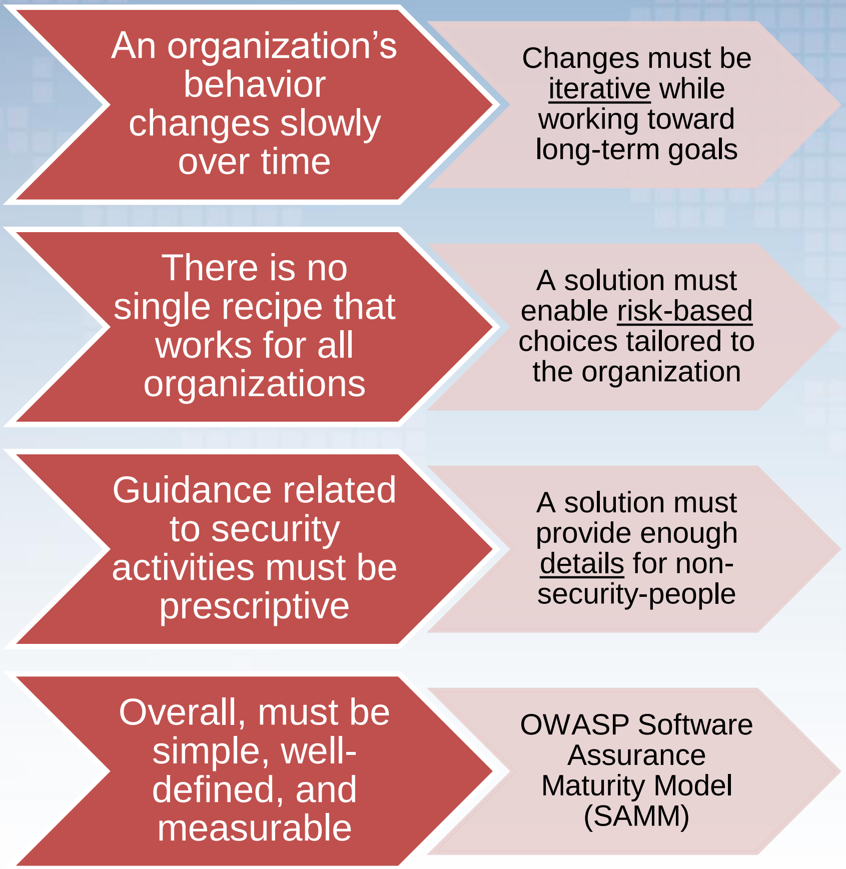
Version - 1.0

SAMM

OWASP Open Web Application Security Project

SAMM Training, OWASP Global AppSec, Amsterdam 2019

Why a Maturity Model ?



An organization's behavior changes slowly over time

Changes must be iterative while working toward long-term goals

There is no single recipe that works for all organizations

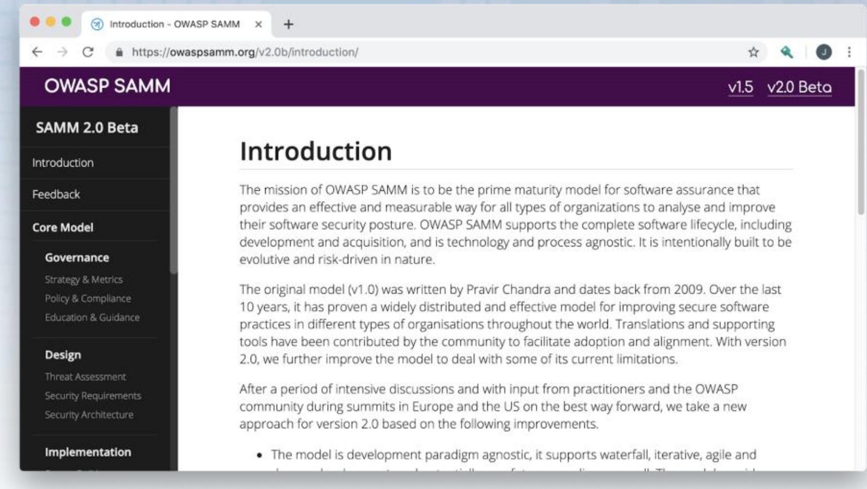
A solution must enable risk-based choices tailored to the organization

Guidance related to security activities must be prescriptive

A solution must provide enough details for non-security-people

Overall, must be simple, well-defined, and measurable

OWASP Software Assurance Maturity Model (SAMM)



Introduction

The mission of OWASP SAMM is to be the prime maturity model for software assurance that provides an effective and measurable way for all types of organizations to analyse and improve their software security posture. OWASP SAMM supports the complete software lifecycle, including development and acquisition, and is technology and process agnostic. It is intentionally built to be evolutive and risk-driven in nature.

The original model (v1.0) was written by Pravir Chandra and dates back from 2009. Over the last 10 years, it has proven a widely distributed and effective model for improving secure software practices in different types of organisations throughout the world. Translations and supporting tools have been contributed by the community to facilitate adoption and alignment. With version 2.0, we further improve the model to deal with some of its current limitations.

After a period of intensive discussions and with input from practitioners and the OWASP community during summits in Europe and the US on the best way forward, we take a new approach for version 2.0 based on the following improvements.

- The model is development paradigm agnostic, it supports waterfall, iterative, agile and

OWASP
Open Web Application Security Project

Key changes in SAMM v2.0

SAMM v1.5

- Four Business Functions - Governance, Construction, Verification, Operations
- 12 Security Practices
- Very little, if any, prescriptive guidance for build and deploy domains

- Maturity level activities could be orphaned, and sometimes unrelated to each other
- Maturity level activities not in order of increasing difficulty, cost of implementation

- Coverage based measurement

SAMM v2.0

- Five Business Functions - Governance, **Design, Implementation,** Verification, Operations
- 15 Security Practices
- New Business Function "Implementation" to accommodate guidance related to build and deploy domains

- Maturity level activities are aligned and linked per Stream. Each **stream** has a clear Objective
- Maturity level activities designed in order of increasing difficulty, implementation cost

- **Coverage & Quality** based measurement

- Also includes supporting infrastructure

OWASP SAMM vs. BSIMM

- Prescriptive vs. Descriptive
- Open vs. Closed
- Low Watermark vs. High Watermark

OWASP SAMM 101

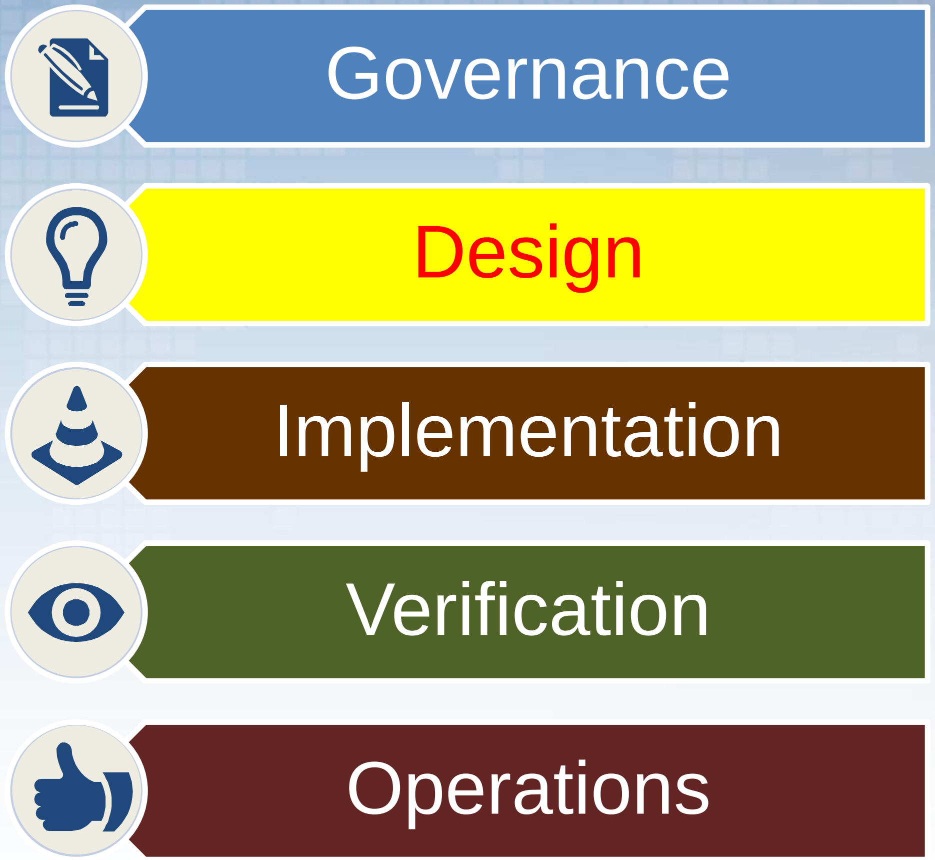
v1.5
STABLE

v2.0
BETA

SAMM Business Functions

- Start with the core activities tied to any organization performing software development

- Named generically, but should resonate with any development stakeholder



Governance

Design

Implementation

Verification

Operations

SAMM Security Practices

- From each of the Business Functions, 3 Security Practices are defined
- The Security Practices cover all areas relevant to software security assurance
- Each one is a ‘silo’ for improvement

Governance	Design	Implementation	Verification	Operations
• Strategy & Metrics • Policy & Compliance • Education & Guidance	• Threat Assessment • Security Requirements • Security Architecture	• Secure Build • Secure Deployment • Defect Management	• Architecture Assessment • Requirements Testing • Security Testing	• Incident Management • Environment Management • Operational Management

SAMM v2.0 Core Framework

Governance		
Strategy & Metrics	Create and Promote	Measure and Improve
Policy & Compliance	Policy and Standards	Compliance Management
Education & Guidance	Training and Awareness	Organization and Culture
Design		
Threat Assessment	Application Risk Profile	Threat Modeling
Security Requirements	Software Requirements	Supplier Security
Secure Architecture	Architecture Design	Technology Management
Implementation		
Secure Build	Build Process	Software Dependencies
Secure Deployment	Deployment Process	Secret Management
Defect Management	Defect Tracking (Flaws/Bugs/Process)	Metrics and Feedback/Learning
Verification		
Architecture Assessment	Architecture Validation	Architecture Compliance
Requirements Testing	**Control Verification**	**Misuse/Abuse Testing**
Security Testing	Scalable Baseline	Deep Understanding
Operations		
Incident Management	Incident Detection	Incident Response
Environment Management	Configuration Hardening	Patching and Updating
Operational Management	Data Protection	System decommissioning / Legacy management

OWASP
Open Web Application
Security Project

Under each Security Practice

- Three successive Objectives under each Practice define how it can be improved over time

  This establishes a notion of a Level at which an organization fulfills a given Practice

- The three Levels for a Practice generally correspond to:

  (0: Implicit starting point with the Practice unfulfilled)

  1: Initial understanding and ad hoc provision of the Practice

  2: Increase efficiency and/or effectiveness of the Practice

  3: Comprehensive mastery of the Practice at scale

Activity Streams & Maturity Levels

Example: Verification - Requirements Testing

Maturity

Streams

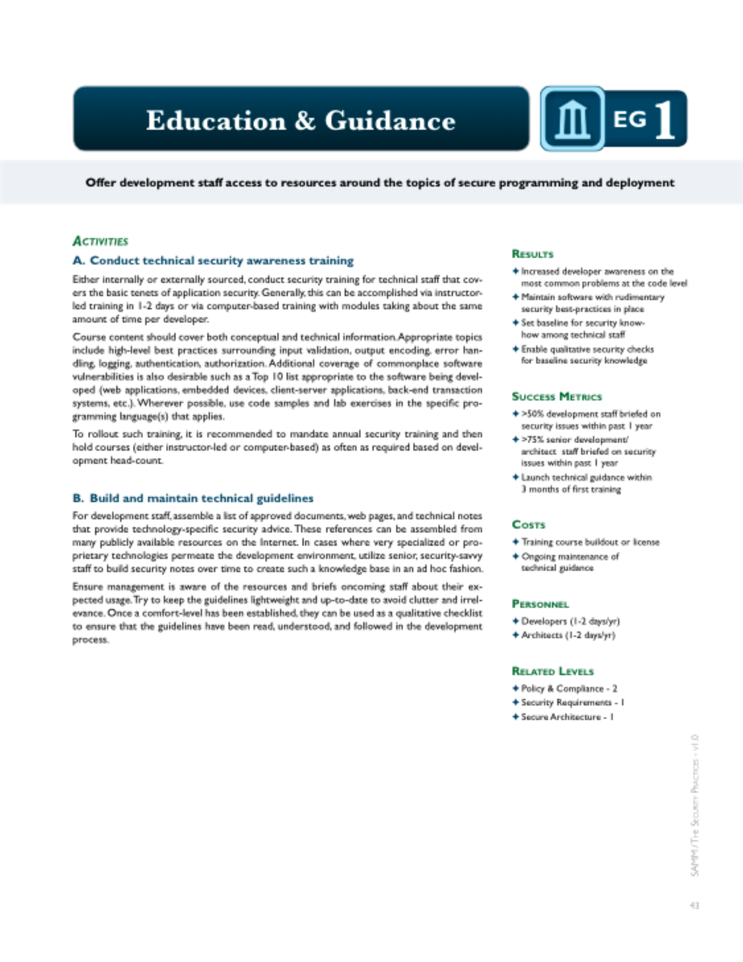
Activities

	A: Control Verification	B: Misuse/Abuse Testing
Maturity 1 - Opportunistically find basic vulnerabilities and other security issues.	Test for standard security controls	Perform security fuzzing testing
Maturity 2 - Perform implementation review to discover application-specific risks against the security requirements.	Derive test cases from known security requirements	Create and test abuse cases and business logic flaw test
Maturity 3 - Maintain the application security level after bug fixes, changes or during maintenance	Perform regression testing (with security unit tests)	Denial of service and security stress testing

- This security practice focuses on creating and integrating both positive (Control Verification) and negative (Misuse/Abuse Testing) security tests based on requirements (user stories)..

Per Level, SAMM defines...

- Objective
- Activities
- Results
- Success Metrics
- Costs
- Personnel
- Related Levels



Education & Guidance

EG 1

Offer development staff access to resources around the topics of secure programming and deployment

Activities

A. Conduct technical security awareness training

Either internally or externally sourced, conduct security training for technical staff that covers the basic tenets of application security. Generally, this can be accomplished via instructor-led training in 1-2 days or via computer-based training with modules taking about the same amount of time per developer.

Course content should cover both conceptual and technical information. Appropriate topics include high-level best practices surrounding input validation, output encoding, error handling, logging, authentication, authorization. Additional coverage of commonplace software vulnerabilities is also desirable such as a Top 10 list appropriate to the software being developed (web applications, embedded devices, client-server applications, back-end transaction systems, etc.). Wherever possible, use code samples and lab exercises in the specific programming language(s) that applies.

To rollout such training, it is recommended to mandate annual security training and then hold courses (either instructor-led or computer-based) as often as required based on development head-count.

B. Build and maintain technical guidelines

For development staff, assemble a list of approved documents, web pages, and technical notes that provide technology-specific security advice. These references can be assembled from many publicly available resources on the Internet. In cases where very specialized or proprietary technologies permeate the development environment, utilize senior, security-savvy staff to build security notes over time to create such a knowledge base in an ad hoc fashion.

Ensure management is aware of the resources and briefs oncoming staff about their expected usage. Try to keep the guidelines lightweight and up-to-date to avoid clutter and irrelevance. Once a comfort-level has been established, they can be used as a qualitative checklist to ensure that the guidelines have been read, understood, and followed in the development process.

Results

- Increased developer awareness on the most common problems at the code level
- Maintain software with rudimentary security best-practices in place
- Set baseline for security know-how among technical staff
- Enable qualitative security checks for baseline security knowledge

Success Metrics

- >50% development staff briefed on security issues within past 1 year
- >75% senior development/ architect staff briefed on security issues within past 1 year
- Launch technical guidance within 3 months of first training

Costs

- Training course buildout or license
- Ongoing maintenance of technical guidance

Personnel

- Developers (1-2 days/yr)
- Architects (1-2 days/yr)

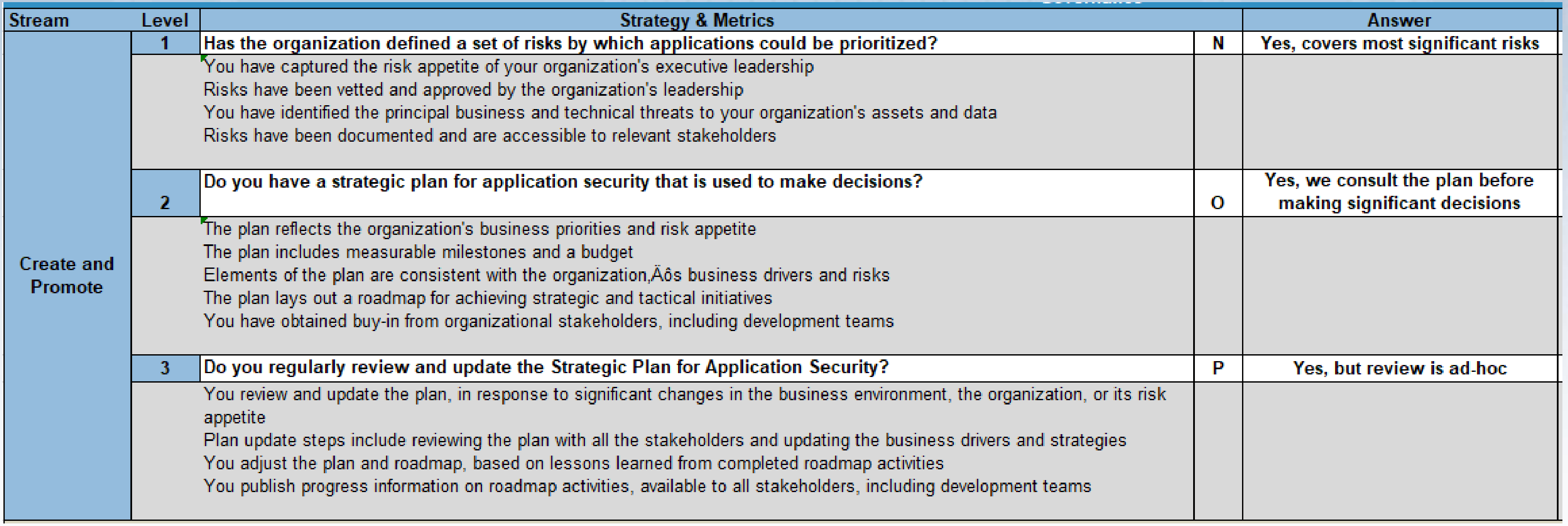
Related Levels

- Policy & Compliance - 2
- Security Requirements - 1
- Secure Architecture - 1

Applying the model

Conducting assessments

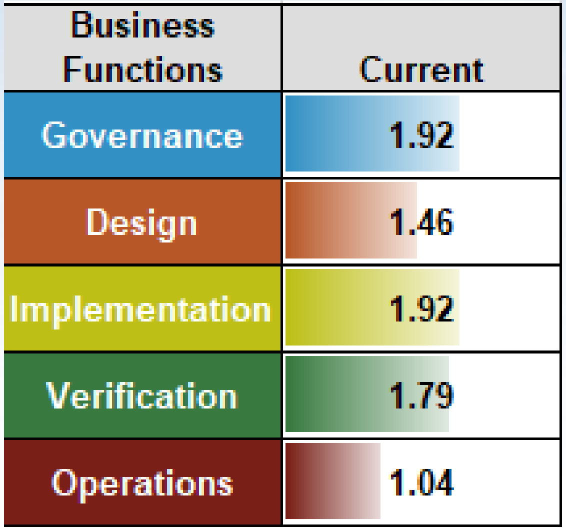
- SAMM includes assessment worksheets for each Security Practice



Stream	Level	Strategy & Metrics		Answer
Create and Promote	1	**Has the organization defined a set of risks by which applications could be prioritized?**	N	Yes, covers most significant risks
		You have captured the risk appetite of your organization's executive leadership Risks have been vetted and approved by the organization's leadership You have identified the principal business and technical threats to your organization's assets and data Risks have been documented and are accessible to relevant stakeholders		
	2	**Do you have a strategic plan for application security that is used to make decisions?**	O	Yes, we consult the plan before making significant decisions
		The plan reflects the organization's business priorities and risk appetite The plan includes measurable milestones and a budget Elements of the plan are consistent with the organization‚Äôs business drivers and risks The plan lays out a roadmap for achieving strategic and tactical initiatives You have obtained buy-in from organizational stakeholders, including development teams		
	3	**Do you regularly review and update the Strategic Plan for Application Security?**	P	Yes, but review is ad-hoc
		You review and update the plan, in response to significant changes in the business environment, the organization, or its risk appetite Plan update steps include reviewing the plan with all the stakeholders and updating the business drivers and strategies You adjust the plan and roadmap, based on lessons learned from completed roadmap activities You publish progress information on roadmap activities, available to all stakeholders, including development teams		

Assessment process

- Assessing activities along two axes:
  - **Coverage**, by means of *questions*
  - **Quality**, by means of mandatory *criteria*



Business Functions	Current
Governance	1.92
Design	1.46
Implementation	1.92
Verification	1.79
Operations	1.04

Creating Scorecards

- Gap analysis

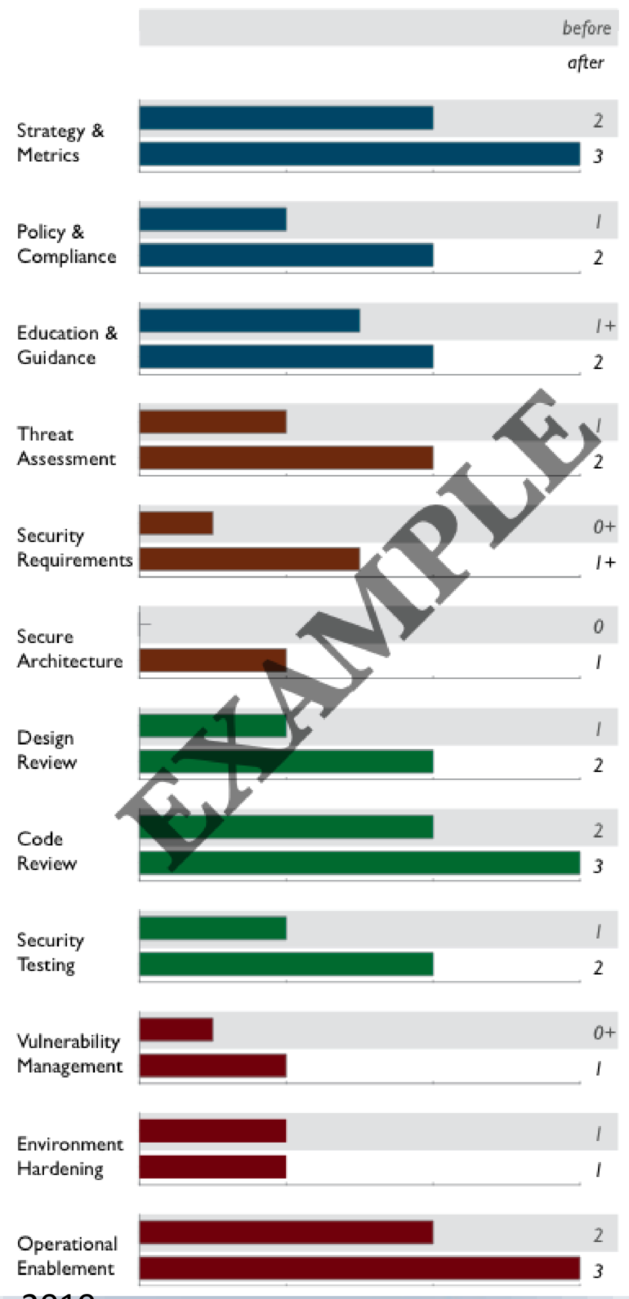
  Capturing scores from detailed assessments versus expected performance levels

- Demonstrating improvement

  Capturing scores from before and a an iteration of assurance program build-out

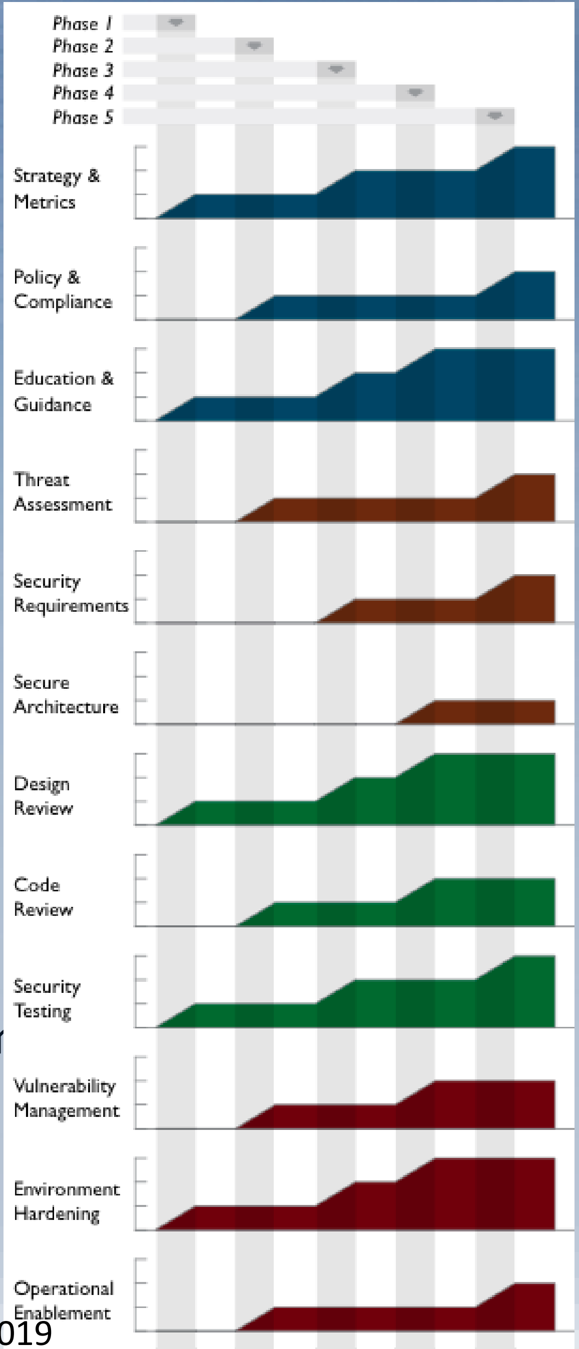
- Ongoing measurement

  Capturing scores over consistent time frames for an assurance progra that is already in place



Roadmap templates

- To make the “building blocks” usable, SAMM defines Roadmaps templates for typical kinds of organizations
  - Independent Software Vendors
  - Online Service Providers
  - Financial Services Organizations
  - Government Organizations
- Organization types chosen because
  - They represent common use-cases
  - Each organization has variations in typical software-induced risk
  - Optimal creation of an assurance program is different for each



Today’s Agenda

1. Introduction to SDLC and OWASP SAMM
2. **Applying OWASP SAMM**

   Methodology

   Assessment Governance

   Assessment Construction

   Assessment Verification

   Assessment Operations

   Setting Improvement Targets
3. OWASP SAMM Tools
4. OWASP SAMM Best Practices

Before you begin

- Organizational Context
- Realistic Goals ?
- Scope ?
- Constraints (budget, timing, resources)
- Affinity with a particular model ?

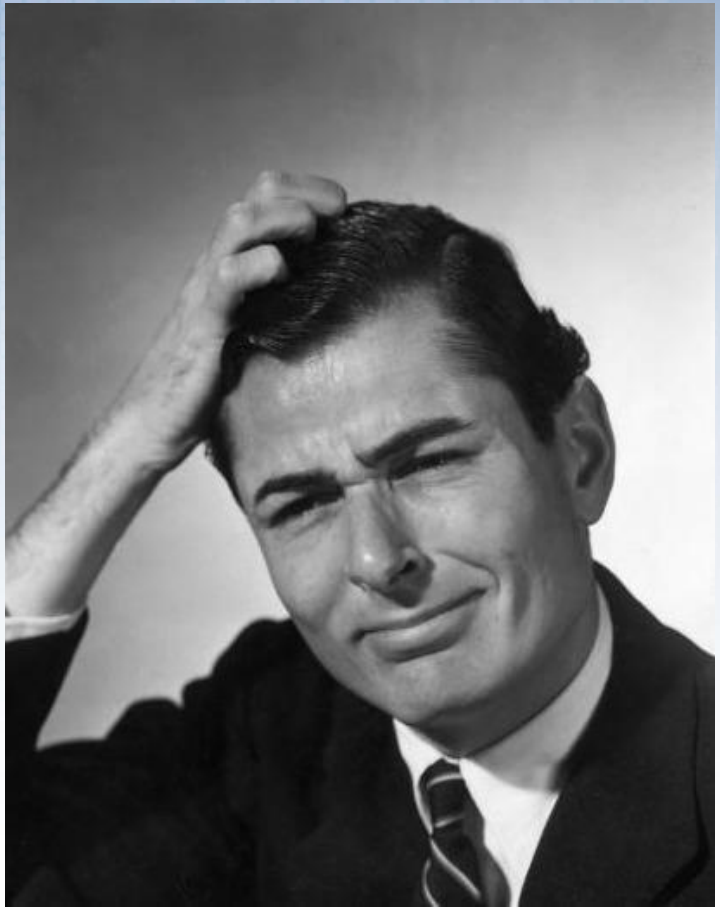
What’s your Company Maturity ?

- In terms of IT **strategy** and application **landscape**
- In terms of software **Development** practices
  - Analysis, Design, Implementation, Testing, Release, Maintenance
  - Structured vs. ad-hoc development
- In terms of **ITSM** practices
  - Configuration, Change, Release, Vulnerability -Mngt.

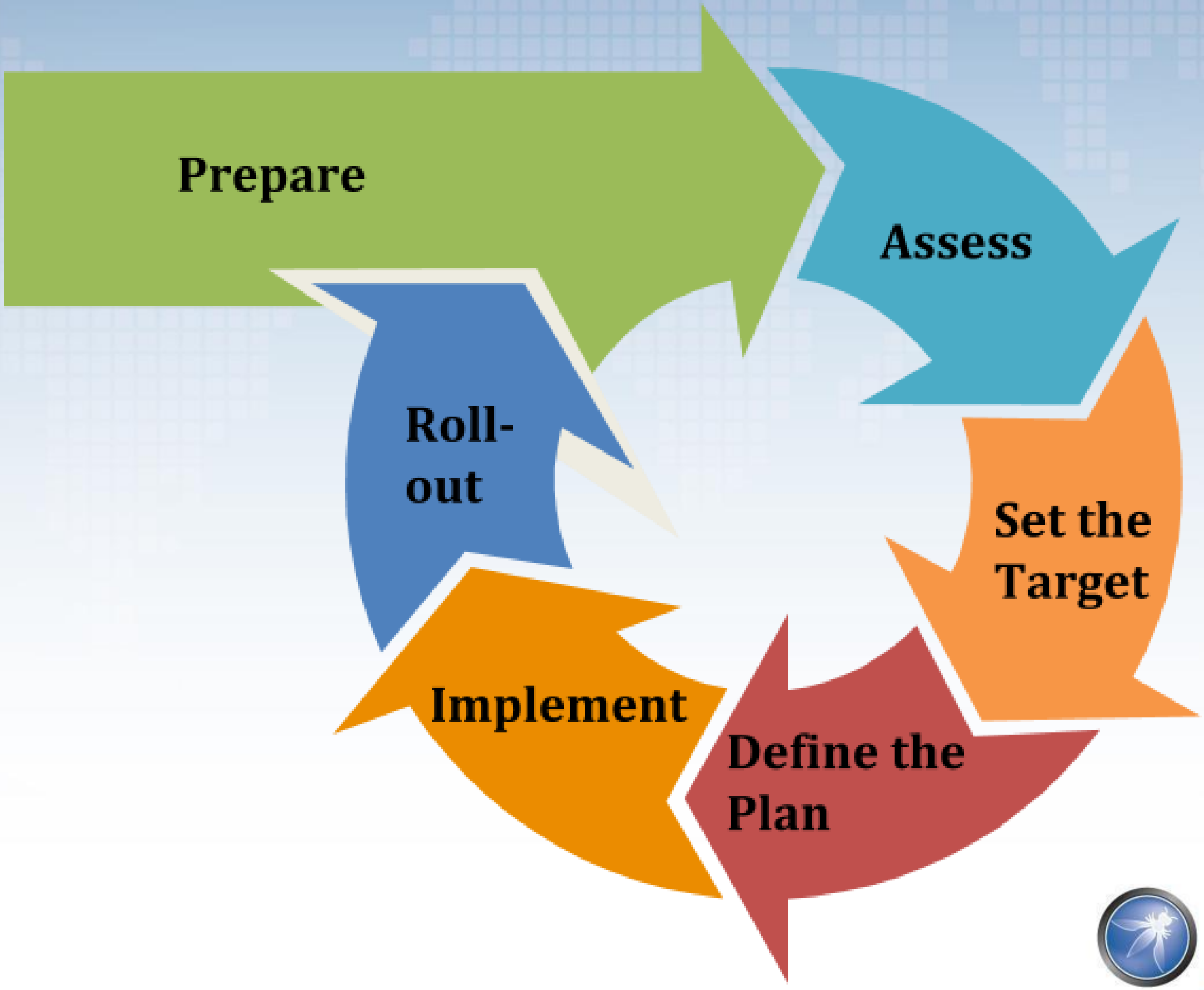
Company Maturity ≈ **Feasibility SDLC Program**

Complicating factors, anyone ?

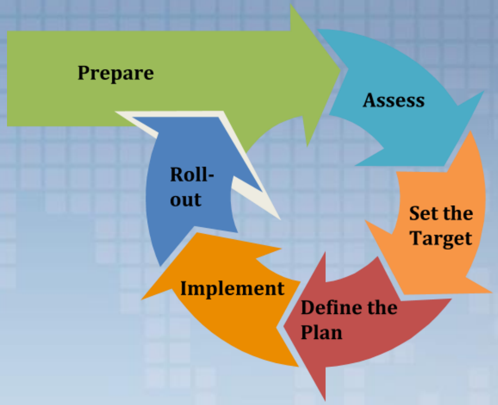
- Different development teams
- Different technology stacks
- Business-IT alignment issues
- Outsourced development
- …



Typical Approach



Prepare



1. Purpose

   Ensure a proper start of the project

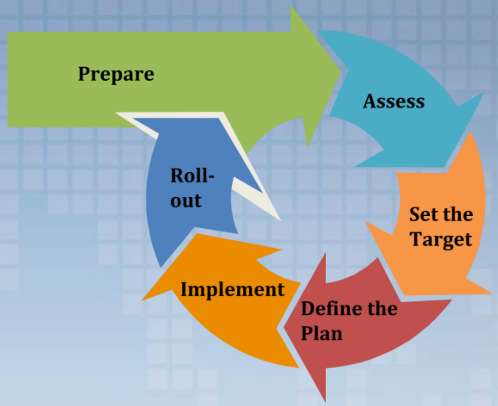
2. Activities

   Define the scope (uniform unit(s))

   Identify stakeholders

   Spread the word

Assess



1. Purpose

Identify and understand the maturity of the 15 practices for the chosen scope

2. Activities

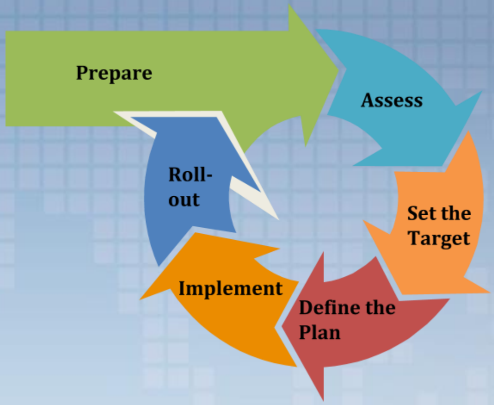
Evaluate current practices

Determine maturity level

INTERVIEW QUESTION
DESCRIBE YOUR BIGGEST MISTAKE AND WHAT YOU LEARNED FROM IT.
Dilbert.com DilbertCartoonist@gmail.com
I TRIED TO GET RID OF AN OLD COUCH BY CHEWING IT INTO TINY PIECES AND LEAVING ONE HANDFUL AT A TIME AT STARBUCKS.
8-20-11 ©2011 Scott Adams, Inc./Dist. by Universal Uclick
AND WHAT DID YOU LEARN?
I LEARNED I SHOULD NEVER TELL THAT STORY.

OWASP
Open Web Application Security Project

Set The Target



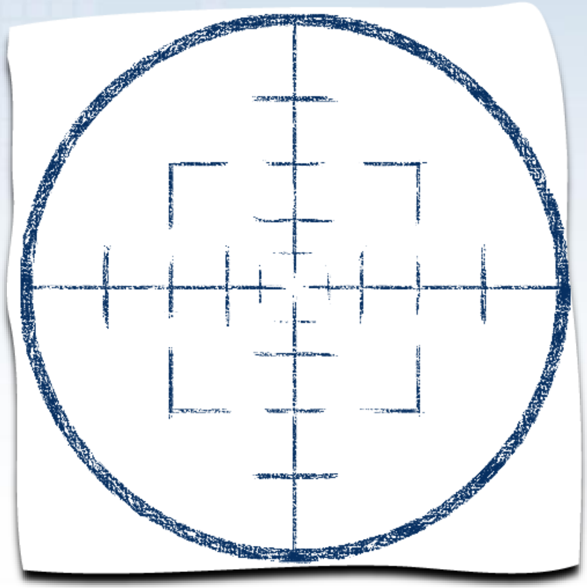
1. Purpose

   Develop a target score to guide you in future improvements

2. Activities

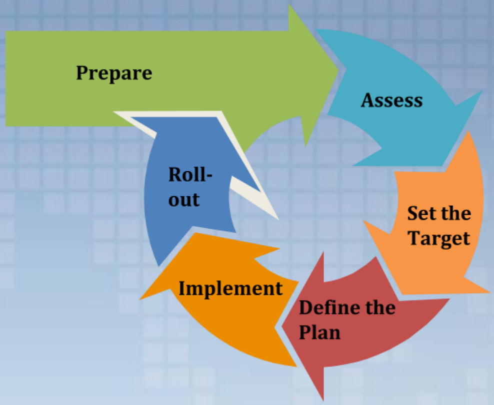
   Define the target

   Estimate overall impact



SAMM Training, OWASP Global AppSec, Amsterdam 2019

Define the plan



1. Purpose

   Define or update the plan to take you to the next level

2. Activities

   Determine change schedule

   Develop/update the roadmap plan

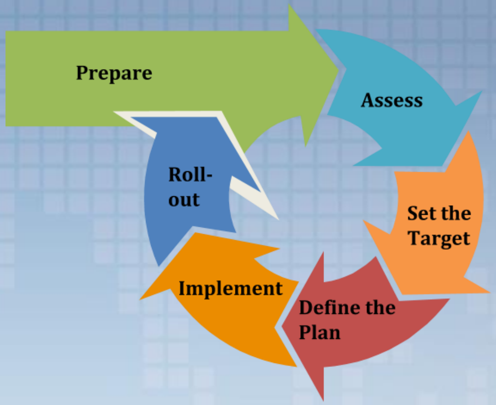
Implement

1. Objective

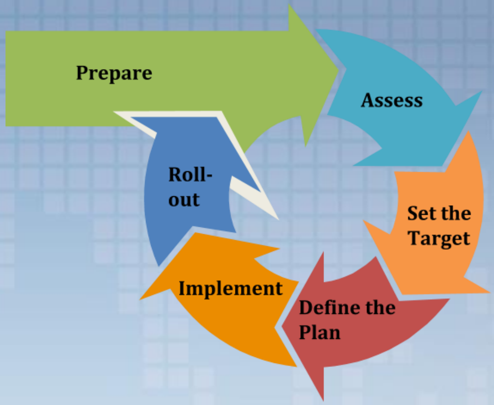
   Work the plan

2. Activities

   Implement activities



Roll-out



1. Objective

   Ensure improvements are available and effectively used

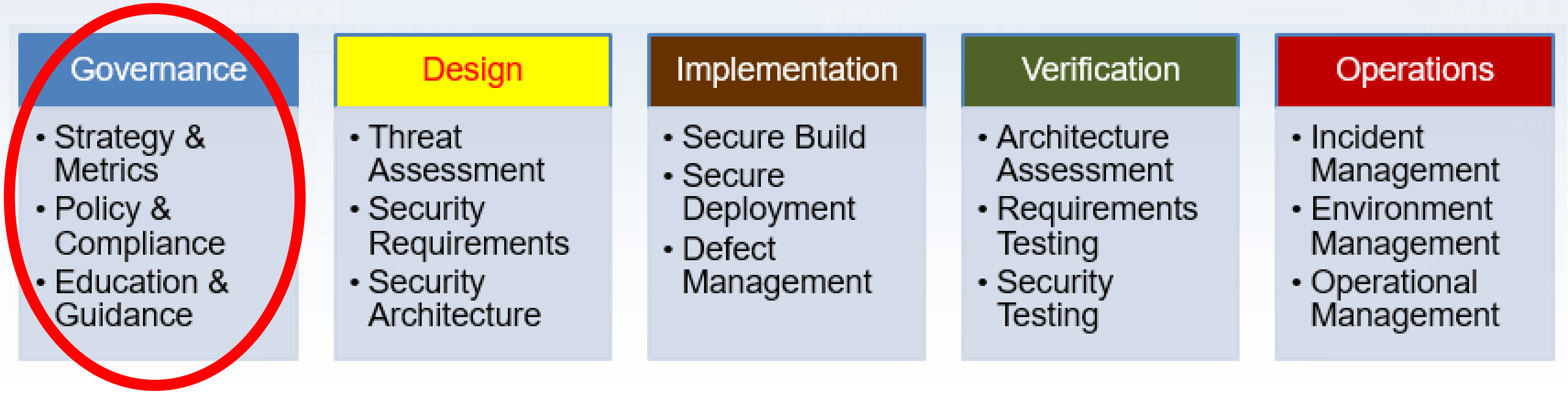
2. Activities

   Evangelize improvements

   Measure effectiveness

Governance

Business Function



Governance

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

Design

- Threat Assessment
- Security Requirements
- Security Architecture

Implementation

- Secure Build
- Secure Deployment
- Defect Management

Verification

- Architecture Assessment
- Requirements Testing
- Security Testing

Operations

- Incident Management
- Environment Management
- Operational Management

Strategy & Metrics

1. Goal is to establish a software assurance framework within an organisation

   Driver for all other OWASP SAMM practices

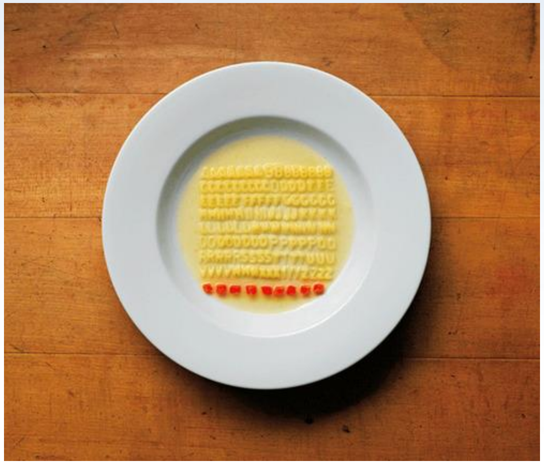
2. Characteristics:

   Measurable

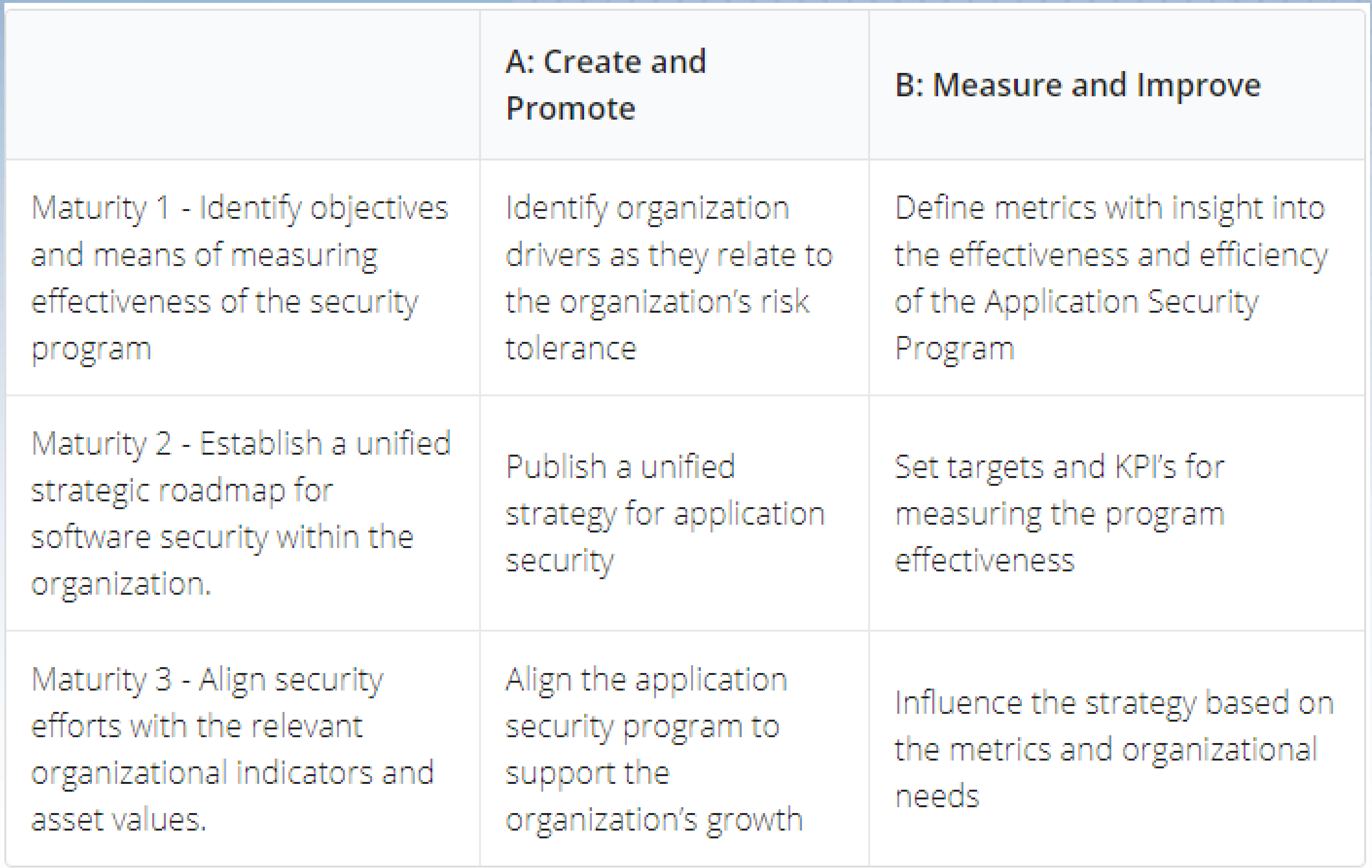
   Aligned with business risk

3. Continuous improvement

VS.



Strategy & Metrics



	A: Create and Promote	B: Measure and Improve
Maturity 1 - Identify objectives and means of measuring effectiveness of the security program	Identify organization drivers as they relate to the organization's risk tolerance	Define metrics with insight into the effectiveness and efficiency of the Application Security Program
Maturity 2 - Establish a unified strategic roadmap for software security within the organization.	Publish a unified strategy for application security	Set targets and KPI's for measuring the program effectiveness
Maturity 3 - Align security efforts with the relevant organizational indicators and asset values.	Align the application security program to support the organization's growth	Influence the strategy based on the metrics and organizational needs

Policy & Compliance

1. Goal is to understand and adhere to legal and regulatory requirements

   Internal standards as well as 3rd party requirements

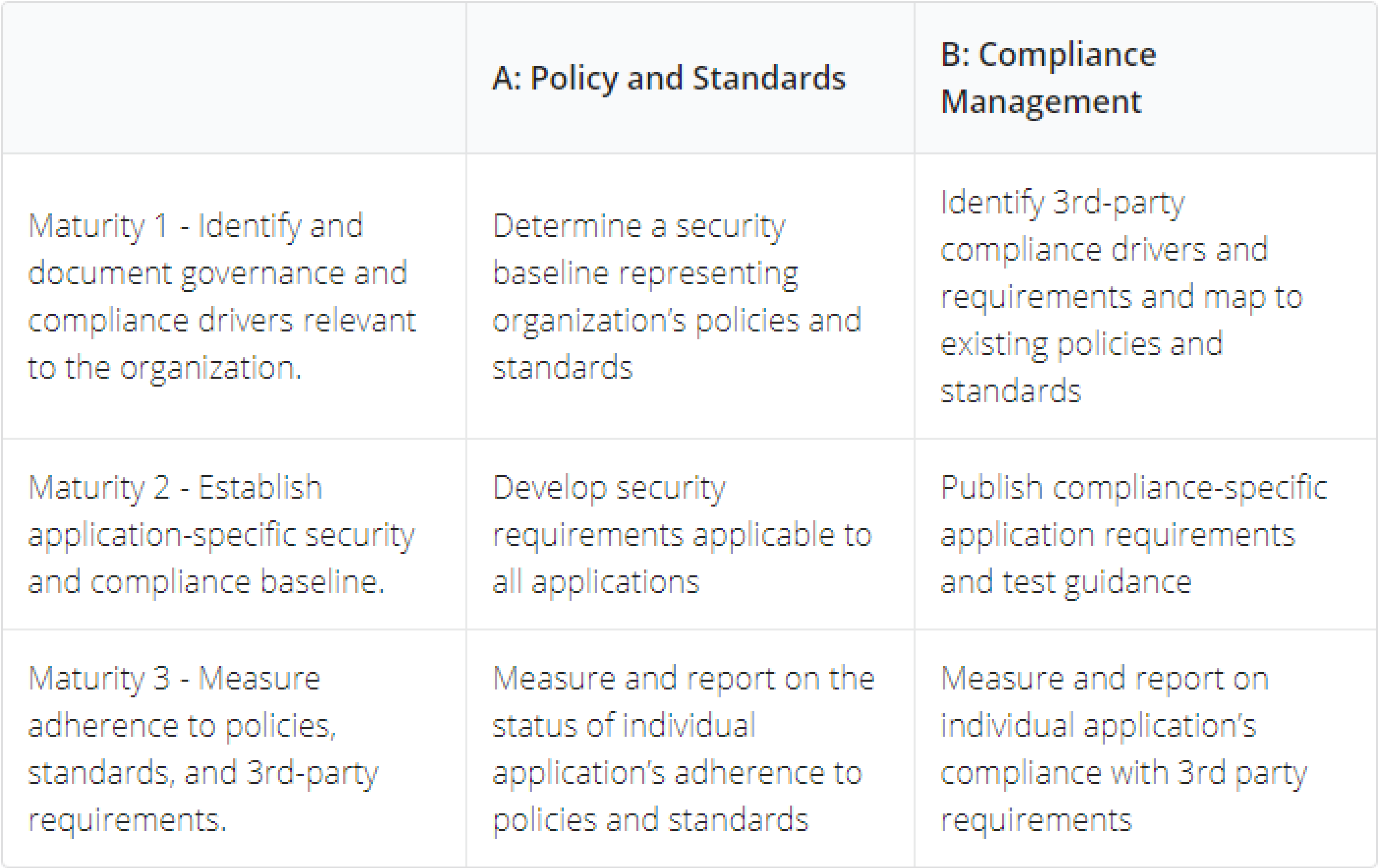
   Both Security and Privacy are important

2. Technical standards become more important

   They are an important driver for software security requirements

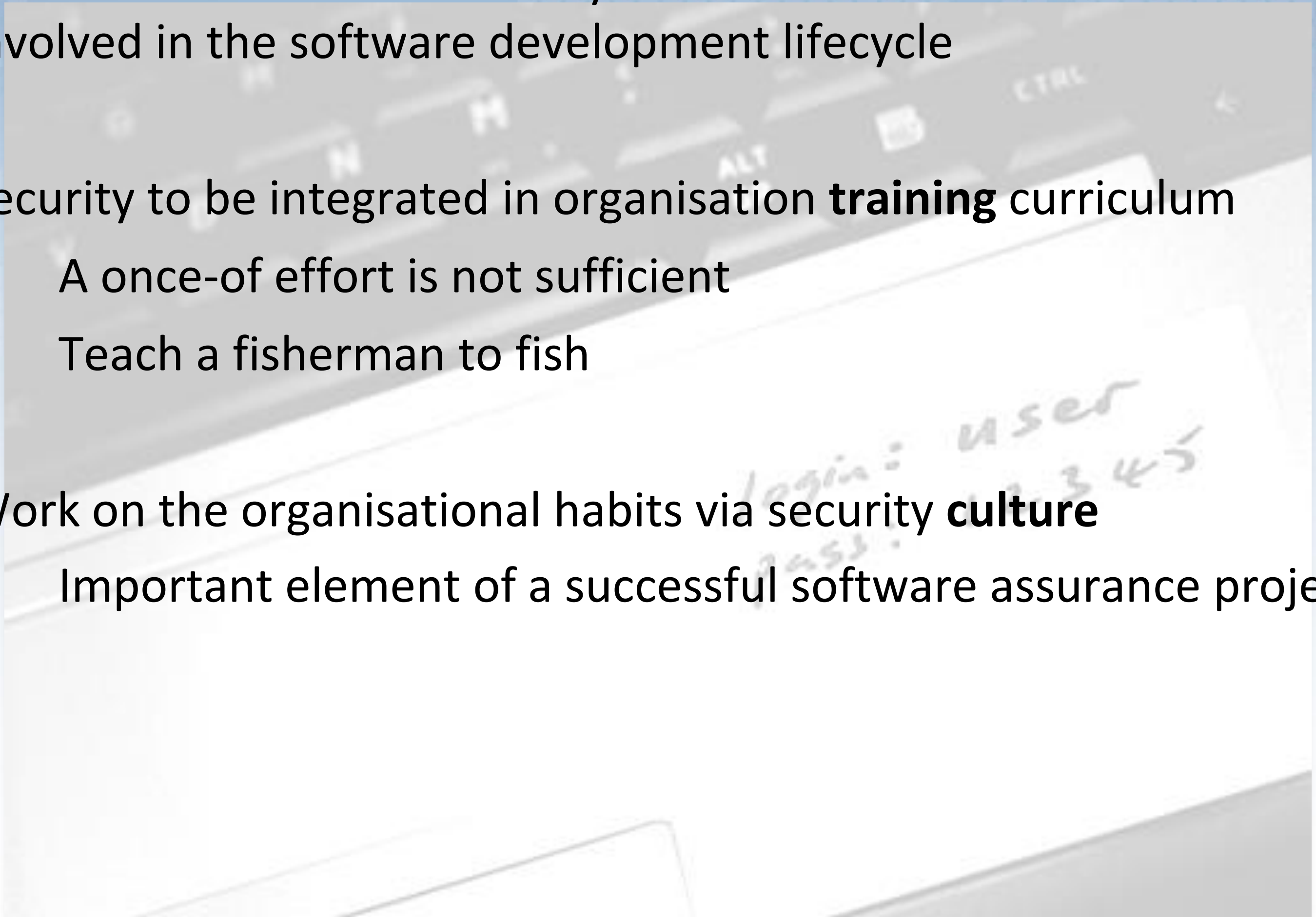
3. Often a very informal practice in organisations

Policy & Compliance



	A: Policy and Standards	B: Compliance Management
Maturity 1 - Identify and document governance and compliance drivers relevant to the organization.	Determine a security baseline representing organization's policies and standards	Identify 3rd-party compliance drivers and requirements and map to existing policies and standards
Maturity 2 - Establish application-specific security and compliance baseline.	Develop security requirements applicable to all applications	Publish compliance-specific application requirements and test guidance
Maturity 3 - Measure adherence to policies, standards, and 3rd-party requirements.	Measure and report on the status of individual application's adherence to policies and standards	Measure and report on individual application's compliance with 3rd party requirements

Education & Guidance



1. Goal is to disseminate security-oriented information to *all* stakeholders involved in the software development lifecycle

2. Security to be integrated in organisation **training** curriculum

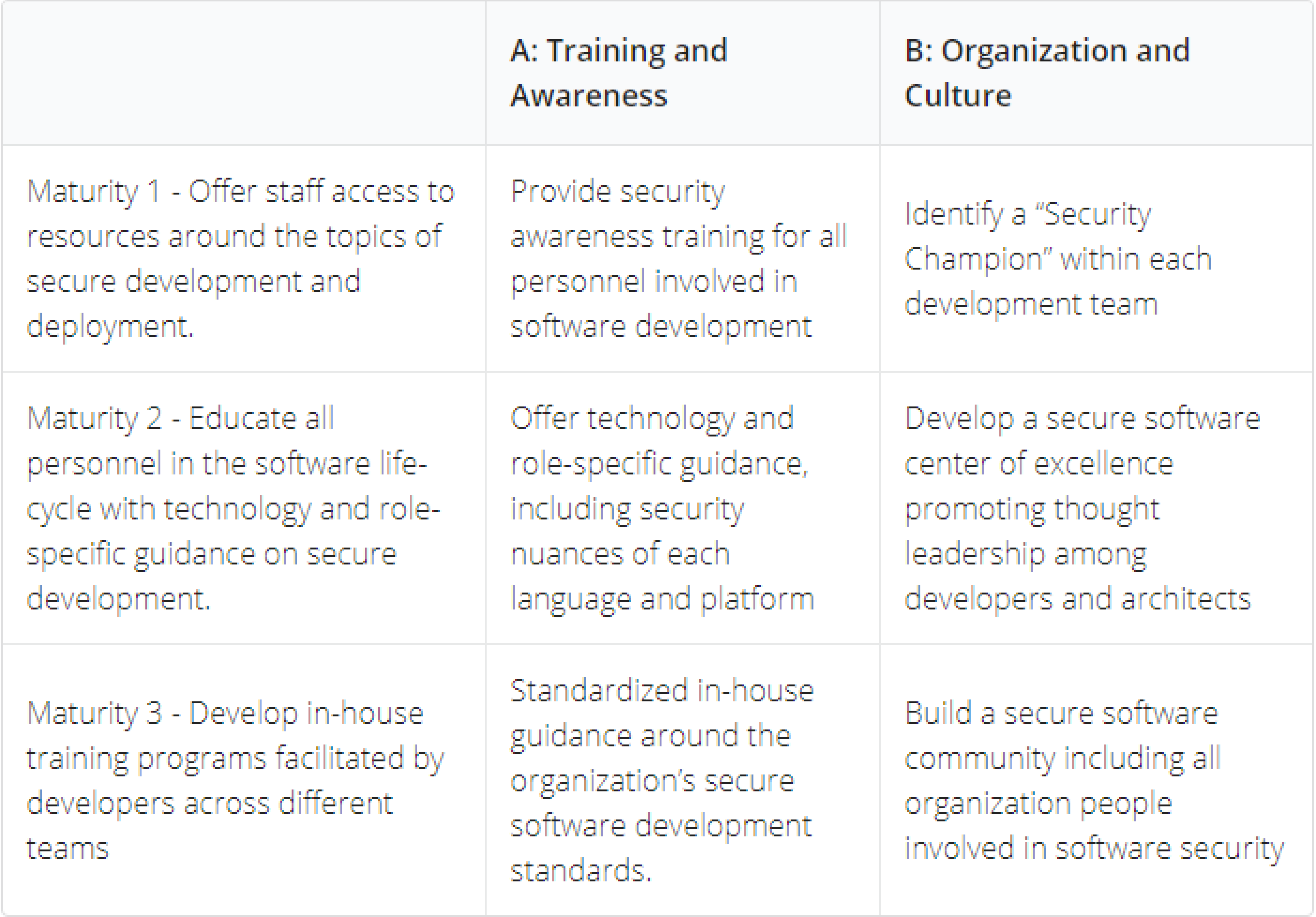
   A once-of effort is not sufficient

   Teach a fisherman to fish

3. Work on the organisational habits via security **culture**

   Important element of a successful software assurance project

Education & Guidance



	A: Training and Awareness	B: Organization and Culture
Maturity 1 - Offer staff access to resources around the topics of secure development and deployment.	Provide security awareness training for all personnel involved in software development	Identify a "Security Champion" within each development team
Maturity 2 - Educate all personnel in the software life-cycle with technology and role-specific guidance on secure development.	Offer technology and role-specific guidance, including security nuances of each language and platform	Develop a secure software center of excellence promoting thought leadership among developers and architects
Maturity 3 - Develop in-house training programs facilitated by developers across different teams	Standardized in-house guidance around the organization's secure software development standards.	Build a secure software community including all organization people involved in software security

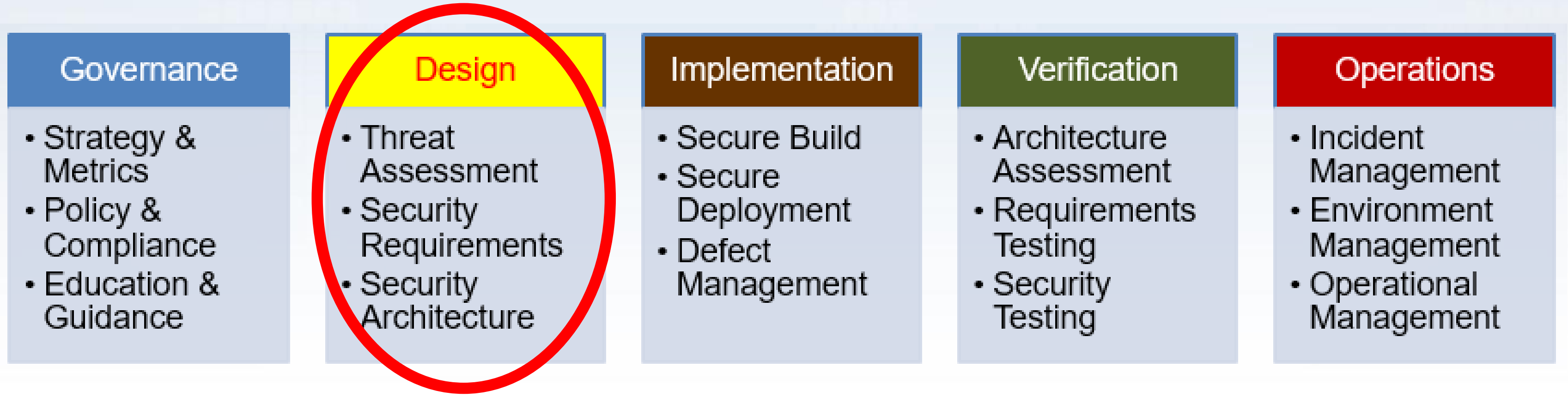
Assessment Exercise

- Use OWASP SAMM to evaluate the development practices in your own company
- Focus on *Governance* Business Function
- Applicable to both Waterfall and Agile models
- Using questionnaires (toolbox)

Assessment wrap-up

- What's your company's score ?
- What's the average scores for the group ?
- Any odd ratings ?

Design
Business Function



Governance	Design	Implementation	Verification	Operations
- Strategy & Metrics - Policy & Compliance - Education & Guidance	- Threat Assessment - Security Requirements - Security Architecture	- Secure Build - Secure Deployment - Defect Management	- Architecture Assessment - Requirements Testing - Security Testing	- Incident Management - Environment Management - Operational Management

Threat Assessment

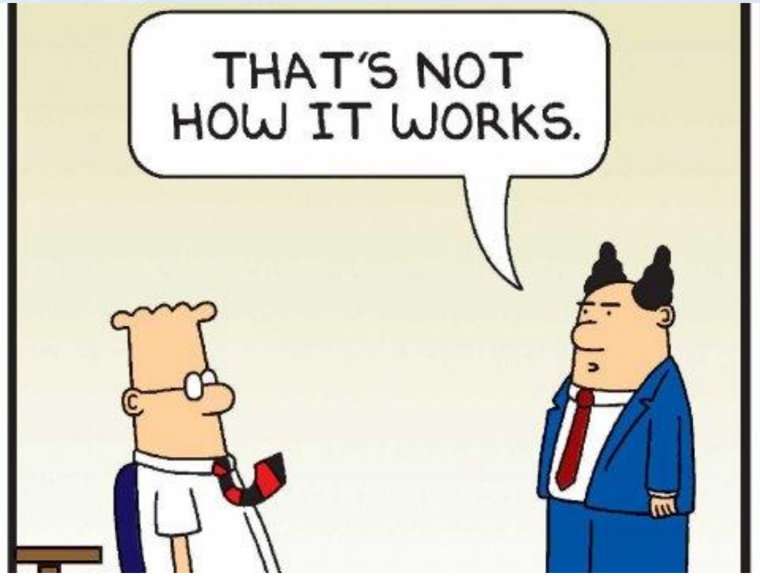
1. Analyze the risks of the application

   from a business perspective, via *risk profiles*

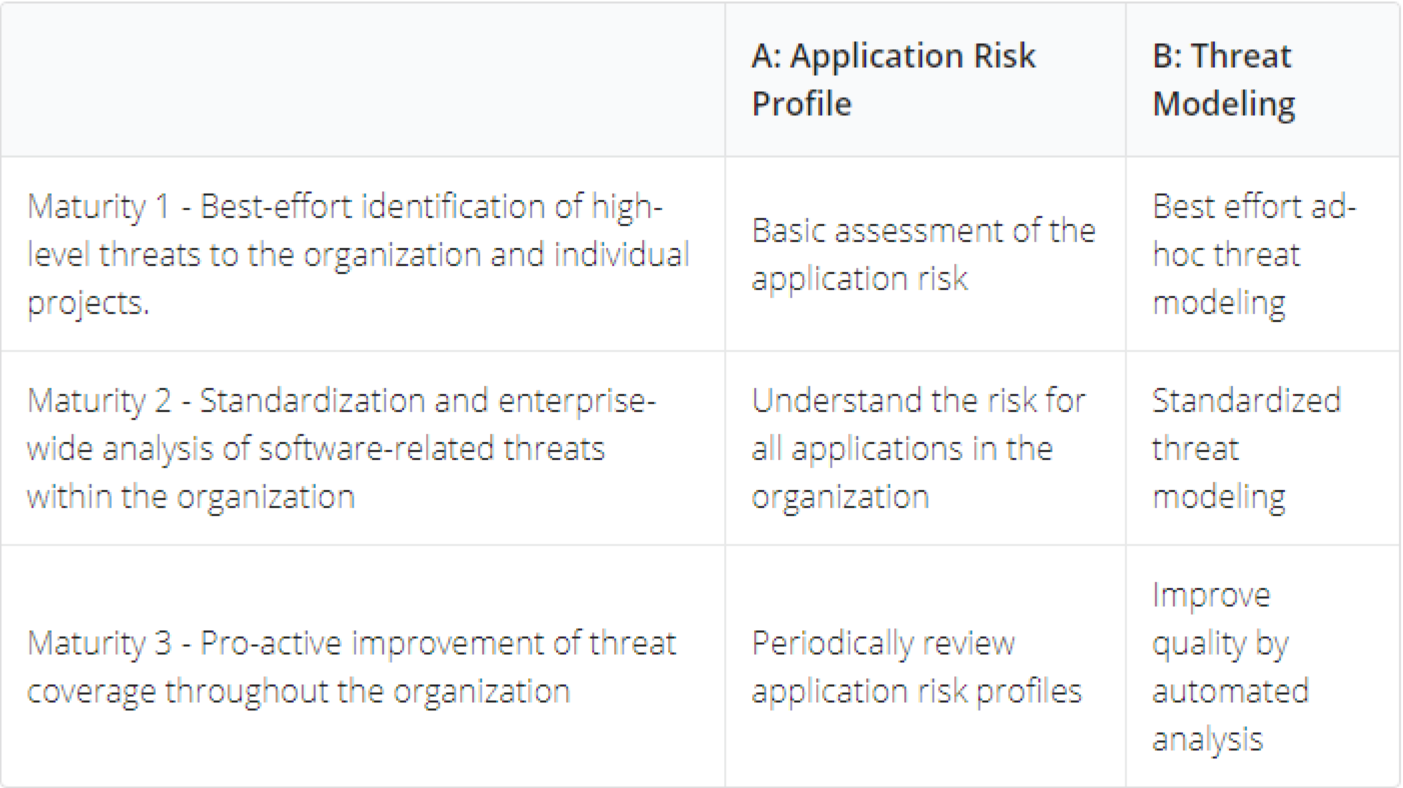
   from a technical perspective, via *threat modeling*

2. Threat modeling is where “the magic” kicks in

   Your imagination is the limit



Threat Assessment



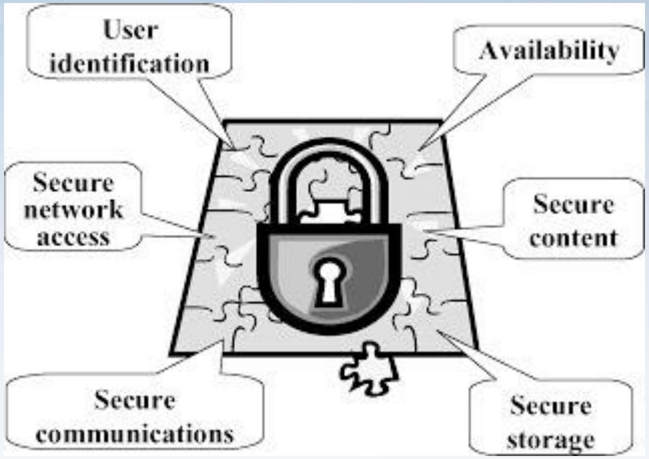
	A: Application Risk Profile	B: Threat Modeling
Maturity 1 - Best-effort identification of high-level threats to the organization and individual projects.	Basic assessment of the application risk	Best effort ad-hoc threat modeling
Maturity 2 - Standardization and enterprise-wide analysis of software-related threats within the organization	Understand the risk for all applications in the organization	Standardized threat modeling
Maturity 3 - Pro-active improvement of threat coverage throughout the organization	Periodically review application risk profiles	Improve quality by automated analysis

Security Requirements

1. Goal is to make security specification more explicit

   Turn security into a positively-spaced problem

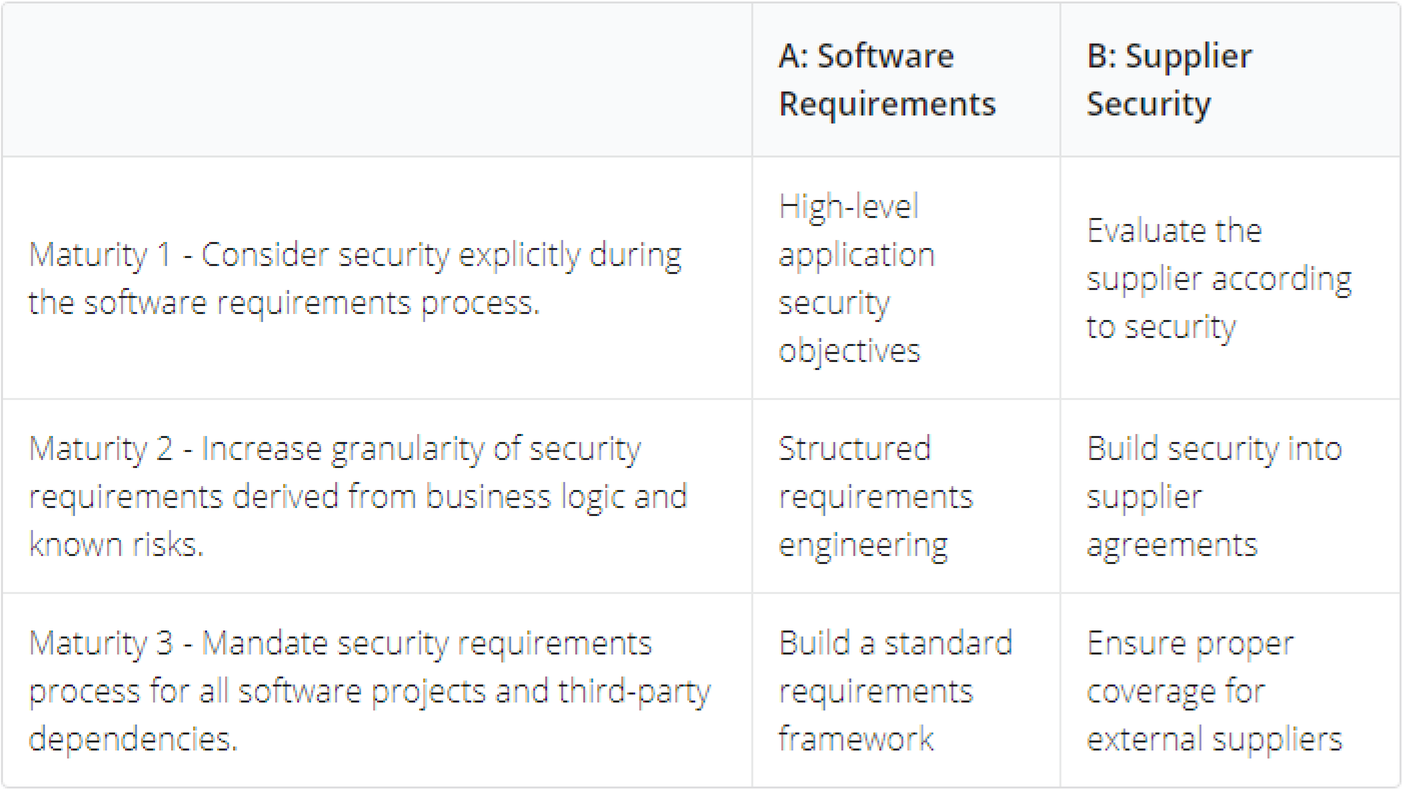
2. Source of security requirements
   - Compliance
   - Standard
   - Functionality
   - Quality



3. Requirements should be specified in a S.M.A.R.T. way

4. Also look into how **suppliers** are dealing with software security

Security Requirements



	A: Software Requirements	B: Supplier Security
Maturity 1 - Consider security explicitly during the software requirements process.	High-level application security objectives	Evaluate the supplier according to security
Maturity 2 - Increase granularity of security requirements derived from business logic and known risks.	Structured requirements engineering	Build security into supplier agreements
Maturity 3 - Mandate security requirements process for all software projects and third-party dependencies.	Build a standard requirements framework	Ensure proper coverage for external suppliers

Secure Architecture

Secure Architecture is a key practice for security. Poor decisions at this step can have major impact, and are often difficult (or costly) to fix.

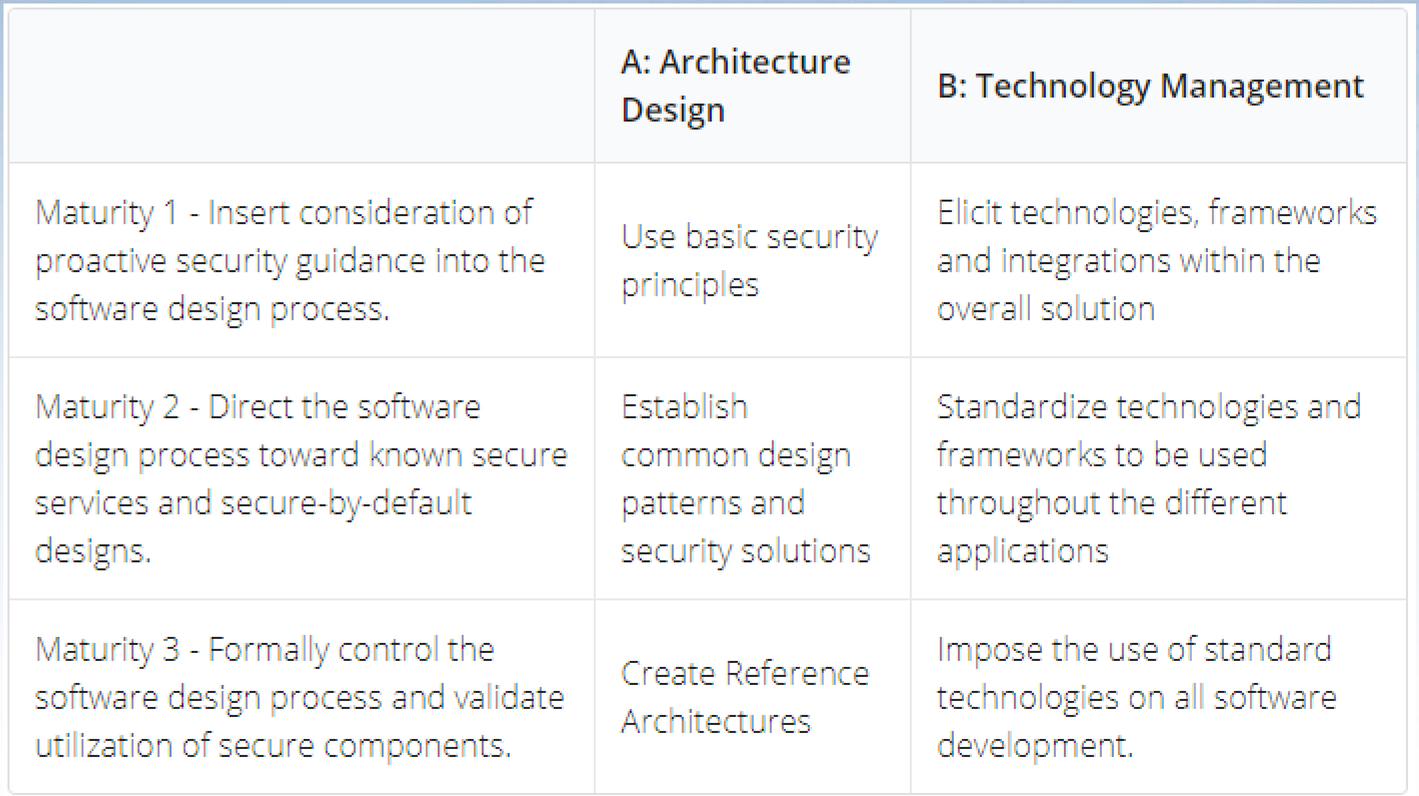
1. Software Architecture **components**

   Ensure that the architecture contains proper elements to meet the security requirements

2. **Supporting Technology**

   Verify that development stacks, deployment tools and other supporting technology are in-line with security expectations

Secure Architecture



	A: Architecture Design	B: Technology Management
Maturity 1 - Insert consideration of proactive security guidance into the software design process.	Use basic security principles	Elicit technologies, frameworks and integrations within the overall solution
Maturity 2 - Direct the software design process toward known secure services and secure-by-default designs.	Establish common design patterns and security solutions	Standardize technologies and frameworks to be used throughout the different applications
Maturity 3 - Formally control the software design process and validate utilization of secure components.	Create Reference Architectures	Impose the use of standard technologies on all software development.

Assessment Exercise

- Use OWASP SAMM to evaluate the development practices in your own company
- Focus on *Design* Business Function
- Applicable to both Waterfall and Agile models
- Using questionnaires (toolbox)

Assessment wrap-up

- What’s your company’s score ?
- What’s the average scores for the group ?
- Any odd ratings ?

Implementation Business Function

Governance	Design	Implementation	Verification	Operations
- Strategy & Metrics - Policy & Compliance - Education & Guidance	- Threat Assessment - Security Requirements - Security Architecture	- Secure Build - Secure Deployment - Defect Management	- Architecture Assessment - Requirements Testing - Security Testing	- Incident Management - Environment Management - Operational Management

OWASP
Open Web Application Security Project

Secure Build

Secure build focuses on using a reliable production process in order to generate secure software artefacts and to avoid issues being introduced during the process

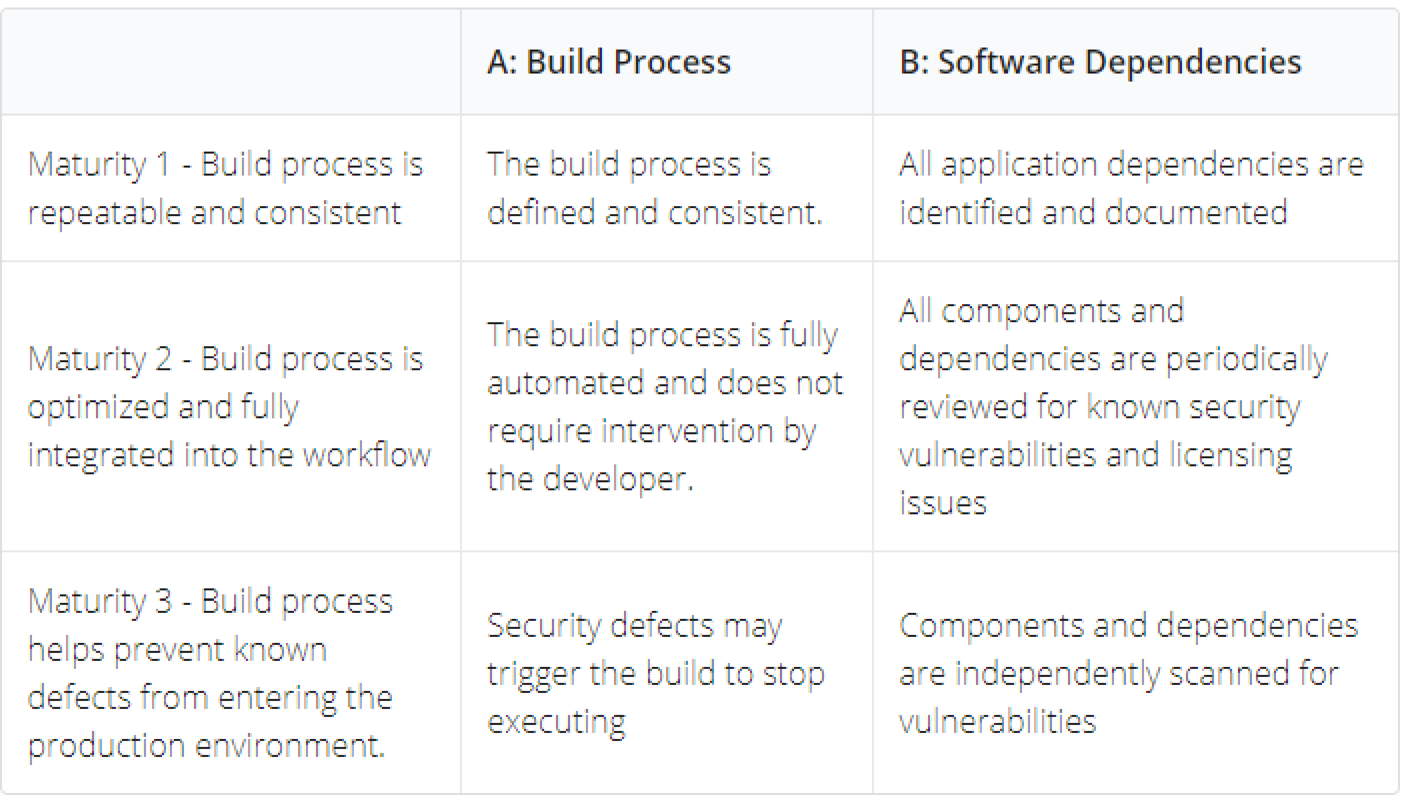
1. An important part is the **build process** itself, where the goal is

   consistent and repeatable

   automated

2. A second part focuses on software dependencies or **3rd party libraries**, aka supply chain security.

Secure Build



	A: Build Process	B: Software Dependencies
Maturity 1 - Build process is repeatable and consistent	The build process is defined and consistent.	All application dependencies are identified and documented
Maturity 2 - Build process is optimized and fully integrated into the workflow	The build process is fully automated and does not require intervention by the developer.	All components and dependencies are periodically reviewed for known security vulnerabilities and licensing issues
Maturity 3 - Build process helps prevent known defects from entering the production environment.	Security defects may trigger the build to stop executing	Components and dependencies are independently scanned for vulnerabilities

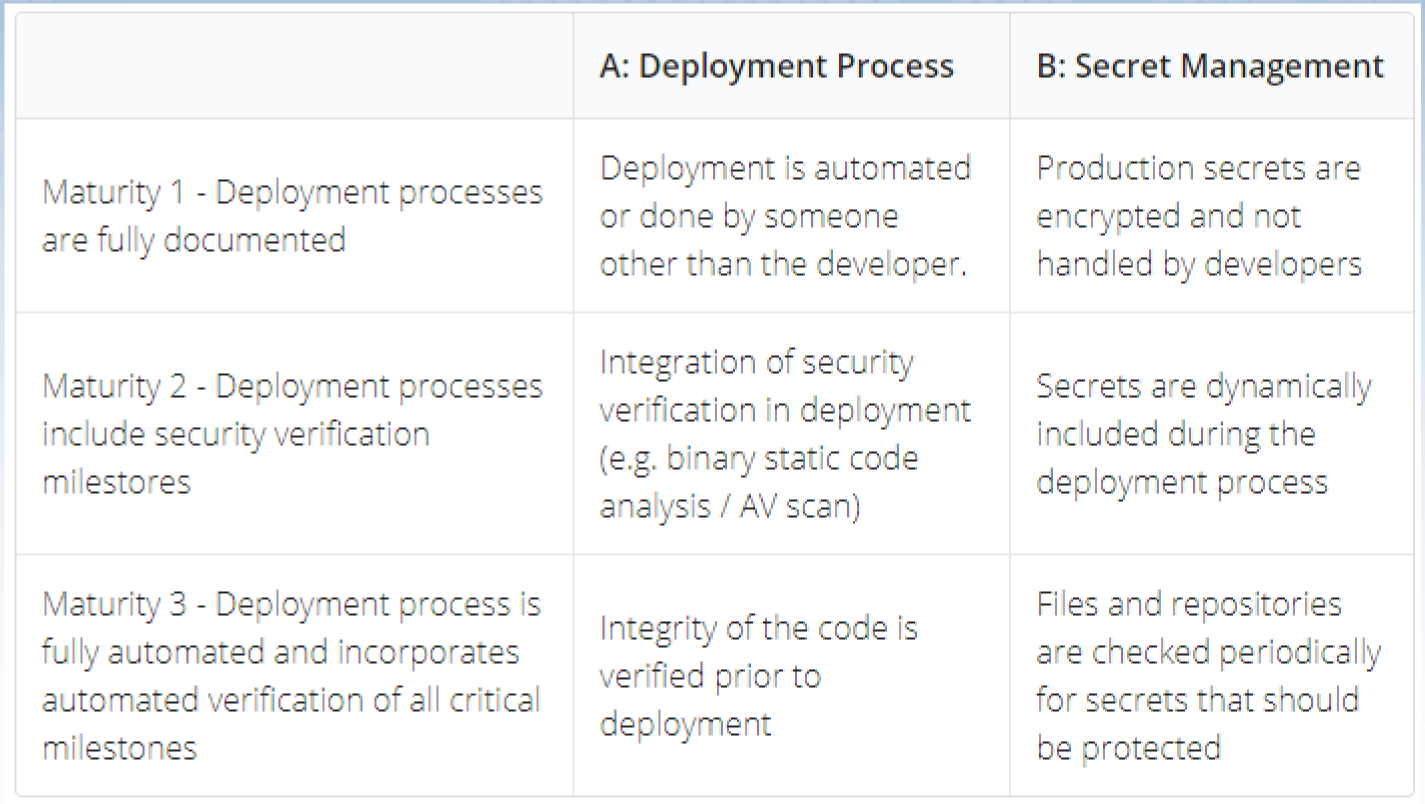
Secure Deployment

One of the final stages in delivering secure software is ensuring the security and integrity of developed applications are not compromised during their deployment.

1. In the **deployment process**, appropriate protection can a repeatable deployment process, separation of duties, etc.

2. All **secrets** required for the software to run must be properly protected for deployment and during execution. Tools such as password vaults can help to achieve this.

Secure Deployment



	A: Deployment Process	B: Secret Management
Maturity 1 - Deployment processes are fully documented	Deployment is automated or done by someone other than the developer.	Production secrets are encrypted and not handled by developers
Maturity 2 - Deployment processes include security verification milestores	Integration of security verification in deployment (e.g. binary static code analysis / AV scan)	Secrets are dynamically included during the deployment process
Maturity 3 - Deployment process is fully automated and incorporates automated verification of all critical milestones	Integrity of the code is verified prior to deployment	Files and repositories are checked periodically for secrets that should be protected

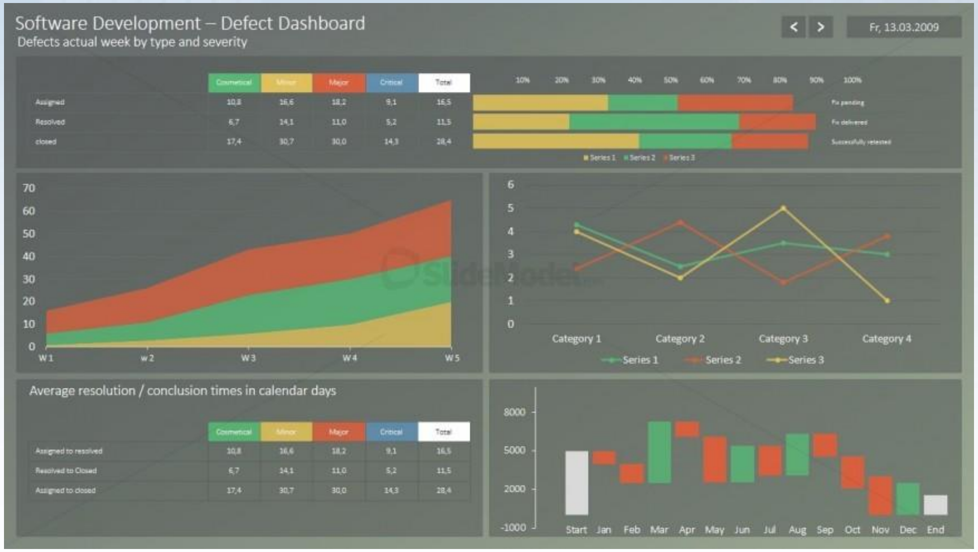
Defect Management

The Defect Management practice focuses on collecting, recording, and analysing software security defects and enriching them with information to drive metrics-based decisions.

It is a central funnel for all defects identified in other security practices.

1. A **defect tracking** solution helps the organization to keep track of identified problems, and to manage them in a controlled manner.

2. **Analysis** of defects supported by **metrics** can help to increase awareness and guide improvement programs in the organization.



Software Development – Defect Dashboard

Defects actual week by type and severity

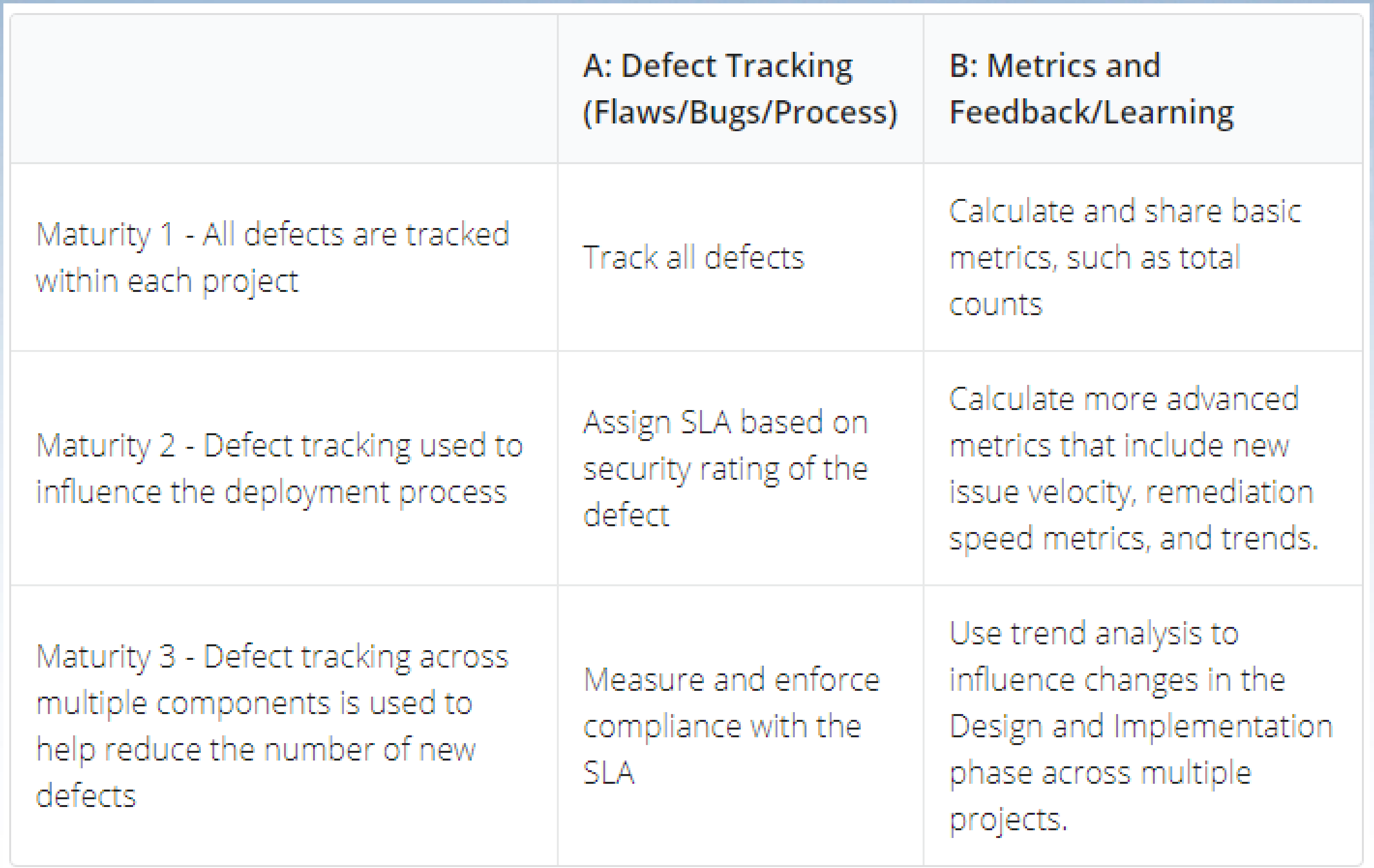
Fr, 13.03.2009

	Cosmetical	Minor	Major	Critical	Total
Assigned	10,8	16,6	18,2	9,1	16,5
Resolved	6,7	14,1	11,0	5,2	11,5
closed	17,4	30,7	30,0	14,3	28,4

Average resolution / conclusion times in calendar days

	Cosmetical	Minor	Major	Critical	Total
Assigned to resolved	10,8	16,6	18,2	9,1	16,5
Resolved to Closed	6,7	14,1	11,0	5,2	11,5
Assigned to closed	17,4	30,7	30,0	14,3	28,4

Defect Management



	A: Defect Tracking (Flaws/Bugs/Process)	B: Metrics and Feedback/Learning
Maturity 1 - All defects are tracked within each project	Track all defects	Calculate and share basic metrics, such as total counts
Maturity 2 - Defect tracking used to influence the deployment process	Assign SLA based on security rating of the defect	Calculate more advanced metrics that include new issue velocity, remediation speed metrics, and trends.
Maturity 3 - Defect tracking across multiple components is used to help reduce the number of new defects	Measure and enforce compliance with the SLA	Use trend analysis to influence changes in the Design and Implementation phase across multiple projects.

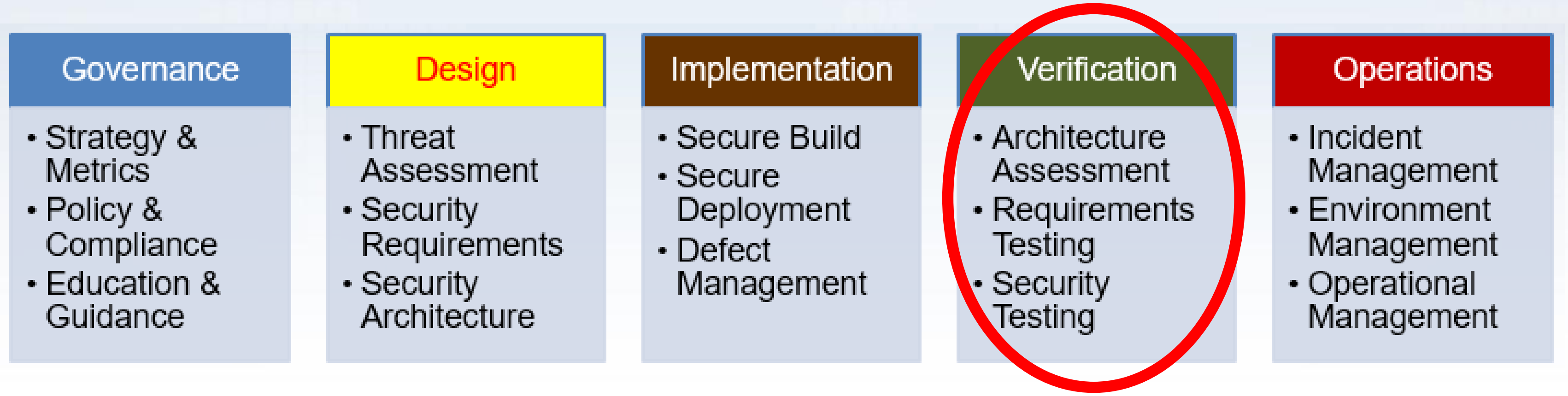
Assessment Exercise

- Use OWASP SAMM to evaluate the development practices in your own company
- Focus on *Implementation* Business Functions
- Applicable to both Waterfall and Agile models
- Using questionnaires (toolbox)

Assessment wrap-up

- What's your company's score ?
- What's the average scores for the group ?
- Any odd ratings ?

Verification
Business Function



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

Design
- Threat Assessment
- Security Requirements
- Security Architecture

Implementation
- Secure Build
- Secure Deployment
- Defect Management

Verification
- Architecture Assessment
- Requirements Testing
- Security Testing

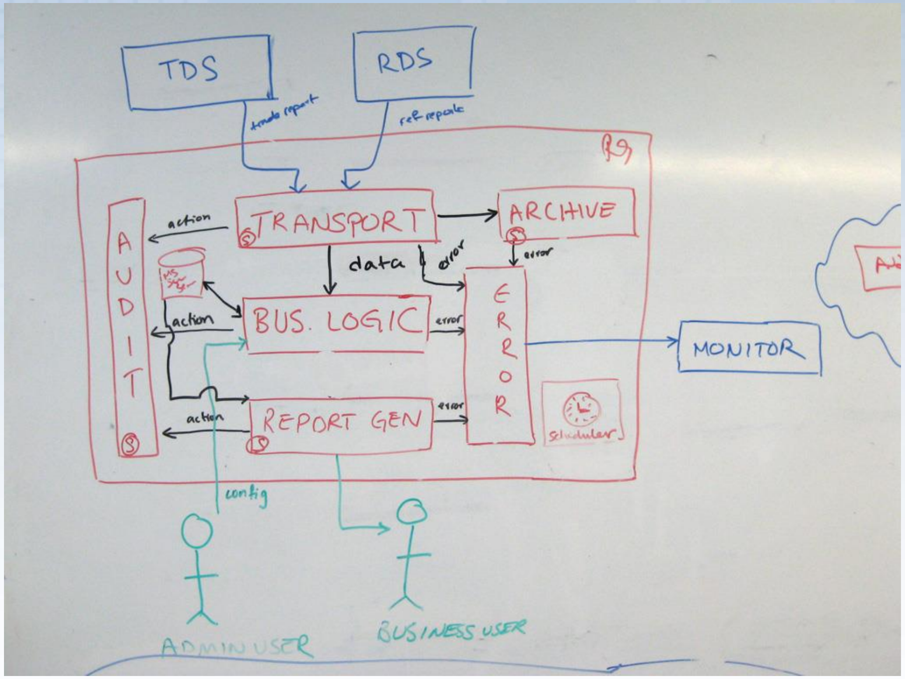
Operations
- Incident Management
- Environment Management
- Operational Management

Architecture Assessment

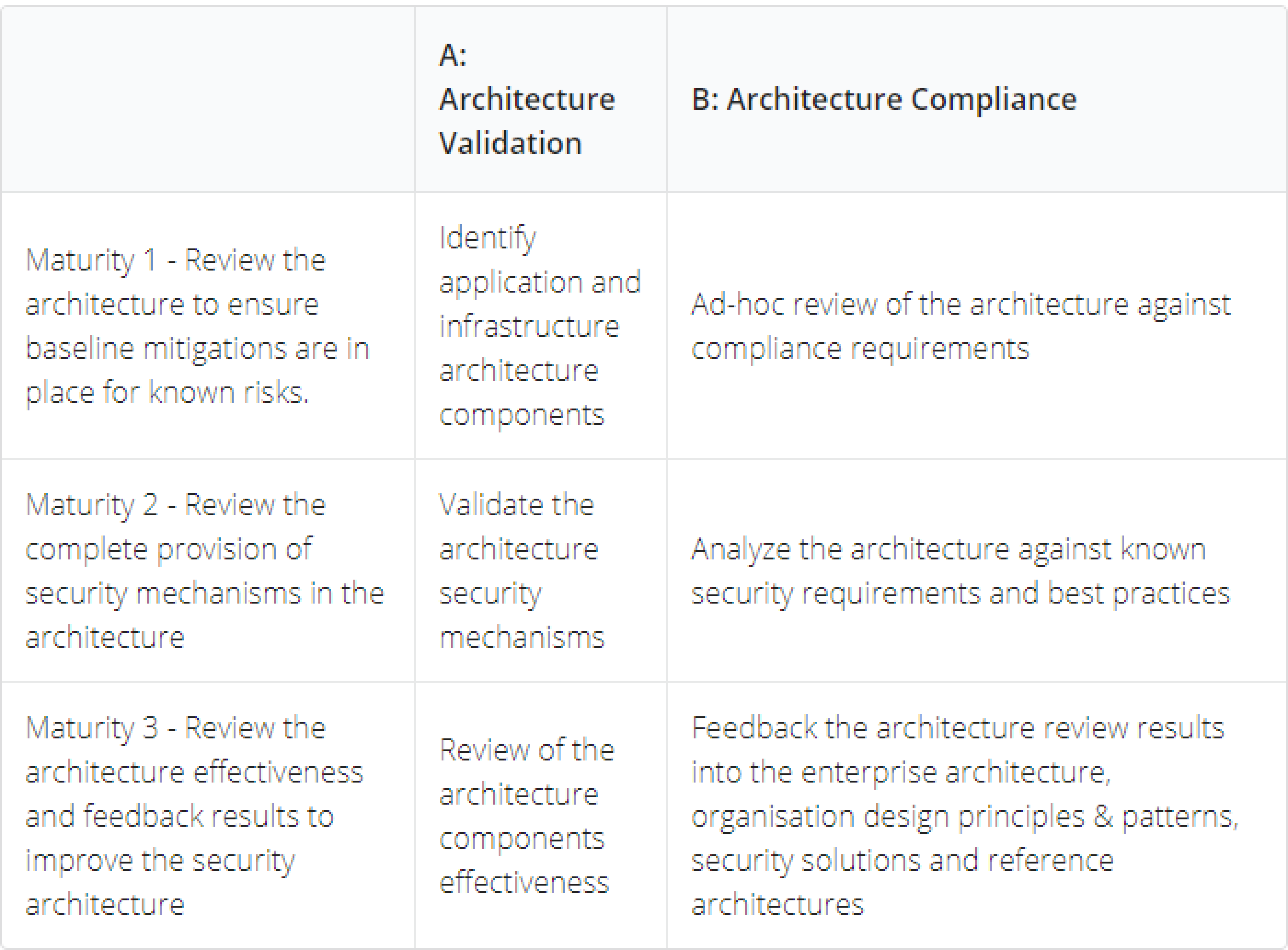
1. Verify whether the meets **security** and **compliance** requirements

2. Covers both software and supporting infrastructure

3. Rigorous inspection of data flows & security mechanisms



Architecture Assessment



	A: Architecture Validation	B: Architecture Compliance
Maturity 1 - Review the architecture to ensure baseline mitigations are in place for known risks.	Identify application and infrastructure architecture components	Ad-hoc review of the architecture against compliance requirements
Maturity 2 - Review the complete provision of security mechanisms in the architecture	Validate the architecture security mechanisms	Analyze the architecture against known security requirements and best practices
Maturity 3 - Review the architecture effectiveness and feedback results to improve the security architecture	Review of the architecture components effectiveness	Feedback the architecture review results into the enterprise architecture, organisation design principles & patterns, security solutions and reference architectures

Requirements Testing

First perspective for testing: test the software according to the requirements

1. Conduct positive and negative security tests to verify that the software operates as specified.

   From the **known security requirements**, identify and implement a set of security test cases to check the software for correct functionality.

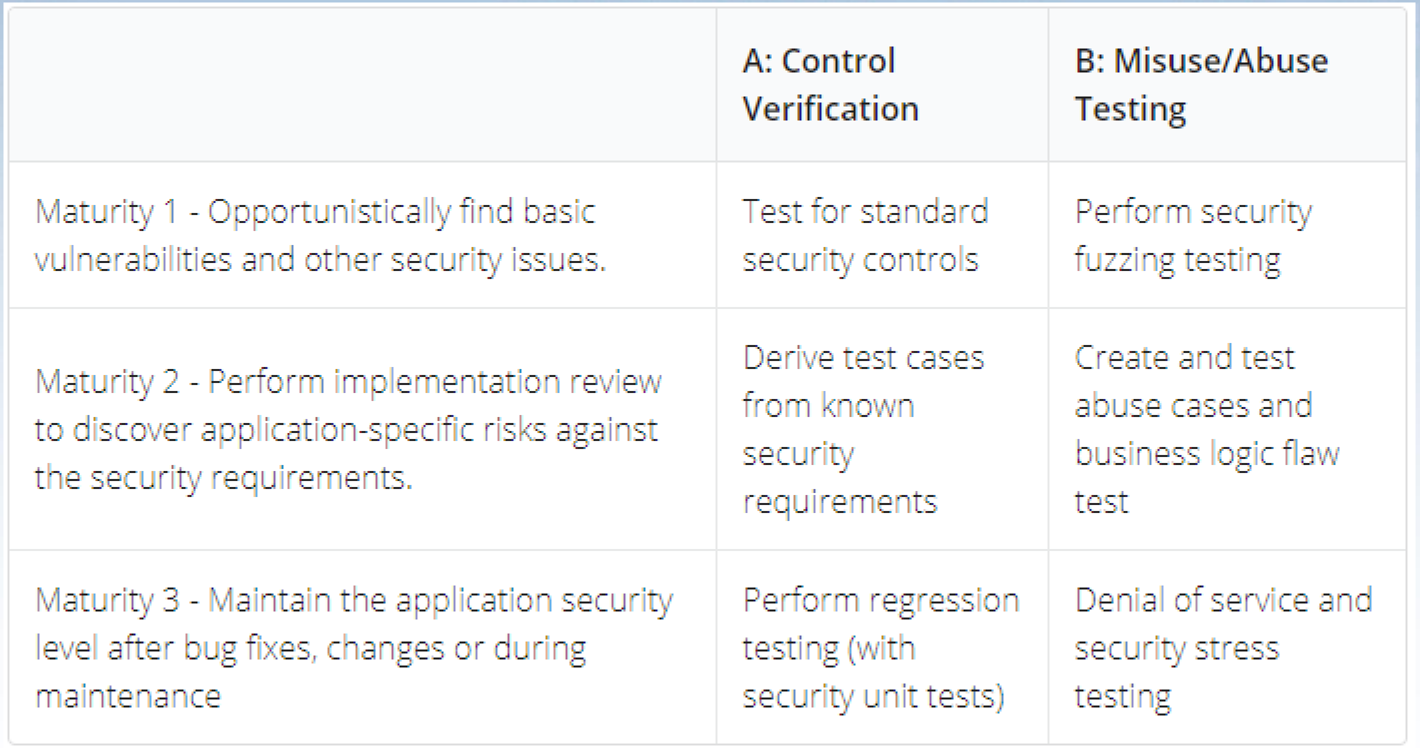
   Use **abuse testing** for an application to run concrete security tests that directly or indirectly exploit identified abuse scenarios.

2. Automate security testing (for each release) via security test automation and automated regression testing

THIS IS RODNEY. HE'S IN CHARGE OF PRODUCT SAFETY TESTING.
IS OUR NEW PRODUCT SAFE ENOUGH TO START SELLING?
$#%⊖
DID THAT SOUND LIKE "SHIP" TO YOU?
Dilbert.com DilbertCartoonist@gmail.com
8·21·10 ©2010 Scott Adams, Inc./Dist. by UFS, Inc.

SAMM Training, OWASP Global AppSec, Amsterdam 2019

Requirements Testing



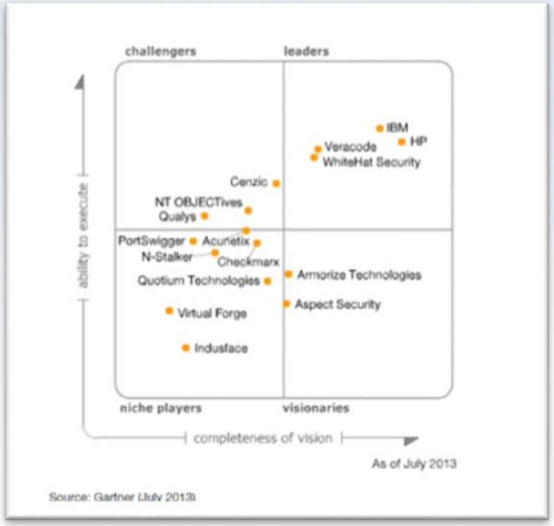
	A: Control Verification	B: Misuse/Abuse Testing
Maturity 1 - Opportunistically find basic vulnerabilities and other security issues.	Test for standard security controls	Perform security fuzzing testing
Maturity 2 - Perform implementation review to discover application-specific risks against the security requirements.	Derive test cases from known security requirements	Create and test abuse cases and business logic flaw test
Maturity 3 - Maintain the application security level after bug fixes, changes or during maintenance	Perform regression testing (with security unit tests)	Denial of service and security stress testing

Security Testing

Second perspective for testing: test the software according to security best practices

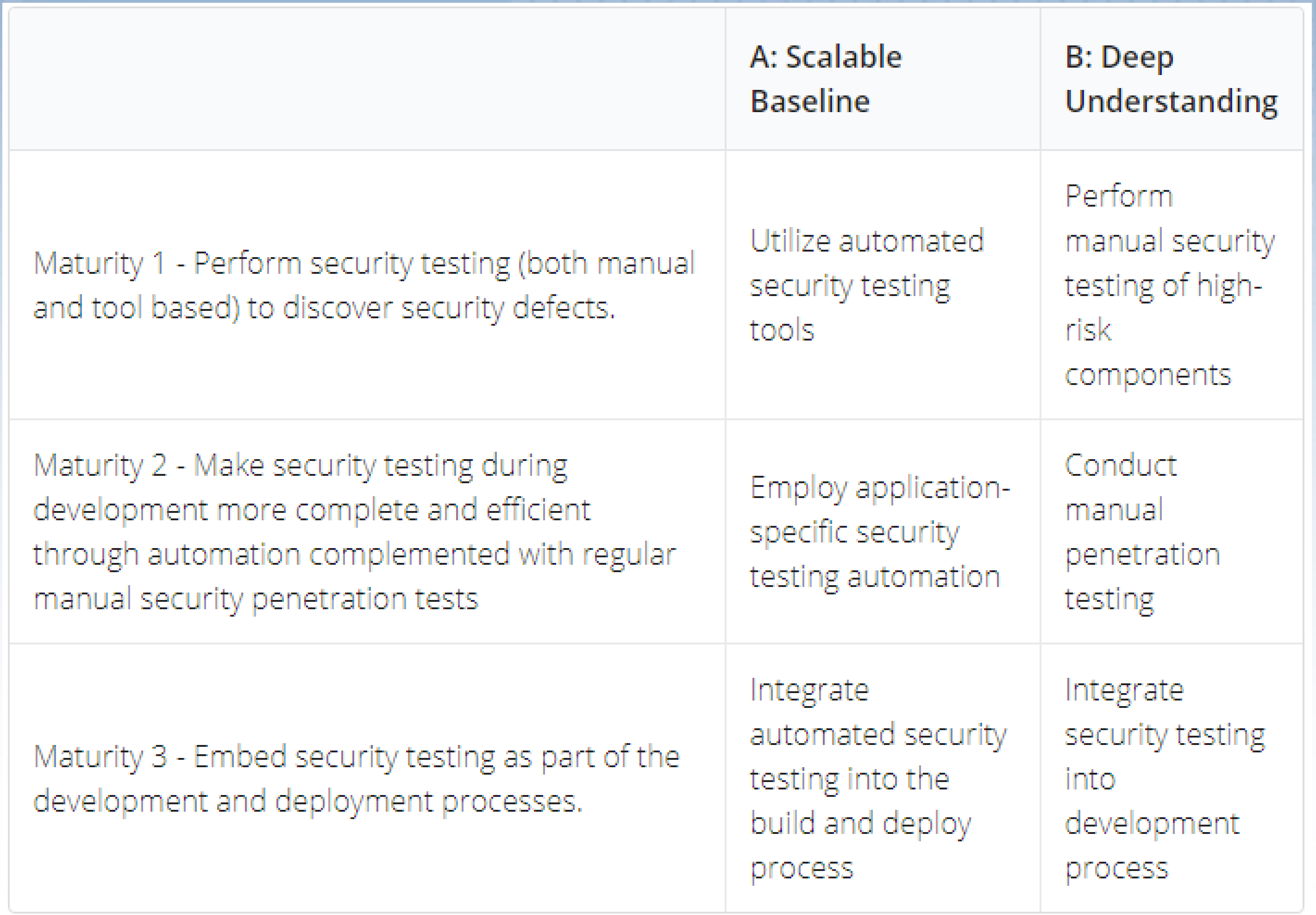
- This is where the typical SAST/DAST/IAST takes place
- Manual testing can achieve more intelligent and intricate verification
- Detected defects will require validation, risk analysis & recommendations to fix

	Manual	Automated
Source Code		
Dynamic behavior		



Source: Gartner (July 2013)

Security Testing



	A: Scalable Baseline	B: Deep Understanding
Maturity 1 - Perform security testing (both manual and tool based) to discover security defects.	Utilize automated security testing tools	Perform manual security testing of high-risk components
Maturity 2 - Make security testing during development more complete and efficient through automation complemented with regular manual security penetration tests	Employ application-specific security testing automation	Conduct manual penetration testing
Maturity 3 - Embed security testing as part of the development and deployment processes.	Integrate automated security testing into the build and deploy process	Integrate security testing into development process

Assessment Exercise

- Use OWASP SAMM to evaluate the development practices in your own company
- Focus on *Verification* Business Functions
- Applicable to both Waterfall and Agile models
- Using questionnaires (toolbox)

Assessment wrap-up

- What’s your company’s score ?
- What’s the average scores for the group ?
- Any odd ratings ?

Operations
Business Function

Governance	Design	Implementation	Verification	Operations
• Strategy & Metrics • Policy & Compliance • Education & Guidance	• Threat Assessment • Security Requirements • Security Architecture	• Secure Build • Secure Deployment • Defect Management	• Architecture Assessment • Requirements Testing • Security Testing	• Incident Management • Environment Management • Operational Management

Incident Management

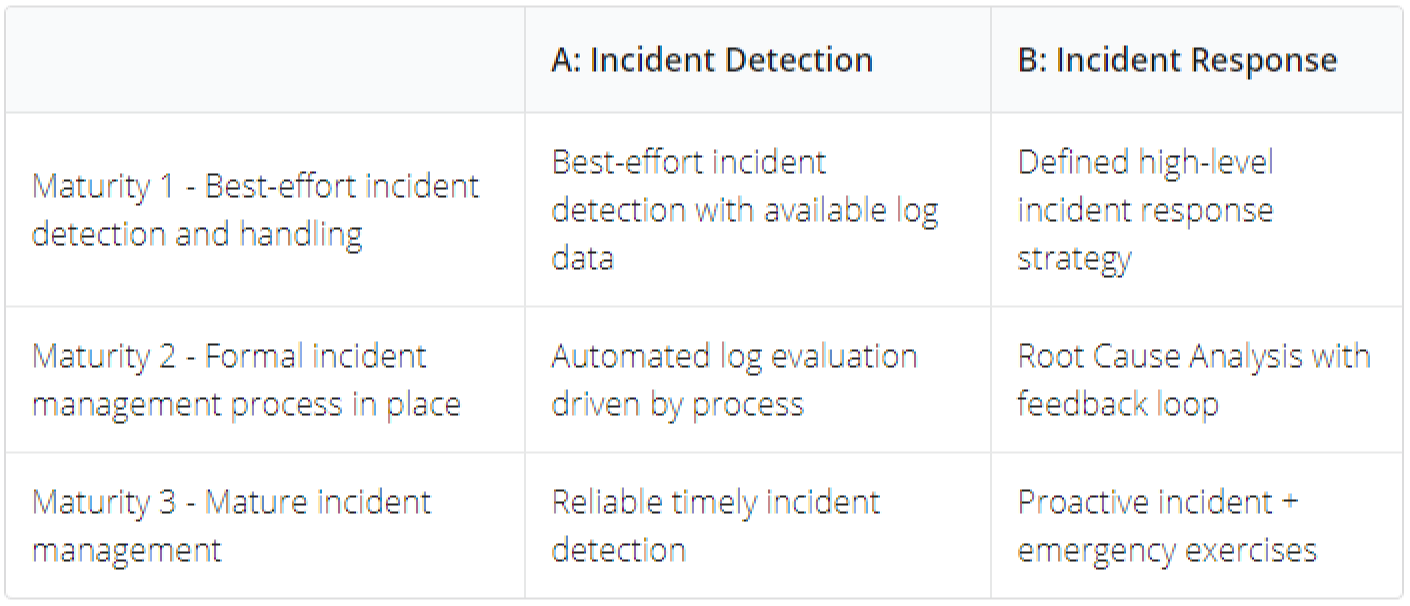
Prepare for WHEN, not IF!

Symptoms of malfunctioning SDLC

1. Examples of a security incidents:
   - successful DoS (Denial of Service) attack against a cloud application
   - application user accessing private data of another one by abusing a security vulnerability
   - attacker modifying the application source code
2. Have a capability in place to **detect** potential incidents
3. Make sure you can **respond** to detected incidents

4. Use vulnerability metrics and root-cause analysis to improve SDLC

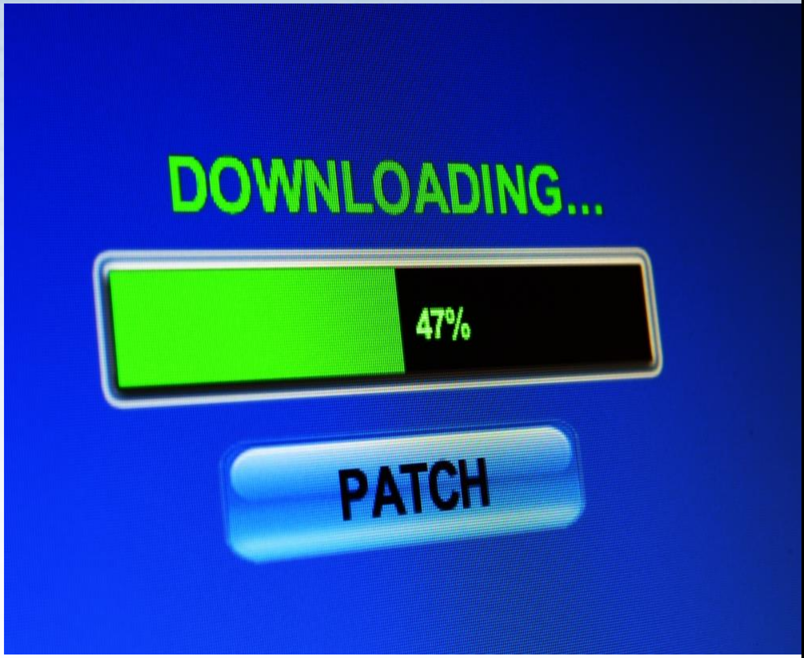
Incident Management



	A: Incident Detection	B: Incident Response
Maturity 1 - Best-effort incident detection and handling	Best-effort incident detection with available log data	Defined high-level incident response strategy
Maturity 2 - Formal incident management process in place	Automated log evaluation driven by process	Root Cause Analysis with feedback loop
Maturity 3 - Mature incident management	Reliable timely incident detection	Proactive incident + emergency exercises

Environment Management

1. Application security is not over once the application becomes operational. New security features and patches are continuously released until the technology stack you're using becomes obsolete.

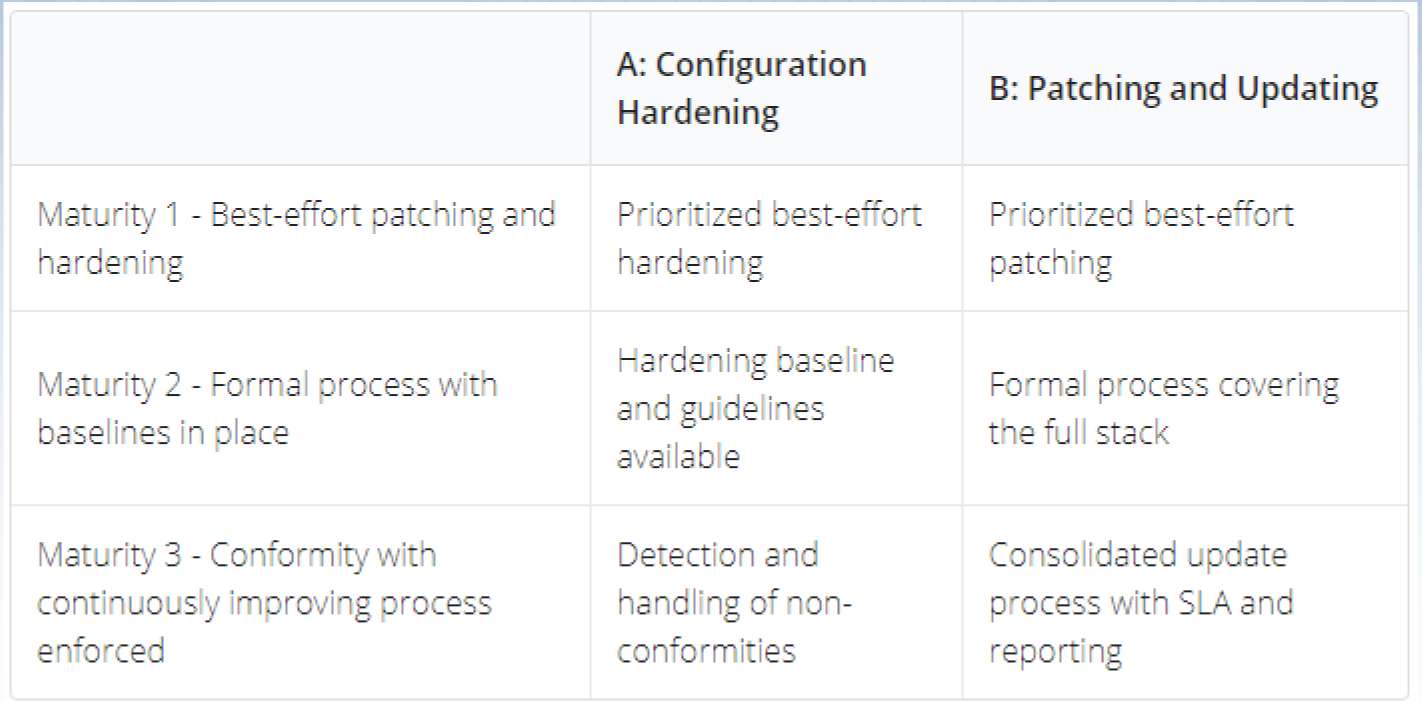


2. Pro-active **hardening** of the different technology components in the software environment by activing security features and removing unnecessary ones.

3. Installing **patches** to ensure technology components can withstand known security attacks.

   Consider virtual patching for temporary fixes

Environment Management



	A: Configuration Hardening	B: Patching and Updating
Maturity 1 - Best-effort patching and hardening	Prioritized best-effort hardening	Prioritized best-effort patching
Maturity 2 - Formal process with baselines in place	Hardening baseline and guidelines available	Formal process covering the full stack
Maturity 3 - Conformity with continuously improving process enforced	Detection and handling of non-conformities	Consolidated update process with SLA and reporting

Operational Management

1. This practice focuses on operational support activities required to maintain security throughout the product lifecycle.

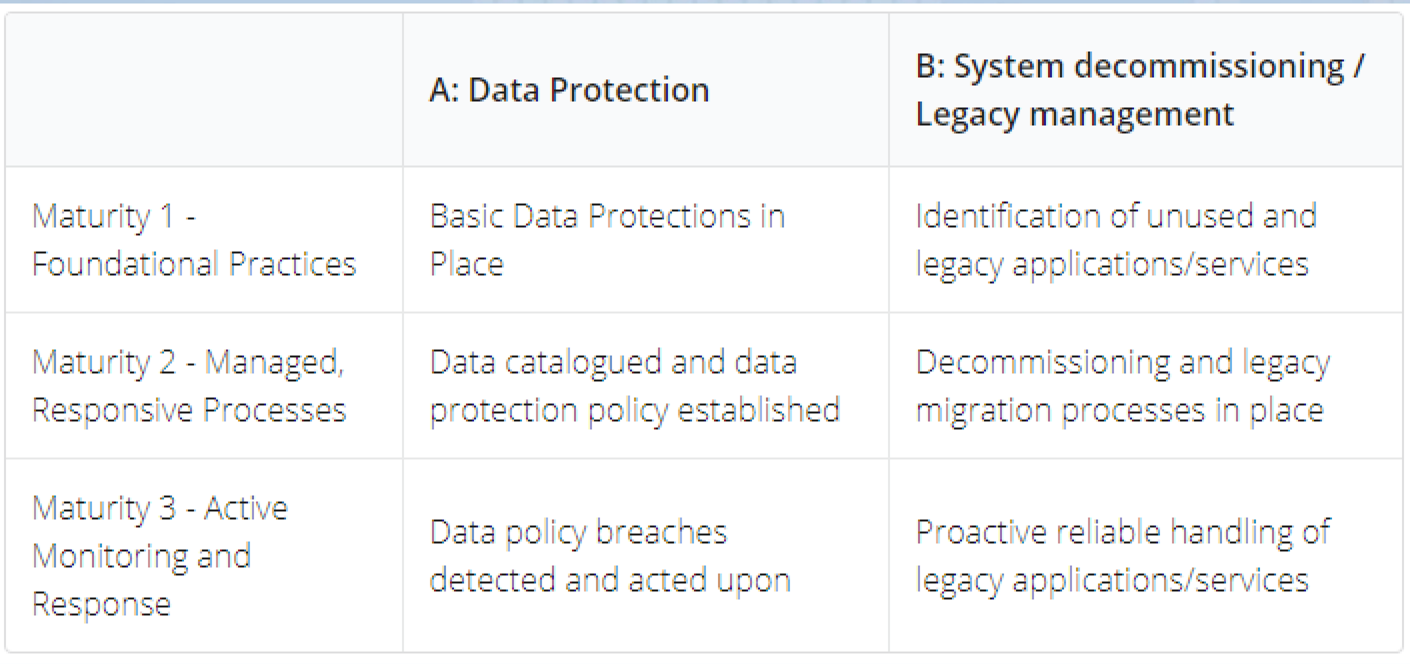
2. **Data** must be sufficiently **protected** in its different forms and environments to ensure a correct operation of the application

3. **Legacy** management ensures there are no loose ends, often forgotten, in the organisation which may form an easy target to attackers.

WE LOST ALL OF OUR COMPANY DATA AND OUR BACKUPS, TOO.
Dilbert.com DilbertCartoonist@gmail.com
SO I HACKED INTO OUR GOVERNMENT'S SECRET DATABASE WHERE THEY KEEP RECORDS OF EVERYTHING WE SAY OR DO AND GOT IT ALL BACK.
9-6-13 ©2013 Scott Adams, Inc./Dist. by Universal Uclick
I FEEL AS IF I SHOULD BE DOING SOMETHING NOW.
NAH. EVERYTHING IS WORKING FINE.

OWASP
Open Web Application Security Project

Operational Management



	A: Data Protection	B: System decommissioning / Legacy management
Maturity 1 - Foundational Practices	Basic Data Protections in Place	Identification of unused and legacy applications/services
Maturity 2 - Managed, Responsive Processes	Data catalogued and data protection policy established	Decommissioning and legacy migration processes in place
Maturity 3 - Active Monitoring and Response	Data policy breaches detected and acted upon	Proactive reliable handling of legacy applications/services

Assessment Exercise

- Use OWASP SAMM to evaluate the development practices in your own company
- Focus on *Operations* Business Functions
- Applicable to both Waterfall and Agile models
- Using questionnaires (toolbox)

Assessment wrap-up

- What’s your company’s score ?
- What’s the average scores for the group ?
- Any odd ratings ?

Setting the Target/Roadmap

1. Roadmap templates can provide direction for targets

   What type of company are you ?

2. Take into account the company’s risk appetite

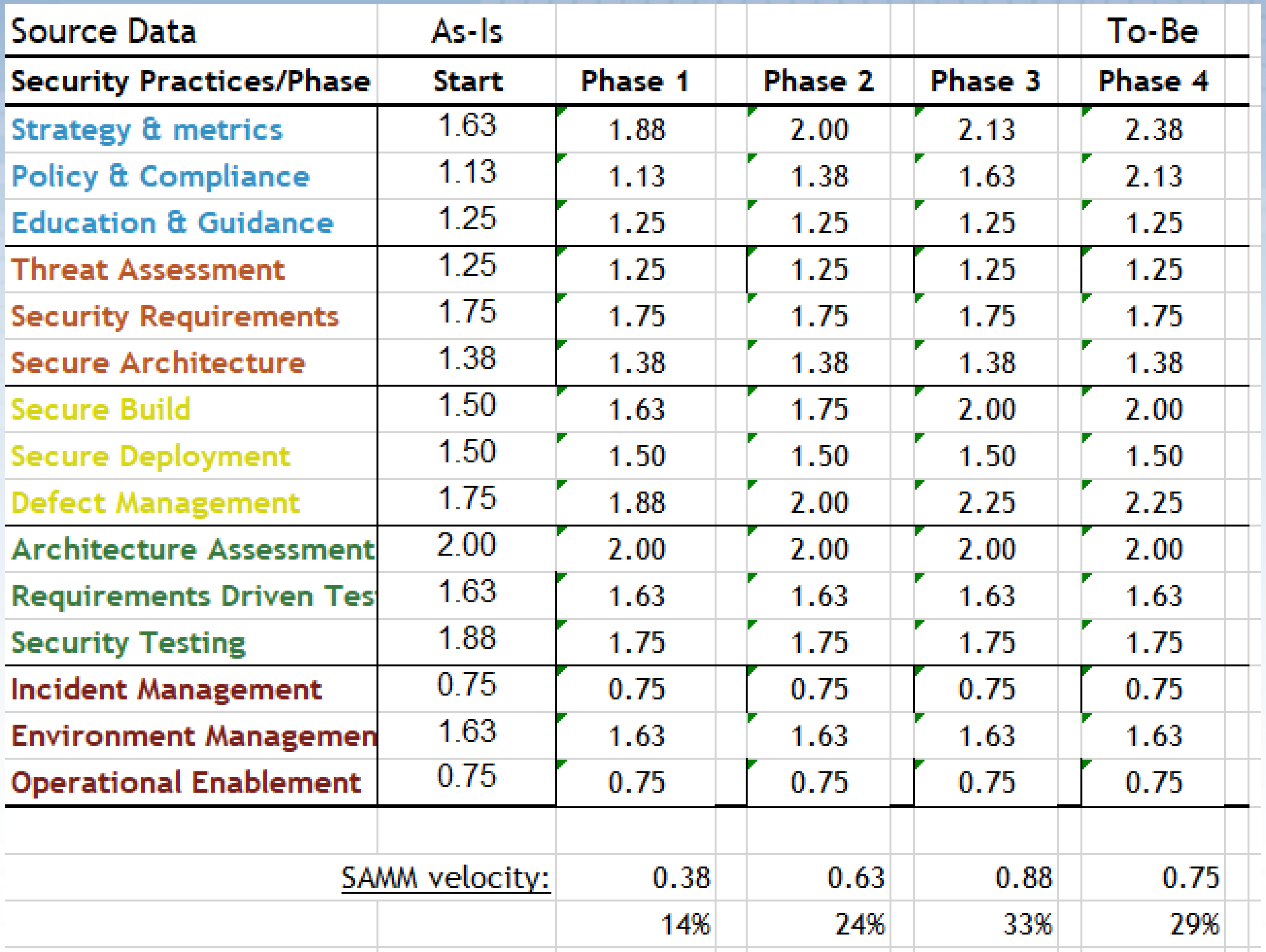
3. Only include activities where you see added value for the company, even for lower levels

4. OWASP SAMM activities have dependencies – use them !

5. Think about links with other practices in the company

   E.g., training, release management, …

Staged Roadmap



Source Data	As-Is				To-Be
Security Practices/Phase	**Start**	**Phase 1**	**Phase 2**	**Phase 3**	**Phase 4**
Strategy & metrics	1.63	1.88	2.00	2.13	2.38
Policy & Compliance	1.13	1.13	1.38	1.63	2.13
Education & Guidance	1.25	1.25	1.25	1.25	1.25
Threat Assessment	1.25	1.25	1.25	1.25	1.25
Security Requirements	1.75	1.75	1.75	1.75	1.75
Secure Architecture	1.38	1.38	1.38	1.38	1.38
Secure Build	1.50	1.63	1.75	2.00	2.00
Secure Deployment	1.50	1.50	1.50	1.50	1.50
Defect Management	1.75	1.88	2.00	2.25	2.25
Architecture Assessment	2.00	2.00	2.00	2.00	2.00
Requirements Driven Tes	1.63	1.63	1.63	1.63	1.63
Security Testing	1.88	1.75	1.75	1.75	1.75
Incident Management	0.75	0.75	0.75	0.75	0.75
Environment Managemen	1.63	1.63	1.63	1.63	1.63
Operational Enablement	0.75	0.75	0.75	0.75	0.75
SAMM velocity:		0.38	0.63	0.88	0.75
		14%	24%	33%	29%

Improvement Exercise

- Define a target for your company and the phased roadmap to get there

- Focus on the most urgent/heavy-impact practices first
  - Decide where you want to put the focus for the coming 3 years

- Try balancing the complexity and effort of the different step-ups

Conclusion Applying OWASP SAMM

Lightweight assessment of 15 security practices

Your thoughts:

- Representative summary ?
- New insights learned ?
- Anything not covered ?
- …

Today's Agenda

1. Introduction to SDLC and OWASP SAMM
2. Applying OWASP SAMM

   Methodology

   Assessment Governance

   Assessment Construction

   Assessment Verification

   Assessment Operations

   Setting Improvement Targets
3. **OWASP SAMM Tools**
4. OWASP SAMM Best Practices

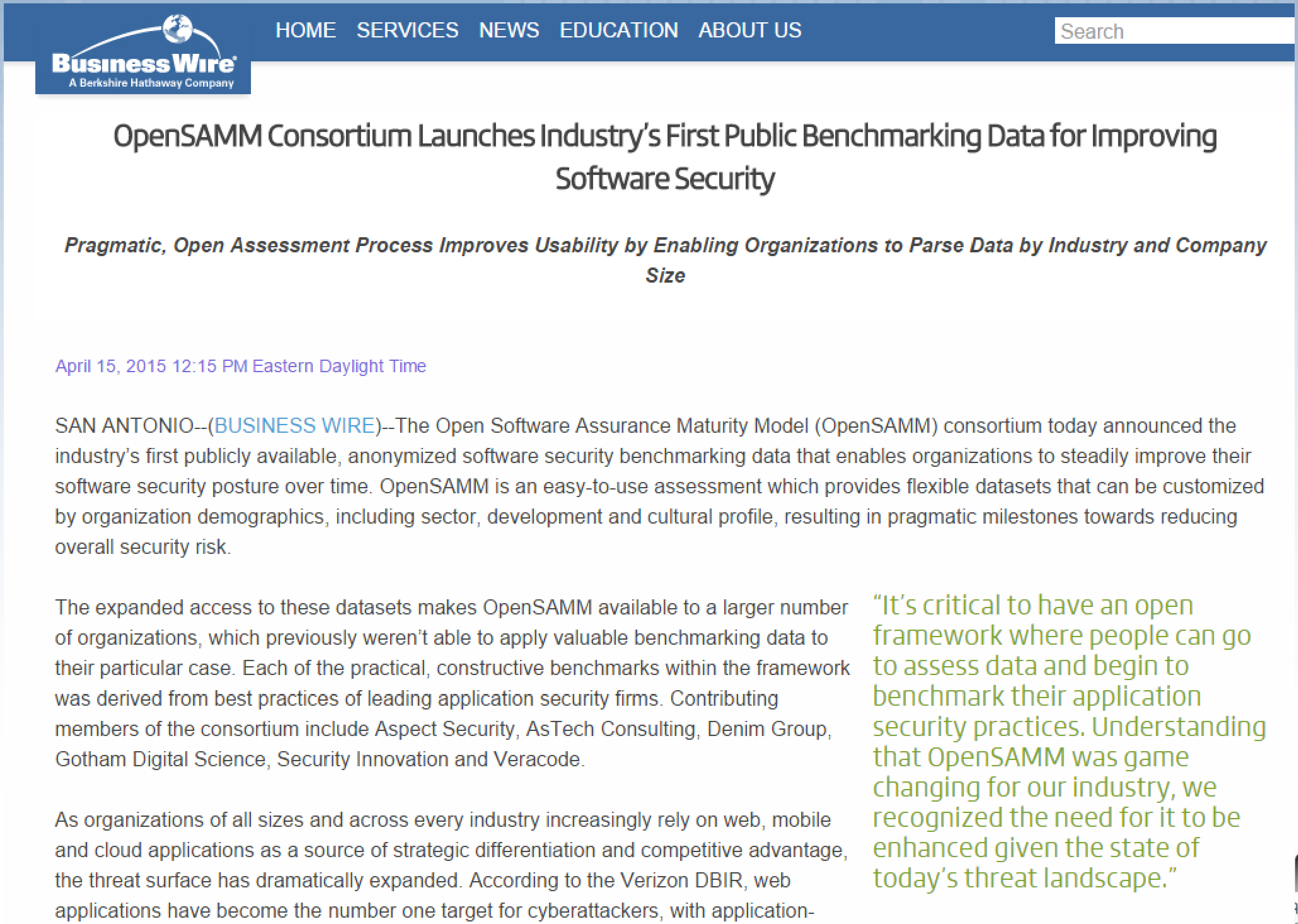
OWASP SAMM Tools

1. Translations of the OWASP SAMM model (Spanish, Japanese, *German,* Ukrainian, …)
2. Assessment questionnaire(s)
3. Roadmap chart template
4. Project plan template
5. OWASP SAMM-BSIMM mapping
6. **Benchmark Project**
7. Mappings to security standards

    ISO/IEC 27034, PCI, …

OWASP SAMM Assessment Toolbox

Benchmark Project


Business Wire
A Berkshire Hathaway Company
HOME SERVICES NEWS EDUCATION ABOUT US
Search

OpenSAMM Consortium Launches Industry's First Public Benchmarking Data for Improving Software Security

Pragmatic, Open Assessment Process Improves Usability by Enabling Organizations to Parse Data by Industry and Company Size

April 15, 2015 12:15 PM Eastern Daylight Time

SAN ANTONIO--(BUSINESS WIRE)--The Open Software Assurance Maturity Model (OpenSAMM) consortium today announced the industry's first publicly available, anonymized software security benchmarking data that enables organizations to steadily improve their software security posture over time. OpenSAMM is an easy-to-use assessment which provides flexible datasets that can be customized by organization demographics, including sector, development and cultural profile, resulting in pragmatic milestones towards reducing overall security risk.

The expanded access to these datasets makes OpenSAMM available to a larger number of organizations, which previously weren't able to apply valuable benchmarking data to their particular case. Each of the practical, constructive benchmarks within the framework was derived from best practices of leading application security firms. Contributing members of the consortium include Aspect Security, AsTech Consulting, Denim Group, Gotham Digital Science, Security Innovation and Veracode.

As organizations of all sizes and across every industry increasingly rely on web, mobile and cloud applications as a source of strategic differentiation and competitive advantage, the threat surface has dramatically expanded. According to the Verizon DBIR, web applications have become the number one target for cyberattackers, with application-

> "It's critical to have an open framework where people can go to assess data and begin to benchmark their application security practices. Understanding that OpenSAMM was game changing for our industry, we recognized the need for it to be enhanced given the state of today's threat landscape."

OWASP SAMM Benchmarking Overview

1. An open model for security benchmarking
2. Consortium of security companies working together
3. Confidentiality maintained between client & security company
4. Public data anonymized for benchmarking between teams, organizations

OWASP SAMM Benchmarking Benefits

1. Validate transformation plans
2. Supporting existing plans
3. Clients of various security companies can utilize platform
4. Find specific maturities in teams and organizations with varying granularity
5. Bring security maturity testing to the masses

150+ OWASP Projects

PROTECT

Tools: AntiSamy Java/:NET, Enterprise Security API (ESAPI), ModSecurity Core Rule Set Project

Docs: Development Guide, .NET, Ruby on Rails Security Guide, Secure Coding Practices - Quick Reference Guide

DETECT

Tools: JBroFuzz, Lice CD, WebScarab, Zed Attack Proxy

Docs: Application Security Verification Standard, Code Review Guide, Testing Guide, Top Ten Project

LIFE CYCLE

SAMM, WebGoat, Legal Project

https://www.owasp.org/index.php/Category:SAMM-Resources

Today's Agenda

1. Introduction to SDLC and OWASP SAMM
2. Applying OWASP SAMM

   Methodology

   Assessment Governance

   Assessment Construction

   Assessment Verification

   Assessment Operations

   Setting Improvement Targets
3. OWASP SAMM Tools
4. **OWASP SAMM Best Practices**

The importance of a Business Case

If you want your company to improve, management buy-in is crucial

⇒ You will need a business case to convince them

Typical arguments:

- Improved security quality
- Better cost efficiency
- Compliance
- Risk management
- Customer satisfaction
- Reputation management

Entry Points

Pick the weak spots that can demonstrate short-term ROI

Typical examples

- Awareness training
- Coding Guidelines
- External Pentesting

Success will help you in continuing your effort

Application categorization

Granularity !

Inter-Connectivity !

Use this to rationalize security effort (according to the application risk)

Communication & Support

Critical success factor !

Spreading the message – broad audience

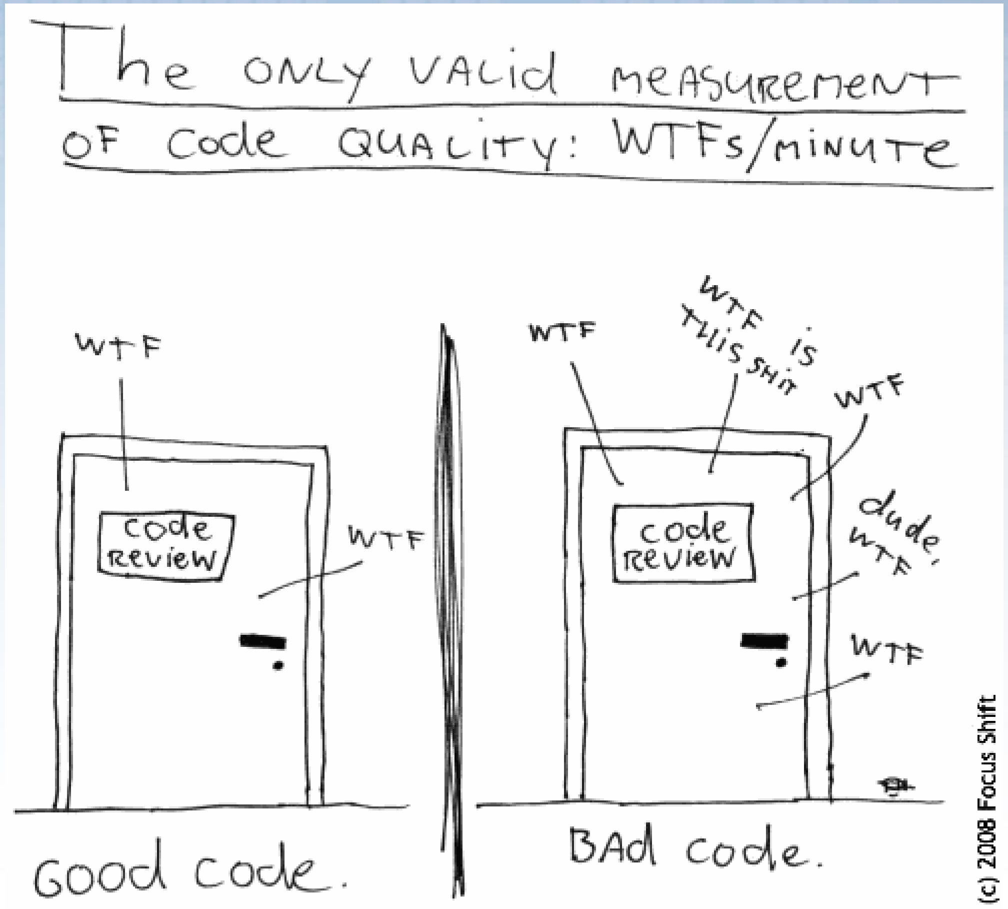
Setup a secure applications portal !

Regular status updates towards management

Monitoring & Metrics

Project vs. Enterprise dashboard

Manual vs. Automated data collection



Responsibilities

Core Security team

- Support vs. Responsible role

Security Satellite

- Analysts
- Architects
- Developers
- Operations
- Management

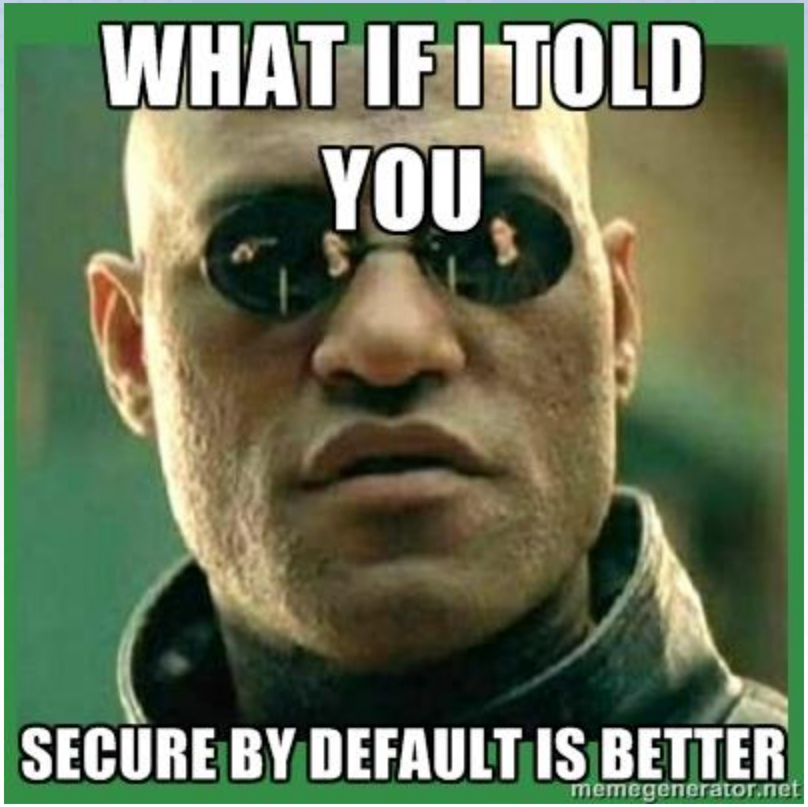
Formalized RACI will be a challenge

The Power of Default Security

Construct development frameworks that are secure by default

Minimizes work for developers

Will lower number of vulns.



Critical Success Factors

- Get initiative buy-in from stakeholders
- Adopt a risk-based approach
- Awareness / education is the foundation
- Integrate & automate security in your development / acquisition and deployment processes
- Measure: Provide management visibility

Conclusions

Developing secure software gets more and more complex

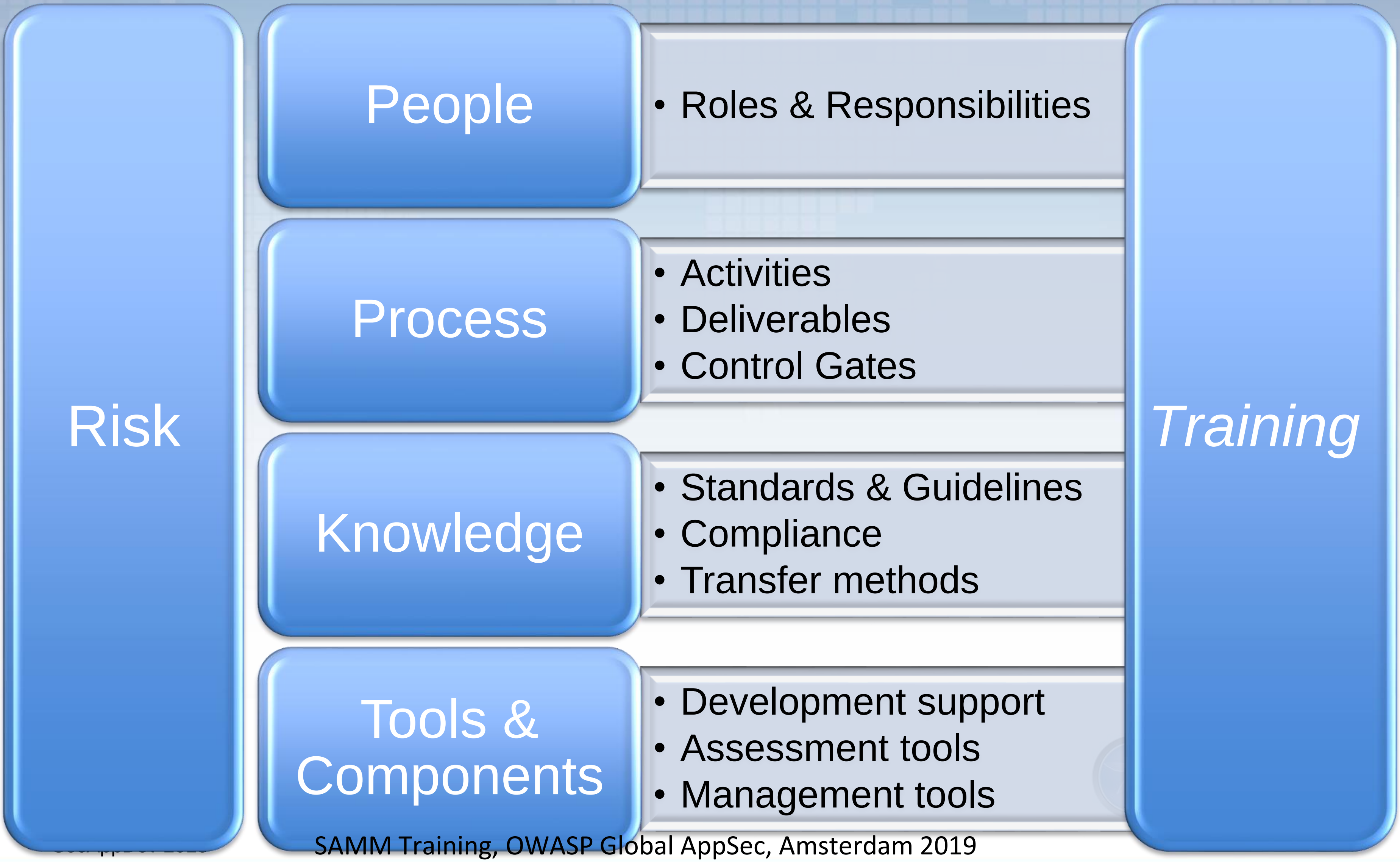
OWASP SAMM2 = global maturity foundation for software assurance, in line with current trends and practices

Applying OWASP SAMM =

- Assessment
- Roadmap
- (Continuous) Implementation

Be ready to face the organisational challenges that will pop up during the journey

SDLC Cornerstones (recap)



Risk

People
- Roles & Responsibilities

Process
- Activities
- Deliverables
- Control Gates

Knowledge
- Standards & Guidelines
- Compliance
- Transfer methods

Tools & Components
- Development support
- Assessment tools
- Management tools

Training

Thank you