



# OPENSAMM

## Software Assurance Maturity Model

<http://www.opensamm.org>

Dan Cornell

Denim Group [dan@denimgroup.com](mailto:dan@denimgroup.com)

OpenSAMM Core Team Member



Original Slides From Pravir Chandra, OpenSAMM Project Lead

[chandra@owasp.org](mailto:chandra@owasp.org)

# Agenda

- Review of existing secure SDLC efforts
- Understanding the model
- Applying the model
- Exploring the model's levels and activities
- SAMM and the real world

# By the end, you'll be able to...

- Evaluate an organization's existing software security practices
- Build a balanced software security assurance program in well-defined iterations
- Demonstrate concrete improvements to a security assurance program
- Define and measure security-related activities throughout an organization



# Review of existing secure SDLC efforts

# CLASP

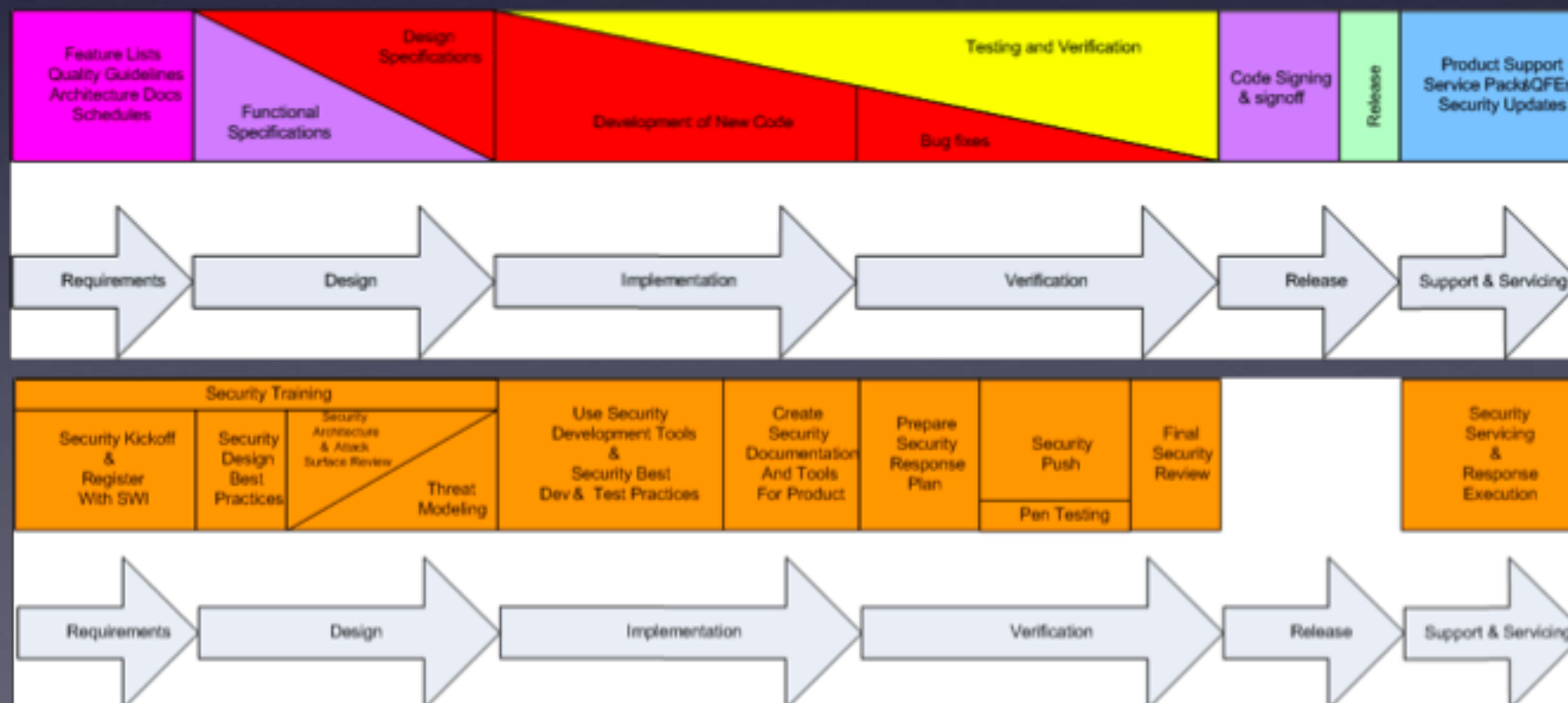
- Comprehensive, Lightweight Application Security Process
  - Centered around 7 AppSec Best Practices
  - Cover the entire software lifecycle (not just development)
- Adaptable to any development process
  - Defines roles across the SDLC
  - 24 role-based process components
  - Start small and dial-in to your needs





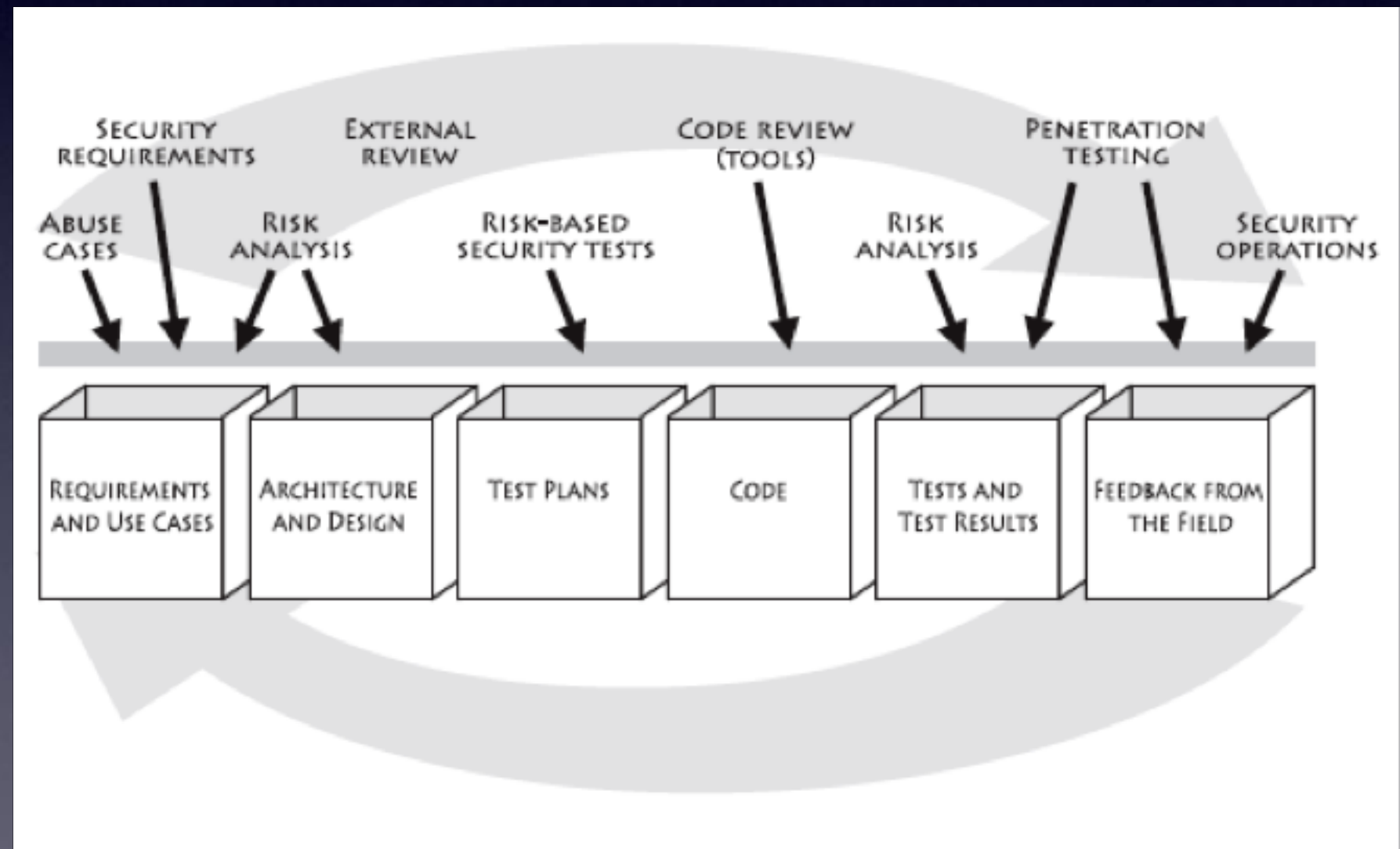
# Microsoft SDL

- Built internally for MS software
- Extended and made public for others
- MS-only versions since public release



# Touchpoints

- Gary McGraw's and Cigital's model



# Lessons Learned

- Microsoft SDL
  - Heavyweight, good for large ISVs
- Touchpoints
  - High-level, not enough details to execute against
- CLASP
  - Large collection of activities, but no priority ordering
- ALL: Good for experts to use as a guide, but hard for non-security folks to use off the shelf



# Drivers for 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 tailor 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

# Therefore, a viable model must...

- Define building blocks for an assurance program
  - Delineate all functions within an organization that could be improved over time
- Define how building blocks should be combined
  - Make creating change in iterations a no-brainer
- Define details for each building block clearly
  - Clarify the security-relevant parts in a widely applicable way (for any org doing software dev)



Understanding the model

# SAMM Business Functions

- Start with the core activities tied to any organization performing software development
- Named generically, but should resonate with any developer or manager



**Governance**



**Construction**



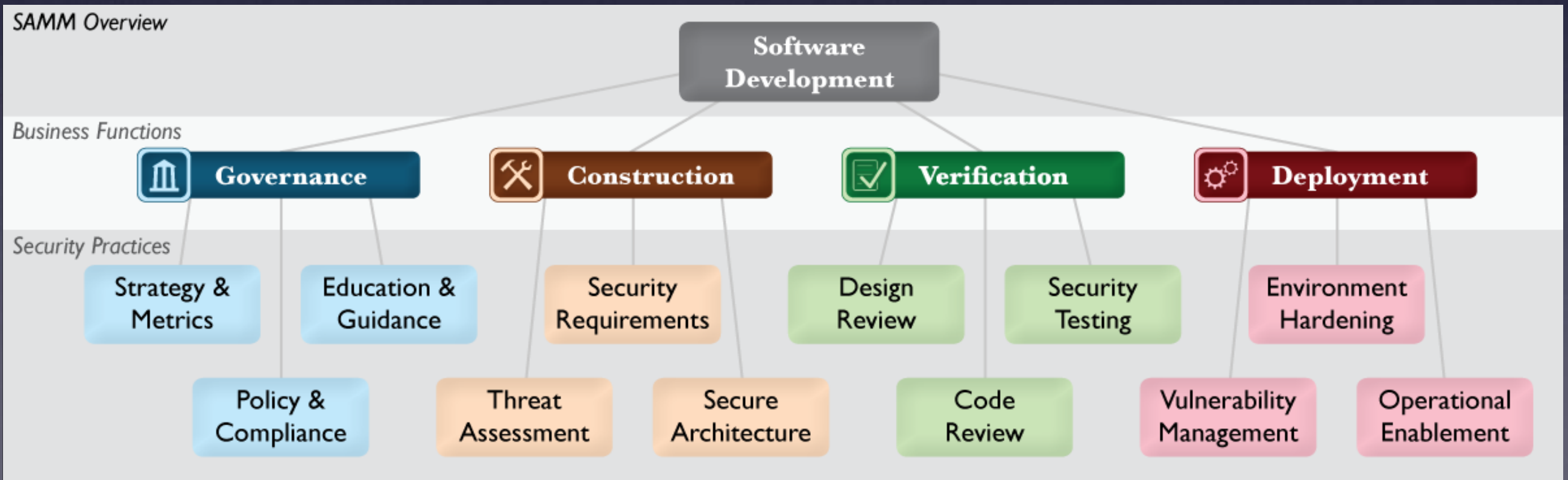
**Verification**



**Deployment**

# 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








# 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


# Check out this one...

Education & Guidance <span>...more on page 42</span>			
	 EG 1	 EG 2	 EG 3
OBJECTIVE	<b>Offer development staff access to resources around the topics of secure programming and deployment</b>	<b>Educate all personnel in the software life-cycle with role-specific guidance on secure development</b>	<b>Mandate comprehensive security training and certify personnel for baseline knowledge</b>
ACTIVITIES	<ul style="list-style-type: none"><li>A. Conduct technical security awareness training</li><li>B. Build and maintain technical guidelines</li></ul>	<ul style="list-style-type: none"><li>A. Conduct role-specific application security training</li><li>B. Utilize security coaches to enhance project teams</li></ul>	<ul style="list-style-type: none"><li>A. Create formal application security support portal</li><li>B. Establish role-based examination/certification</li></ul>

# Per Level, SAMM defines...

- Objective
- Activities
- Results
- Success Metric
- 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

SAMM / THE SECURITY PRACTICES - v1.0  
43

# Approach to iterative improvement

- Since the twelve Practices are each a maturity area, the successive Objectives represent the “building blocks” for any assurance program
- Simply put, improve an assurance program in phases by:
  1. Select security Practices to improve in next phase of assurance program
  2. Achieve the next Objective in each Practice by performing the corresponding Activities at the specified Success Metrics





Applying the model



# Conducting assessments

- SAMM includes assessment worksheets for each Security Practice

## Education & Guidance

Yes/No

- |   |  |
|---|--|
| ◆ Have most developers been given high-level security awareness training?                                     |  |
| ◆ Does each project team have access to secure development best practices and guidance?                       |  |
| ◆ Are most roles in the development process given role-specific training and guidance?                        |  |
| ◆ Are most stakeholders able to pull in security coaches for use on projects?                                 |  |
| ◆ Is security-related guidance centrally controlled and consistently distributed throughout the organization? |  |
| ◆ Are most people tested to ensure a baseline skill-set for secure development practices?                     |  |



EG 1



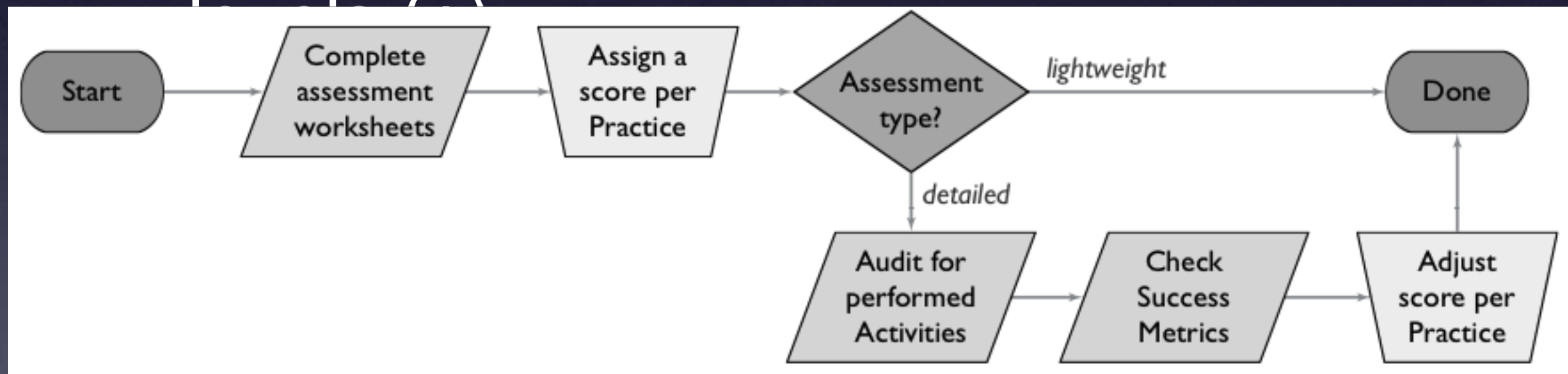
EG 2



EG 3

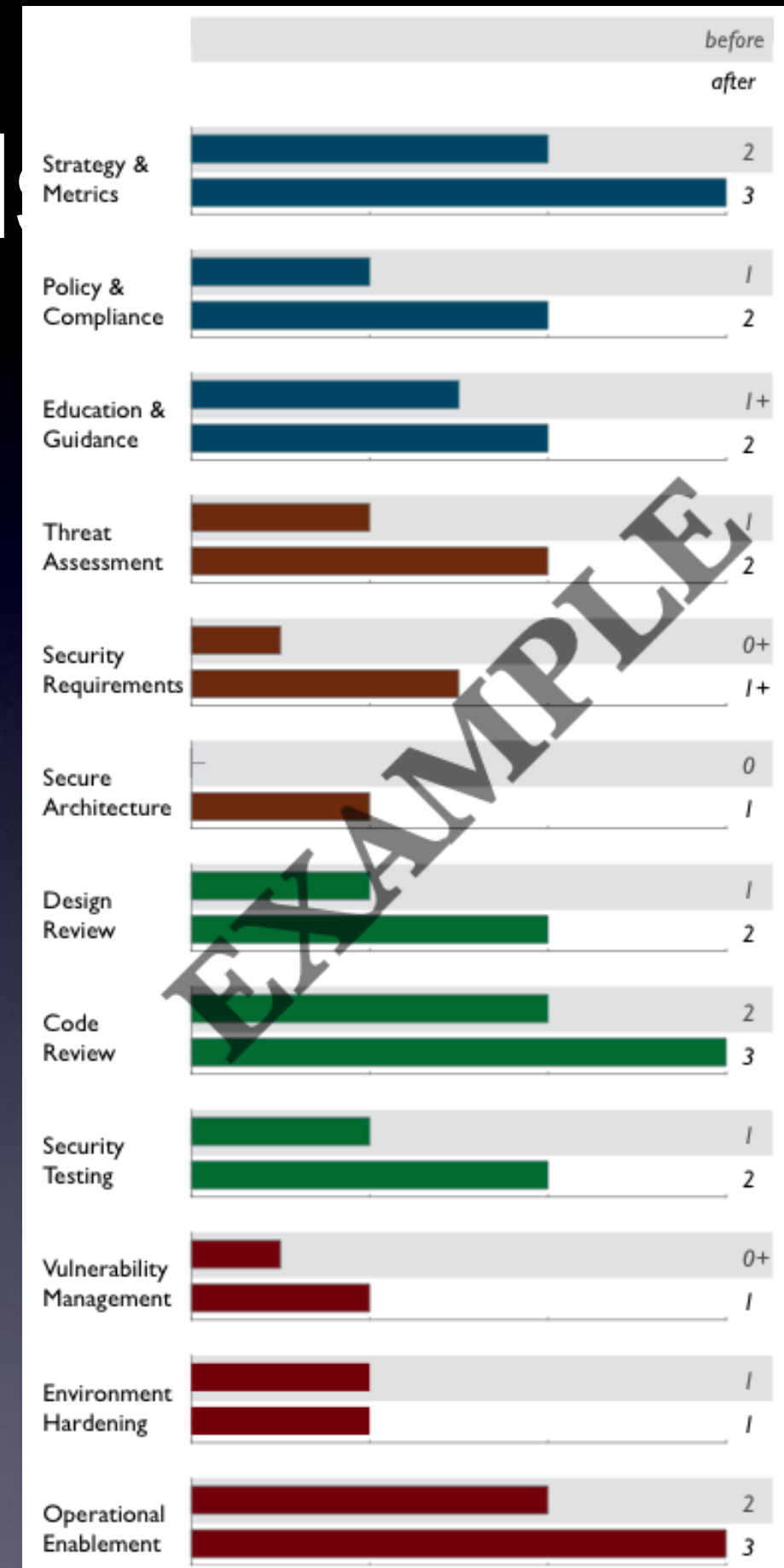
# Assessment process

- Supports both lightweight and detailed assessments
- Organizations may fall in between



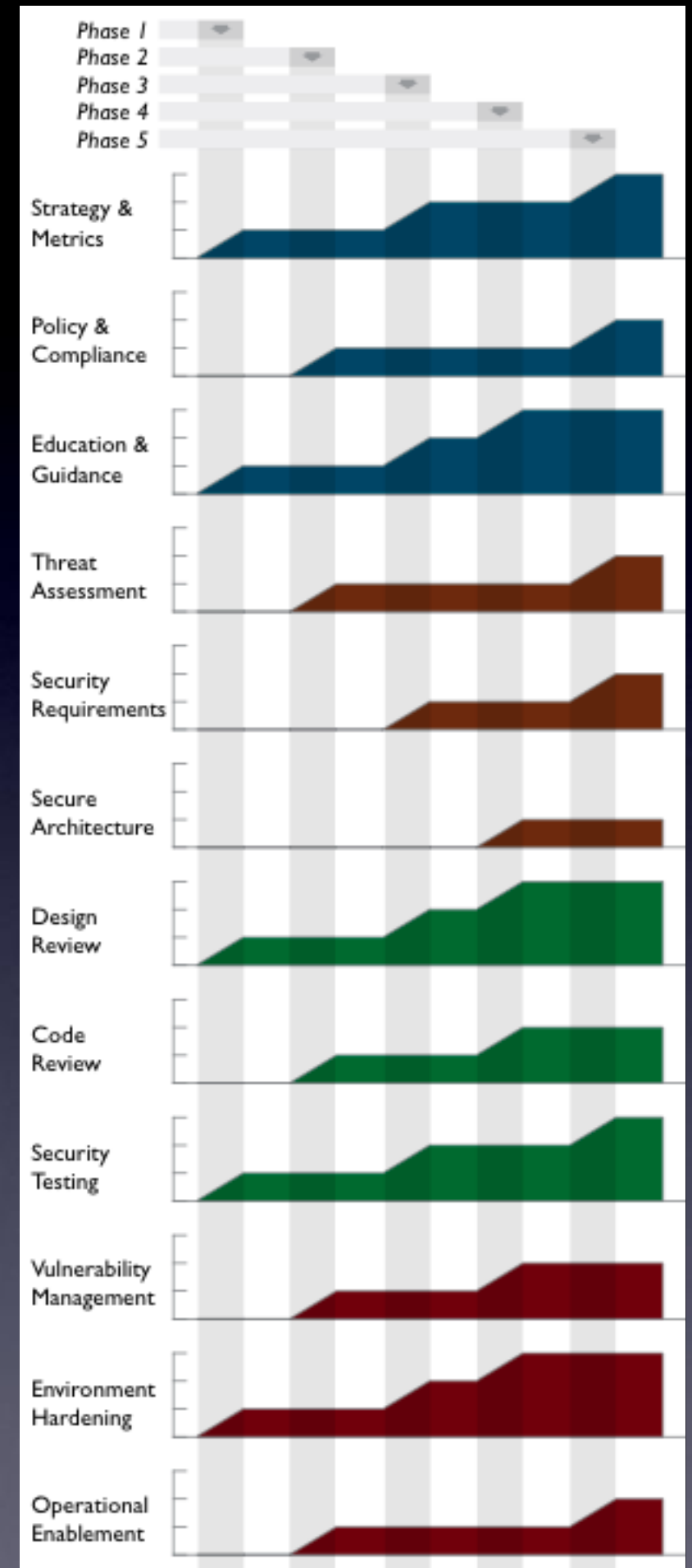
# Creating Scorecards

- Gap analysis
  - Capturing scores from detailed assessments versus expected performance levels
- Demonstrating improvement
  - Capturing scores from before and after an iteration of assurance program build-out
- Ongoing measurement
  - Capturing scores over consistent time frames for an assurance program that is already in place

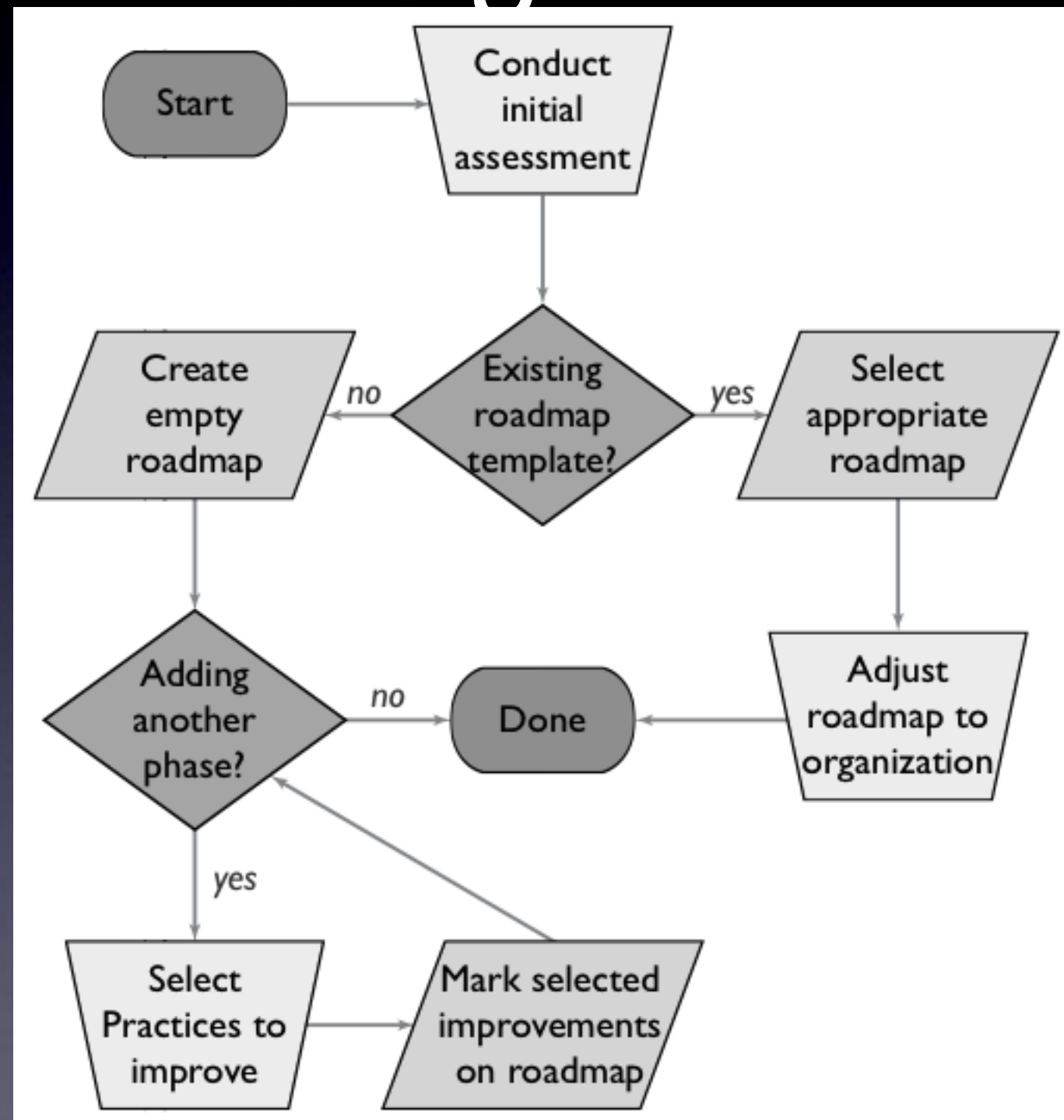


# Roadmap templates

- 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



# Building Assurance Programs





# Case Studies

- A full walkthrough with prose explanations of decision-making as an organization improves
- Each Phase described in detail
  - Organizational constraints
  - Build/buy choices
- One case study exists today, several more in progress using industry partners



Exploring the model's levels and  
activities

# The SAMM 1.0 release





SAMM and the real world

# SAMM history

- Beta released August 2008
  - 1.0 released March 2009
- Originally funded by Fortify
  - Still actively involved and using this model
- Released under a Creative Commons Attribution Share-Alike license
- Donated to OWASP and is currently an OWASP project



# Expert contributions

- Built based on collected experiences with 100's of organizations
- Including security experts, developers, architects, development managers, IT managers

## ***AUTHOR & PROJECT LEAD***

Pravir Chandra

## ***CONTRIBUTORS/REVIEWERS***

Fabio Arciniegas

Matt Bartoldus

Sebastien Deleersnyder

Jonathan Carter

Darren Challey

Brian Chess

Dinis Cruz

Justin Derry

Bart De Win

James McGovern

Matteo Meucci

Jeff Payne

Gunnar Peterson

Jeff Piper

Andy Steingruebl

John Steven

Chad Thunberg

Colin Watson

Jeff Williams

# Industry support

- Several more case studies underway



# The OpenSAMM Project

- <http://www.opensamm.org>
- Dedicated to defining, improving, and testing the SAMM framework
- Always vendor-neutral, but lots of industry participation
  - Open and community driven
- Targeting new releases every 6-12 months
- Change management process
  - SAMM Enhancement Proposals (SEP)

# Future plans

- Mappings to existing standards and regulations (many underway currently)
  - PCI, COBIT, ISO-17799/27002, ISM3, etc.
- Additional roadmaps where need is identified
- Additional case studies
- Feedback for refinement of the model
- Translations into other languages

# Other “modern” approachs

- Microsoft SDL Optimization Model
- Fortify/Cigital Building Security In Maturity Model (BSIMM)



# SDL Optimization Model

- Built by MS to make SDL adoption easier



# BSIMM

- Based on collected data from 9 large firms
- Recently expanded to 30

Governance	Intelligence	SSDL Touchpoints	Deployment
Strategy and Metrics	Attack Models	Architecture Analysis	Penetration Testing
Compliance and Policy	Security Features and Design	Code Review	Software Environment
Training	Standards and Requirements	Security Testing	Configuration Management and Vulnerability Management

# OpenSAMM Resources

- Nick Coblentz - SAMM Assessment Interview Template (xls/googledoc)
- Christian Frichot - SAMM Assessment Spreadsheet (xls)
- Colin Watson - Roadmap Chart Template (xls)
- Jim Weiler - MS Project Plan Template (mpp)
- Denim Group – Vulnerability Manager (web application)

# Quick re-cap on using SAMM

- Evaluate an organization's existing software security practices
- Build a balanced software security assurance program in well-defined iterations
- Demonstrate concrete improvements to a security assurance program
- Define and measure security-related activities throughout an organization

# Get involved

- Use SAMM and tell us about it
  - Blog, email, etc.
- Latest news at <http://www.opensamm.org>
- Sign up for the mailing list





# OPENSAMM

**Thanks for your time! Questions?**

<http://www.opensamm.org>

Dan Cornell  
Denim Group [dan@denimgroup.com](mailto:dan@denimgroup.com)  
OpenSAMM Core Team Member

Original Slides From Pravir Chandra, OpenSAMM Project  
[chandra@owasp.org](mailto:chandra@owasp.org)

