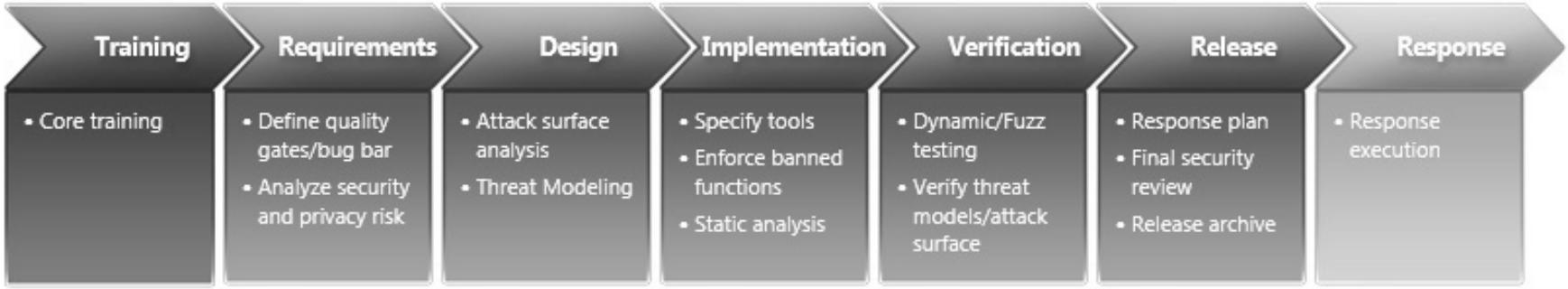
The Abridged Application Security Program

Max Caceres

Approaches to application security

- Microsoft-style (SDL)
  - Website, books, whitepapers
  - 80+ activities
- Building Security In Maturity Model (BSIMM)
  - 100+ activities


Training
• Core training
Requirements
• Define quality gates/bug bar
• Analyze security and privacy risk
Design
• Attack surface analysis
• Threat Modeling
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

Governance		Intelligence		SSDL Touchpoints		Deployment	
Activity	Observed	Activity	Observed	Activity	Observed	Activity	Observed
[SM1.1]	18	[AM1.1]	12	[AA1.1]	22	[PT1.1]	28
[SM1.2]	18	[AM1.2]	20	[AA1.2]	18	[PT1.2]	17
[SM1.3]	16	[AM1.3]	14	[AA1.3]	19	[PT2.1]	17
[SM1.4]	24	[AM1.4]	10	[AA1.4]	15	[PT2.2]	10
[SM1.5]	13	[AM2.1]	7	[AA2.1]	9	[PT2.3]	11
[SM2.1]	12	[AM2.2]	9	[AA2.2]	6	[PT3.1]	9
[SM2.2]	13	[AM2.3]	13	[AA2.3]	11	[PT3.2]	5
[SM2.3]	16	[AM2.4]	9	[AA3.1]	5		
[SM2.4]	19	[AM3.1]	2	[AA3.2]	3		
[SM3.1]	7	[AM3.2]	2				
[SM3.2]	4						
[CP1.1]	24	[SFD1.1]	29	[CR1.1]	10	[SE1.1]	11
[CP1.2]	24	[SFD1.2]	16	[CR1.2]	19	[SE1.2]	30
[CP1.3]	26	[SFD2.1]	18	[CR1.4]	20	[SE2.2]	16
[CP2.1]	13	[SFD2.2]	11	[CR2.2]	11	[SE2.3]	7
[CP2.2]	18	[SFD2.3]	10	[CR2.3]	8	[SE2.4]	13
[CP2.3]	13	[SFD3.1]	5	[CR2.4]	12	[SE3.2]	6
[CP2.4]	9	[SFD3.2]	10	[CR2.5]	11		
[CP2.5]	17			[CR3.1]	7		
[CP3.1]	4			[CR3.2]	1		
[CP3.2]	7			[CR3.3]	2		
[CP3.3]	5						
[T1.1]	24	[SR1.1]	22	[ST1.1]	21	[CMVM1.1]	21
[T1.2]	6	[SR1.2]	13	[ST1.2]	9	[CMVM1.2]	22
[T1.3]	5	[SR1.3]	12	[ST2.1]	18	[CMVM2.1]	18
[T1.4]	11	[SR1.4]	11	[ST2.2]	16	[CMVM2.2]	11
[T2.1]	14	[SR2.1]	10	[ST2.3]	5	[CMVM2.3]	11
[T2.2]	13	[SR2.2]	8	[ST3.1]	7	[CMVM3.1]	2
[T2.4]	14	[SR2.3]	13	[ST3.2]	10	[CMVM3.2]	4
[T2.5]	7	[SR2.4]	13	[ST3.3]	3		
[T3.1]	4	[SR2.5]	11	[ST3.4]	4		
[T3.2]	3	[SR3.1]	10				
[T3.3]	4						
[T3.4]	2						

Dangers of getting big too early

- Developing program takes a long time
- Executing will require a large investment upfront
- You'll have to do a lot of selling to get the organization onboard

The Abridged Security Program

- Focus on adding value from the start
- Translate traction and credibility into a program that grows organically into what your organization needs

MVP: Minimum Viable Program

Engage on just those activities (and not more) that can reduce security defects today

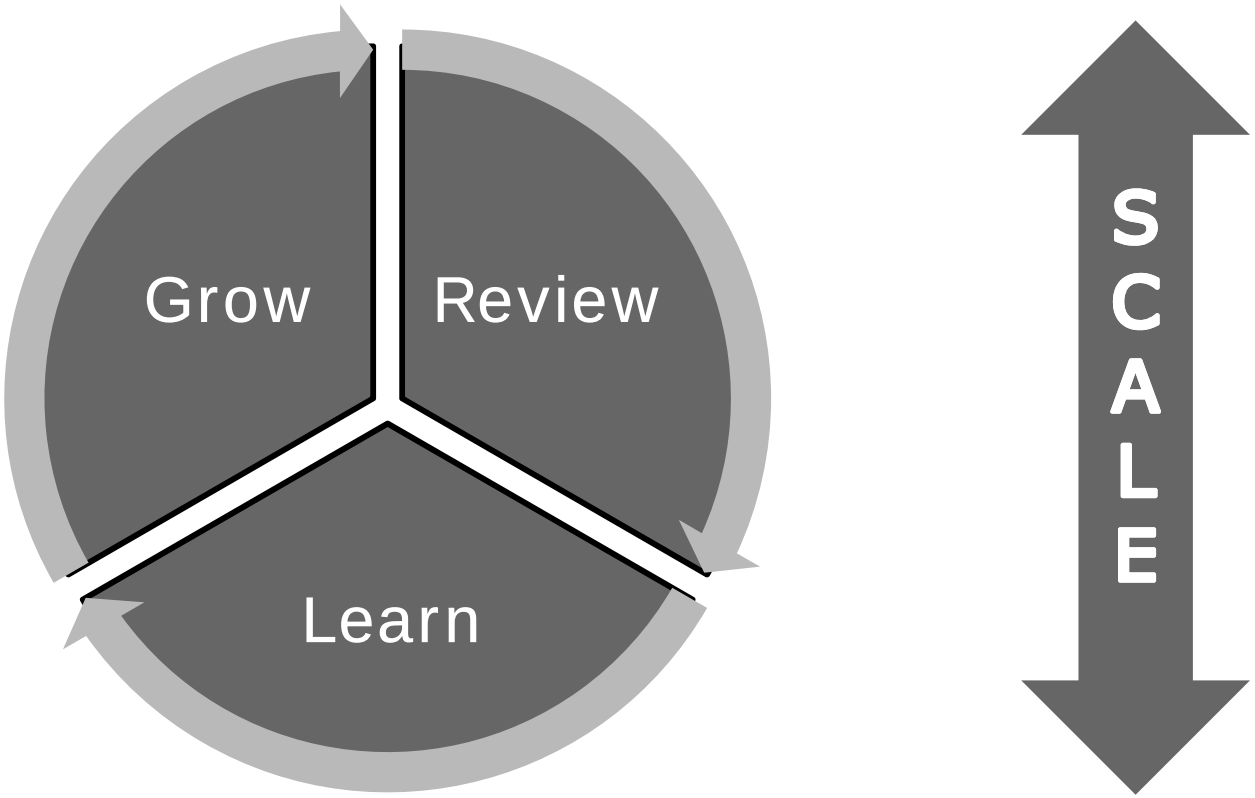
Why MVP?

- Avoid creating bureaucracy before absolutely necessary
- Avoid creating a complex program that you then have to retrofit to the organization
- Deploy limited resources wisely

Growing an appsec program

- Iterative process (*release early, release often*)
- Constantly show tangible results
- Develop program as you go

Iterative process



Building block #1: the security review

- Benefits
  - Minimal requirements
  - Can be partially automated or outsourced
  - Produces actionable results
  - Opportunity to learn and gain credibility

Minimal process

1st document: review report

- Communicate findings and expectations clearly
- Key elements
  - Succinct description and remediation
  - Simple risk score scaled by app criticality
  - Risk waiver scale

Example security finding

- **Finding**: Authentication bypass
- **Severity**: High
- **Adjusted Risk Score**: 5
- **Description**: An unauthenticated attacker can execute an account transfer
- **Recommendation**: Ensure only authenticated and authorized users can transfer funds

Be selective and focus

- Look for sensitive applications about to complete a release cycle
- Focus on one review at a time
- Scope reviews to last 2 wks max
- Gain credibility by engaging with dev team throughout review

In source | Outsource
White box | Black box

- Use internal source code review to learn about development organization and grow program organically
- Use external penetration testers for highly critical applications with public exposure

Learn from your customers

- Adjust risk scoring model
- Learn about your firm's development process
- Learn about your firm's technology choices
- Identify your future advocates

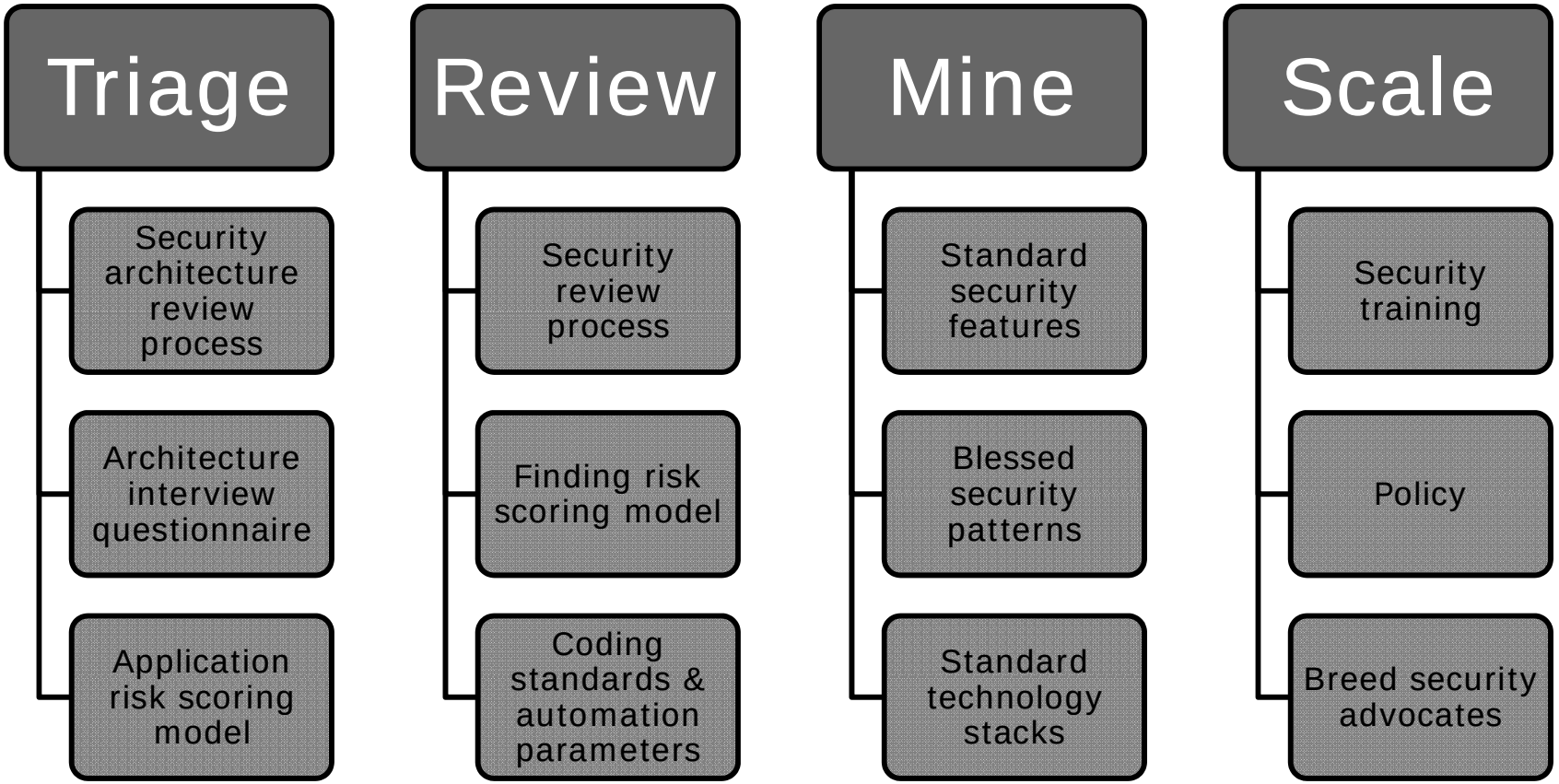
Mine common security components

- Ideal candidates are authentication, authorization, encryption, logging
- The firm already has code that does this. Find it, review it, then make it the standard
- You don’t have to own it

Apply leverage

- Look at automated tools, staff augmentation, and training
- Standardize on key security components
- Institutionalize security architecture review
- Breed developer advocates and delegate

Develop program as you go



Practices to avoid early-on

- Coding standards & policy
- Complex taxonomies and vulnerability lists
- Gate application deployment
- Delegating reviews to developers

Benefits of the abridged program

- You'll be reducing security defects from day one
- If you truly added value dev teams will come to you directly
- Resource requirements will grow with the program, but you'll have traction and supporters to help you get funding

Questions