The secure Agile Development - Microsoft Agile + SDL

Sept, 2010

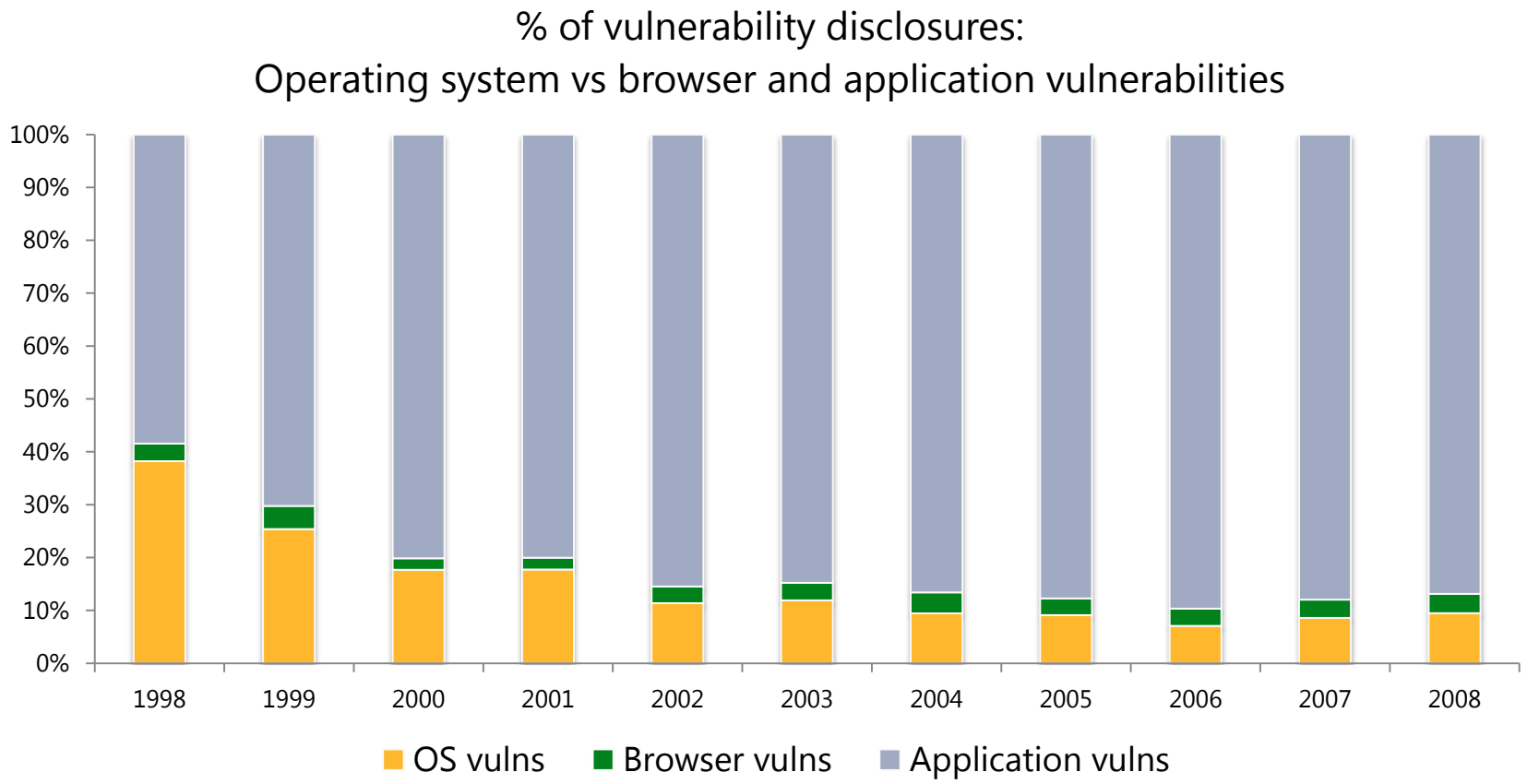
Lei Xu

Agenda

- Why secure?
- Origins of the Microsoft SDL
- Simplified SDL in detail
- SDL application to existing development methodologies
- Will SDL fit in Agile? – How?
- MSF Agile + SDL Process Template
- Resources for Development Organizations

SSW

Attacks focus on applications

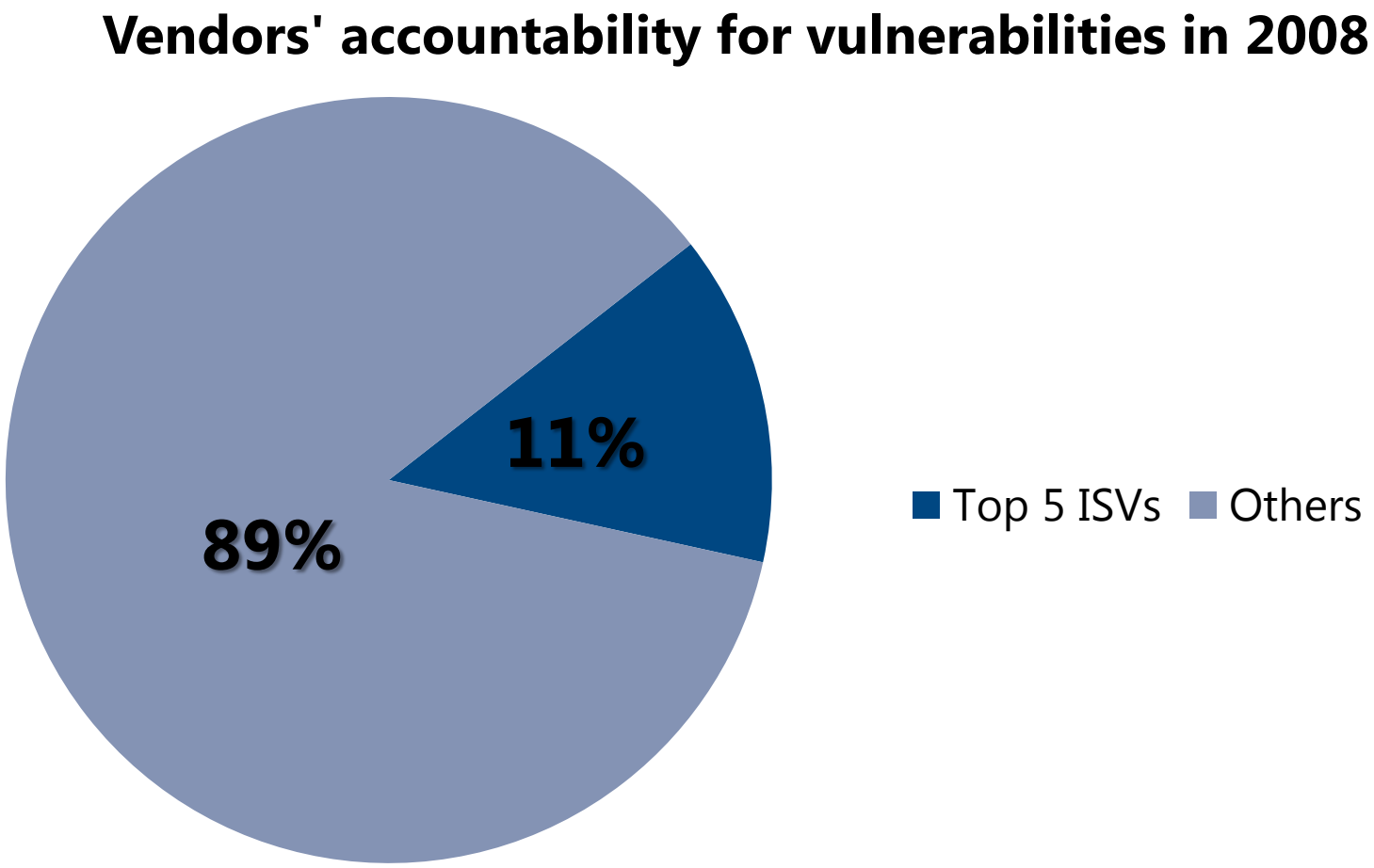


Calculated from the Microsoft Security Intelligence Report V6

90% of vulnerabilities are remotely exploitable

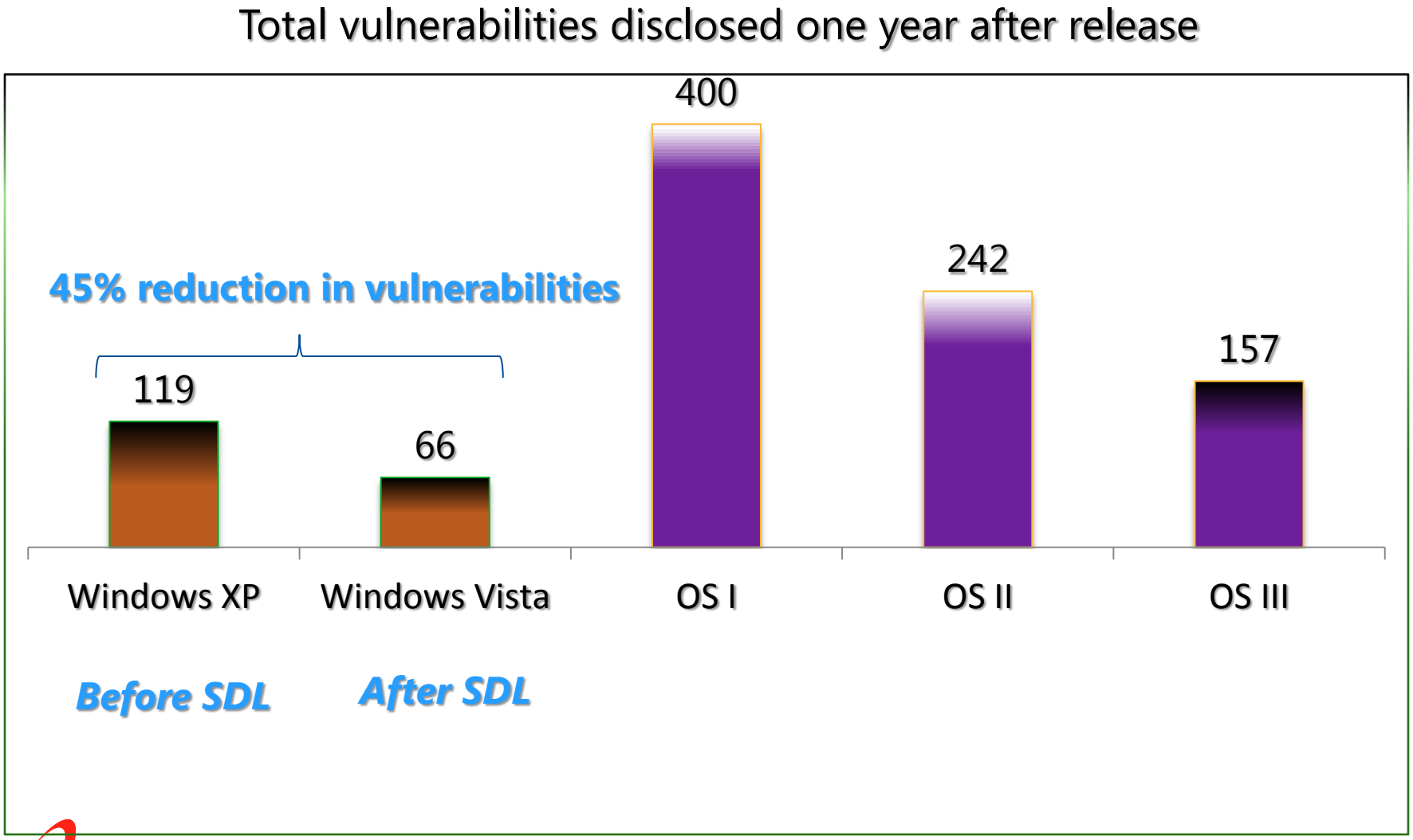
Sources: IBM X-Force, 2008

Most vulnerabilities are in smaller organizations' applications



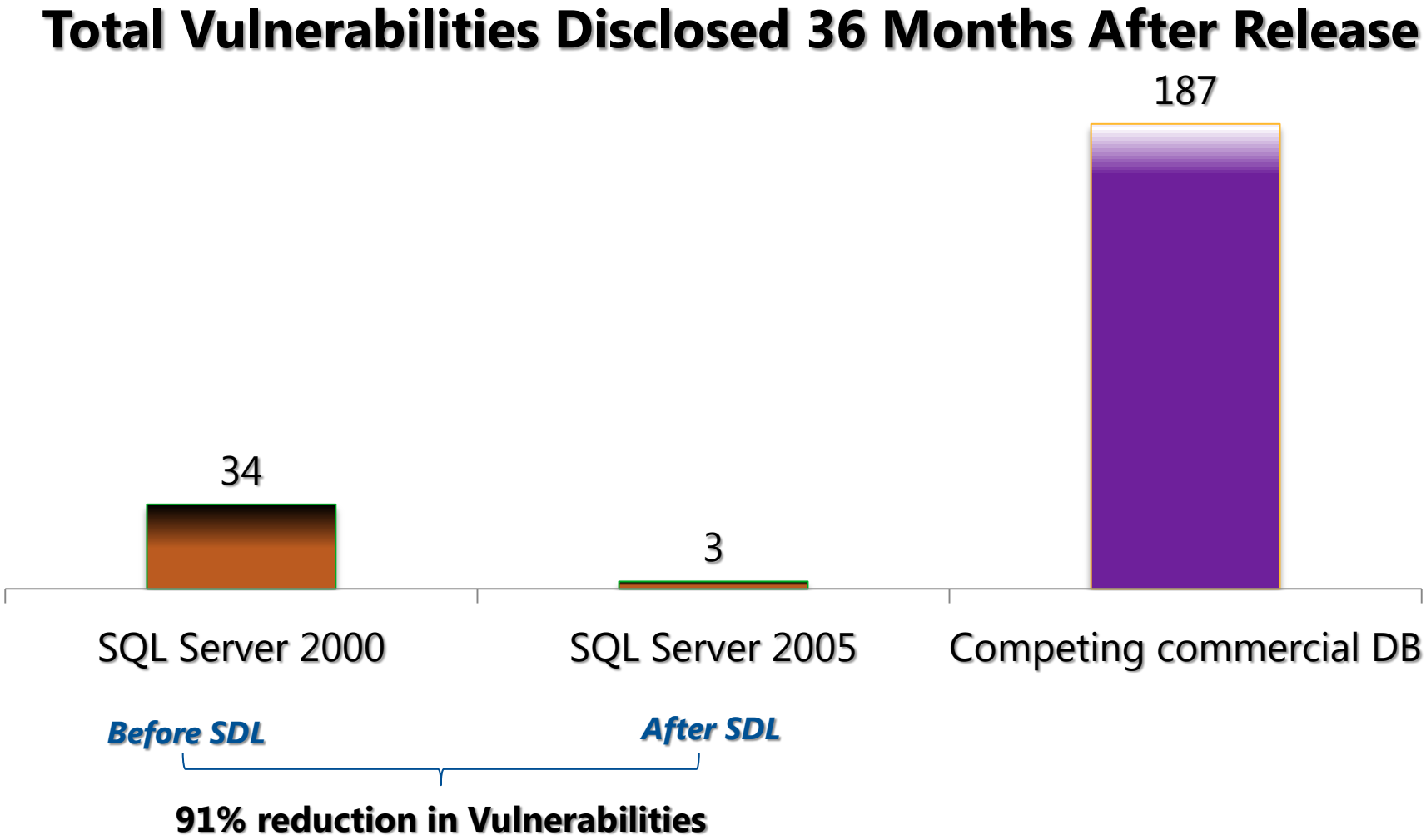
Sources: IBM X-Force 2008 Security Report

Windows: 45% reduction of vulnerabilities



Source: *Windows Vista One Year Vulnerability Report, Microsoft Security Blog 23 Jan 2008*

Microsoft SDL and SQL Server



Sources: Analysis by Jeff Jones (Microsoft technet security blog)

Origins of the Microsoft SDL…

Security Timeline at Microsoft…

2002-2003

- Bill Gates writes "Trustworthy Computing" memo early 2002
- "Windows Security Push" for Windows Server 2003
- Security push and FSR extended to other products

2004

- Microsoft Senior Leadership Team agrees to require SDL for all products that:
  - Are exposed to meaningful risk and/or
  - Process sensitive data

2005-2007

- SDL is enhanced
  - "Fuzz" testing
  - Code analysis
  - Crypto design requirements
  - Privacy
  - Banned APIs
  - and more…
- Windows Vista is the first OS to go through full SDL cycle

Now

- Optimize the process through feedback, analysis and automation
- Evangelize the SDL to the software development community:
  - SDL Process Guidance
  - SDL Optimization Model
  - SDL Pro Network
  - SDL Threat Modeling Tool
  - SDL Process Templates for VSTS

Common Misconceptions about SDL

"…only for Windows"

- *Appropriate for non-Microsoft platforms*
- *Based on proven, generally accepted security practices*

"…for shrink-wrapped products"

- *Also covers Line of Business (LOB) and online services development*

"…for waterfall or spiral development"

- *Agile methods are also supported*

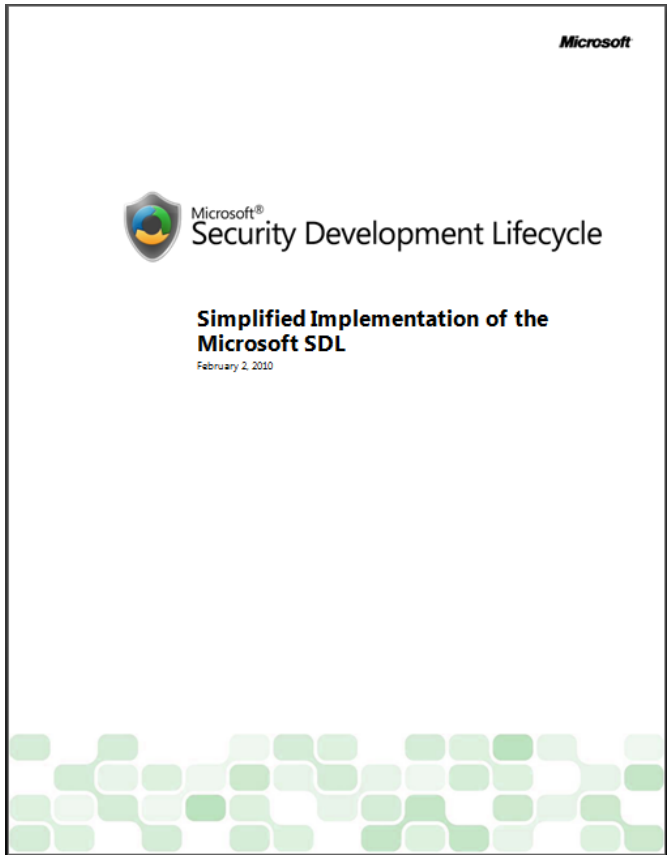
"…requires Microsoft tools"

- *Use the appropriate tools for the job – if those are MS tools great, if not, so be it.*

"…requires Microsoft-level resources to implement"

- *SDL as its applied at Microsoft != SDL for other development orgs.*

Simplified Implementation of the Microsoft SDL



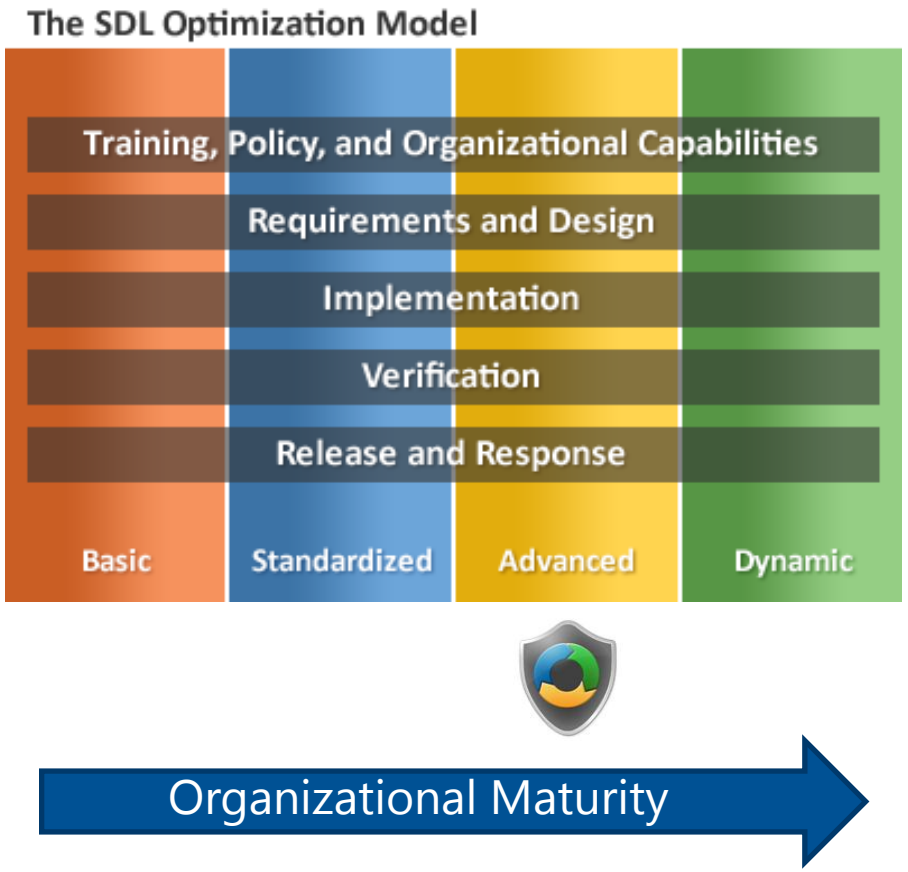
- *Non-proprietary*
- *Suitable for organizations of any size*
- *Platform agnostic*
- *Core elements based off the SDL process used at Microsoft*

Simplified SDL in Detail

SSW

Simplified SDL Core Concepts

- Maps to the "*Advanced*" level of the SDL Optimization Model
- Results in compliance with the spirit and practice of the SDL when followed completely
- Requires an above average degree of technical sophistication
- Prescriptive; seeks to avoid the "list of lists" approach



Pre-SDL Requirement: Security Training

Assess organizational knowledge – establish training program as necessary

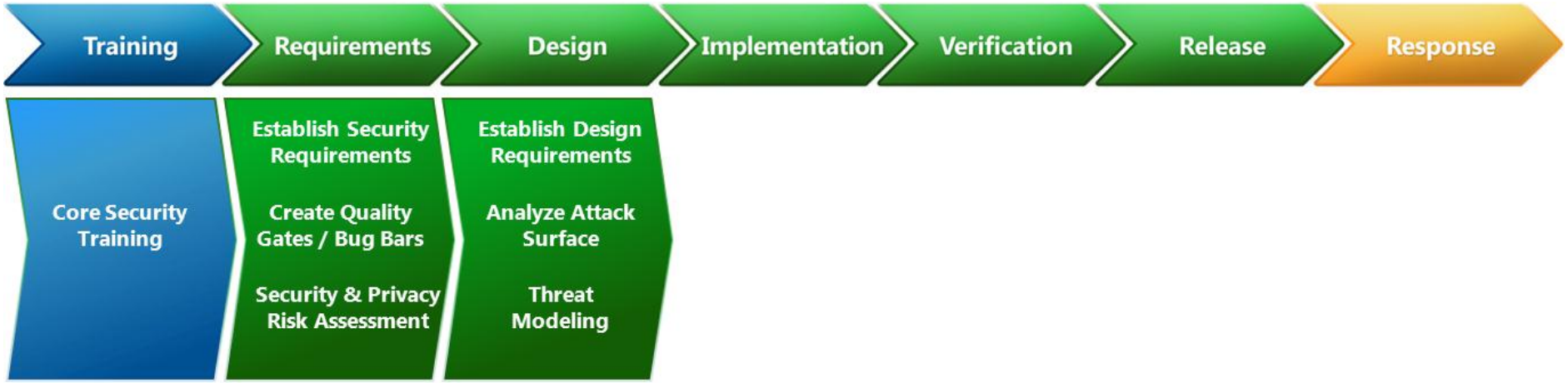
- Establish training criteria
  - Content covering secure design, development, test and privacy
- Establish minimum training frequency
  - Employees must attend *n* classes per year
- Establish minimum acceptable group training thresholds
  - Organizational training targets (e.g. 80% of all technical personnel trained prior to product RTM)

Phase One: Requirements

Opportunity to consider security at the outset of a project

- Establish Security Requirements
  - Project wide requirements – security leads identified, security bug tracking process mandated, architectural requirements set given the planned operational environment
- Create Quality Gates / Bug Bars
  - Minimum performance and quality criteria for each stage and for the project as a whole,
- Security and Privacy Risk Assessment
  - Risk assessment performed to determine critical components for the purposes of deep security and privacy review

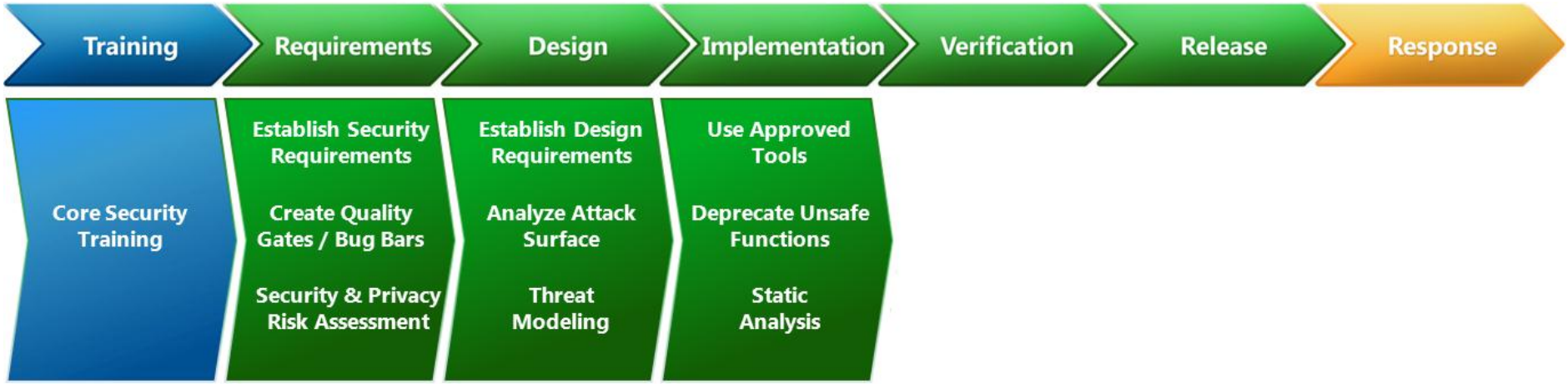
Phase Two: Design



Define and document security architecture, identify security critical components

- Establish Design Requirements
  - Required activities which include creation of design specifications, analysis of proposed security technologies (e.g. crypto requirements) and reconciliation of plans against functional specs.
- Analyze Attack Surface
  - Defense in depth strategies employed – use of layered defenses used to mitigate severity.
- Threat Modeling
  - Structured, component-level analysis of the security implications of a proposed design.

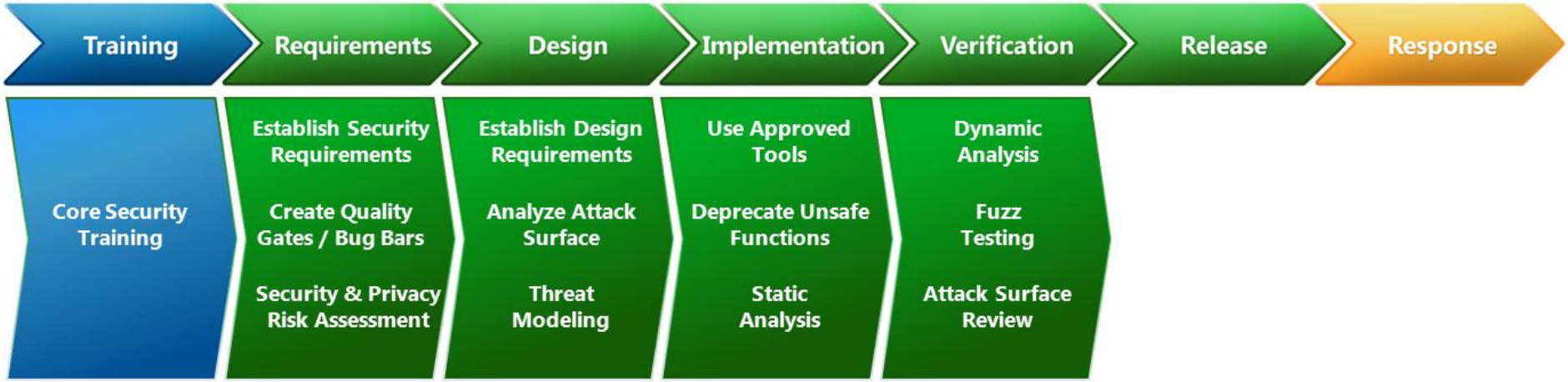
Phase Three: Implementation



Determine processes, documentation and tools necessary to ensure secure development

- Use approved tools
  - Approved list for compilers, security test tools, switches and flags; enforced project wide.
- Deprecate Unsafe Functions
  - Ban of unsafe functions, APIs, when using native (C/C++) code.
- Static Code Analysis
  - Scalable in-depth code review, augmentation by other methods as necessary to address weaknesses in static analysis tools.

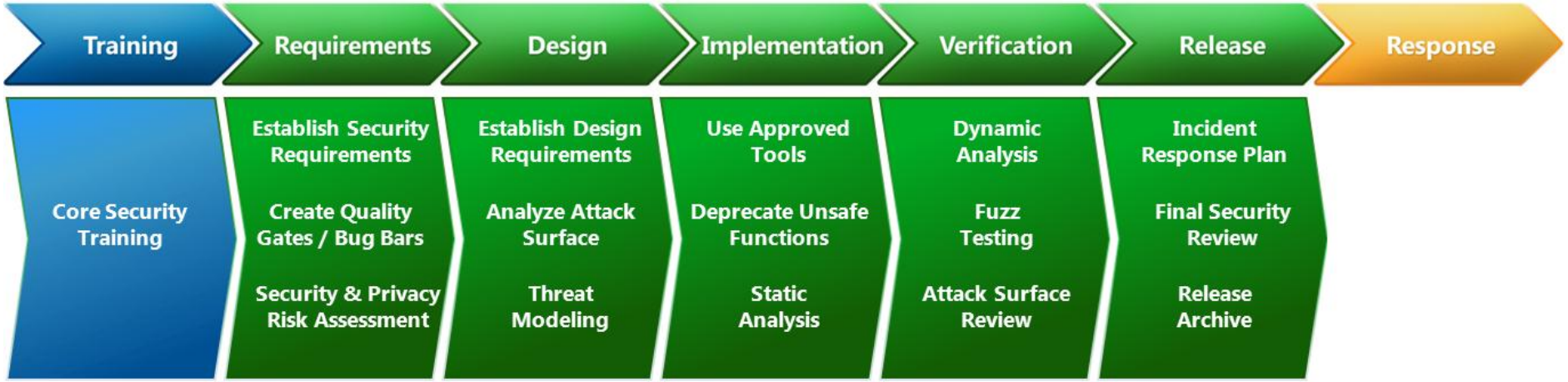
Phase Four: Verification



Verification of SDL security and privacy activities performed earlier in the process

- Dynamic Analysis
  - Runtime verification and analysis of programs to identify critical security problems
- Fuzz Testing
  - Specialized dynamic analysis technique used to deliberately cause program failure by injection of random, deliberately malformed inputs.
- Attack Surface / TM review
  - Re-review of attack surface and threat models when the program is “code complete” to ensure security assumptions and mitigations specified at design time are still relevant.

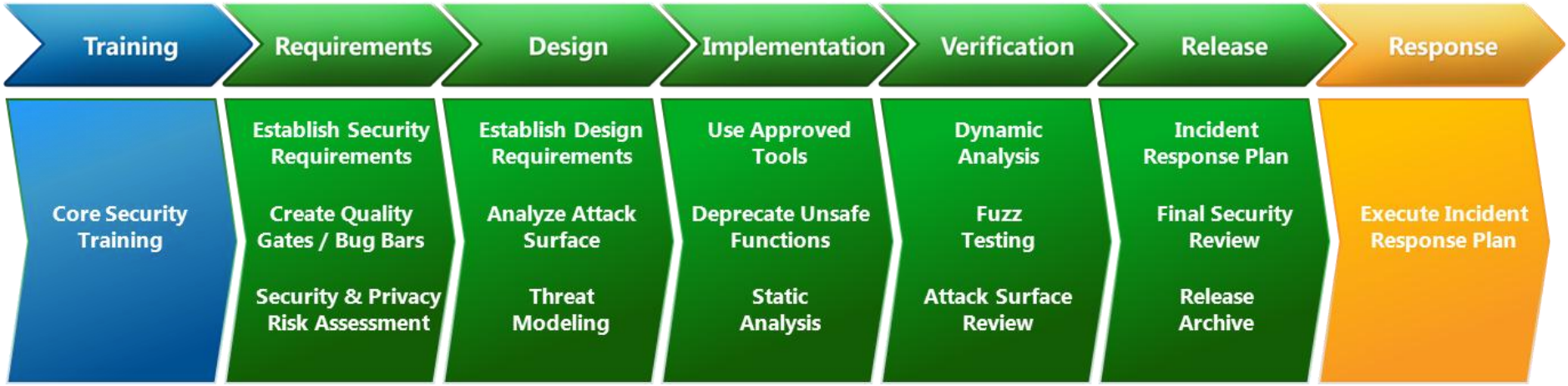
Phase Five: Release



Satisfaction of clearly defined release criteria – consistent with organizational policy

- Incident Response Plan
  - Creation of a plan that outlines engineering, management and "on-call" contacts, security servicing plans for all code, including 3rd party artifacts.
- Final Security Review
  - Deliberate examination of all security and privacy activities conducted during development
- Release Archive
  - SDL compliance certification and archival of all information and data necessary for post-release servicing of the software.

Post-SDL Requirement: Response



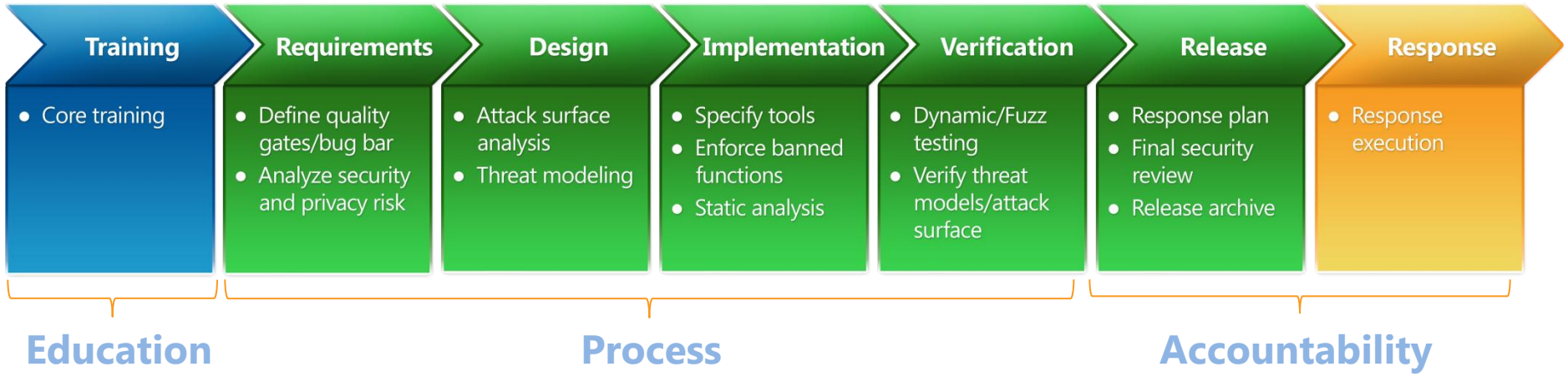
"Plan the work, work the plan…"

- Execute Incident Response Plan
  - Performance of activities outlined in response plan created during Release phase
- Other non-development, post-release process requirements
  - Root cause analysis of found vulnerabilities; failure of human, process, or automation. Addressed immediately and tagged for inclusion in next revision of SDL

SDL application to existing development methodologies

SSW

SDL for Spiral/Waterfall Development

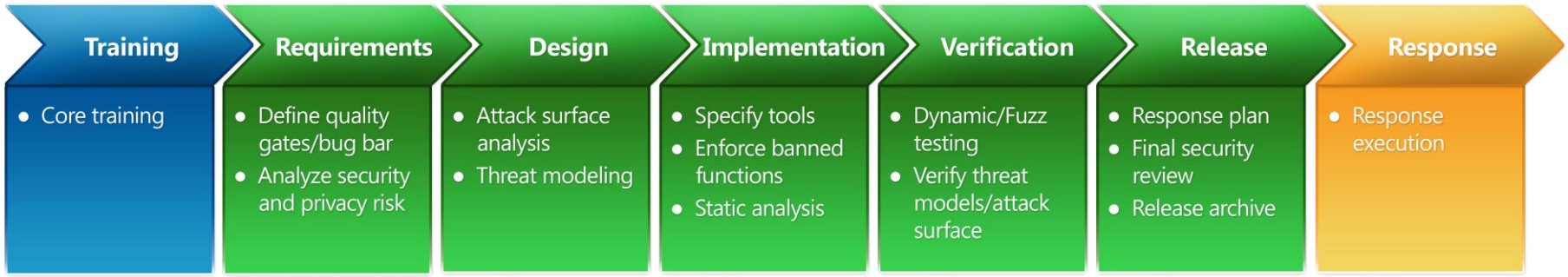


Ongoing Process Improvements

Will SDL Agile?
How …?

SSW

Security Development Lifecycle



- Fits spiral or waterfall…
- …but Agile doesn't have phases
- …and it may not even have a "release"

Idea: Move SDL to product backlog

- Very Agile…
- …but not secure

Idea: Do the full SDL every iteration

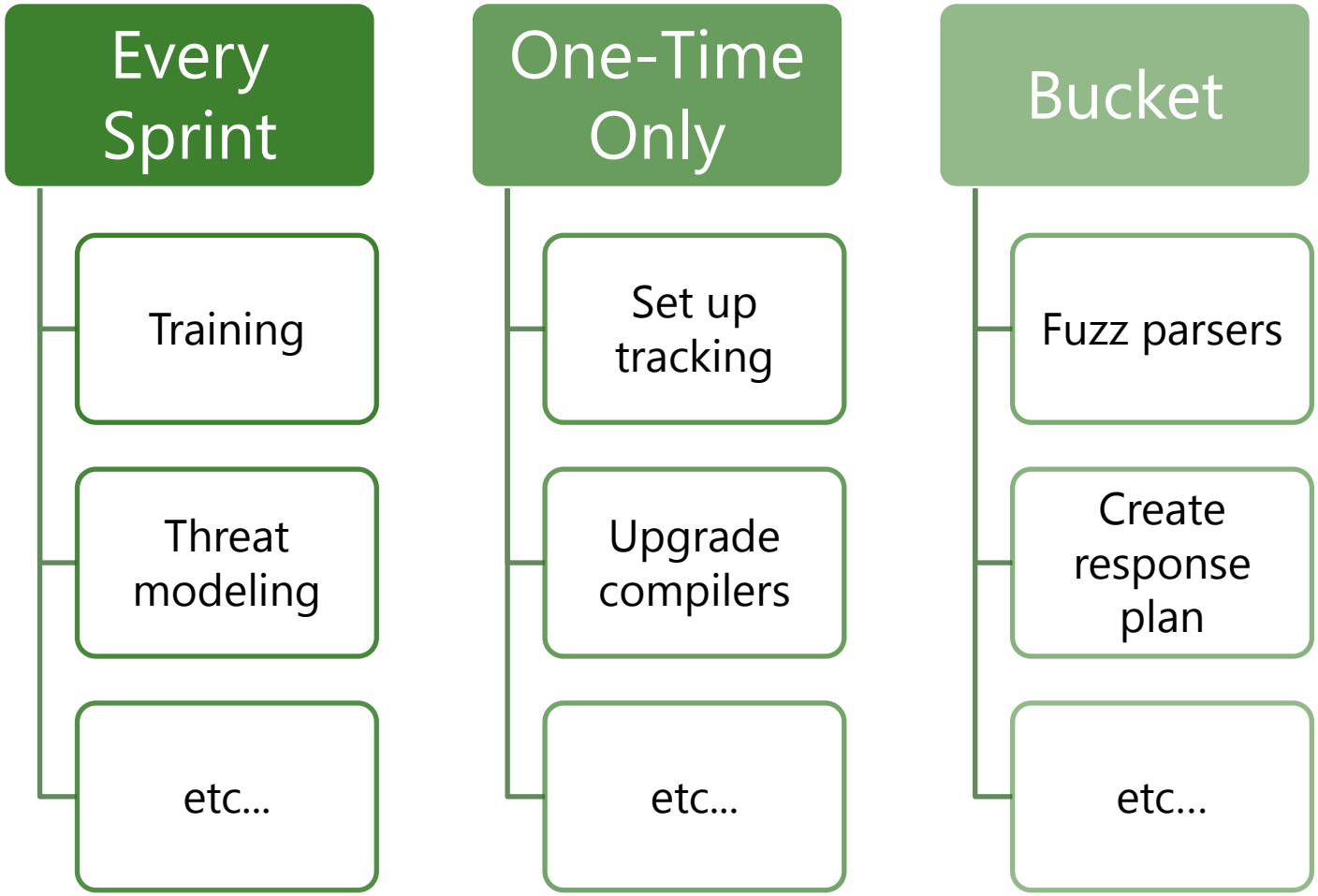
- Very secure…
- …but not Agile!

Idea: Drop some requirements

- But every requirement is, well, required
- Need to keep all requirements
- Need to reorganize into Agile-friendly form

SSW

Three classes of requirements



Every Sprint Requirements … a few examples

Title	Requirement /Recommendation	Applies to Online Services	Applies to Managed Code	Applies to Native Code
Communicate privacy-impacting design changes to the team's privacy advisor	Requirement	X	X	X
Compile all code with the /GS compiler option	Requirement	X		X
Comply with SDL firewall requirements	Requirement		X	X
Do not use banned APIs in new code	Requirement	X		X
Ensure all ASP.NET applications use the ValidateRequest cross-site scripting input validation attribute	Requirement	X	X	
Ensure all database access is performed through parameterized queries to stored procedures	Requirement	X	X	X
Ensure all team members have had security education within the past year	Requirement	X	X	X
Ensure the application domain group is granted only execute permissions on the database stored procedures	Requirement	X	X	X
Fix all issues identified by code analysis tools for unmanaged code	Requirement	X		X
Fix all security issues identified by CAT.NET and FxCop static analysis	Requirement	X	X	
Follow input validation and output encoding guidelines to defend against cross-site scripting attacks	Requirement	X	X	X

http://msdn.microsoft.com/en-us/library/ee790610.aspx

Bucket Requirements … a few examples

Every Sprint: 1 task from every bucket

Verification Tasks	Design Review	Planning
ActiveX fuzzing	Conduct a privacy review	Create privacy support documents
Attack surface analysis	Review crypto design	Update security response contacts
Binary analysis (BinScope)	Assembly naming and APTCA	Update network down plan
File fuzz testing	User Account Control	Define/update security bug bar

http://msdn.microsoft.com/en-us/library/ee790611.aspx

One-time requirements … a few examples

To be completed within a specific timeframe

Title	Requirement/Recommendation	Completion Deadline (months)	Applies to Online Services	Applies to Managed Code	Applies to Native Code
Avoid writable PE segments	Requirement	6	X		X
Create a baseline threat model	Requirement	3	X	X	X
Determine security response standards	Requirement	6	X	X	X
Establish a security response plan	Requirement	6	X	X	X
Identify primary security and privacy contacts	Requirement	1	X	X	X
Identify your team's privacy expert	Requirement	1	X	X	X
Identify your team's security expert	Requirement	1	X	X	X
Use approved XML parsers	Requirement	6	X		X
Use latest compiler versions	Requirement	12	X	X	X
Configure bug tracking to track the cause and effect of security bugs	Recommendation	3	X	X	X
Designate full-time security program manager	Recommendation	3	X	X	X
Remove dependencies on NTLM authentication	Recommendation	12	X	X	X

http://msdn.microsoft.com/en-us/library/ee790612.aspx

Requirements as backlog items

- One-time requirements get added to the Product Backlog (with deadlines)
- So do bucket requirements
- Every-sprint requirements go to the Sprint Backlog directly

Product Backlog

- Set up tracking system
- Upgrade to VS2010
- Fuzz image parser
- Fuzz network parser
- …

Sprint Backlog

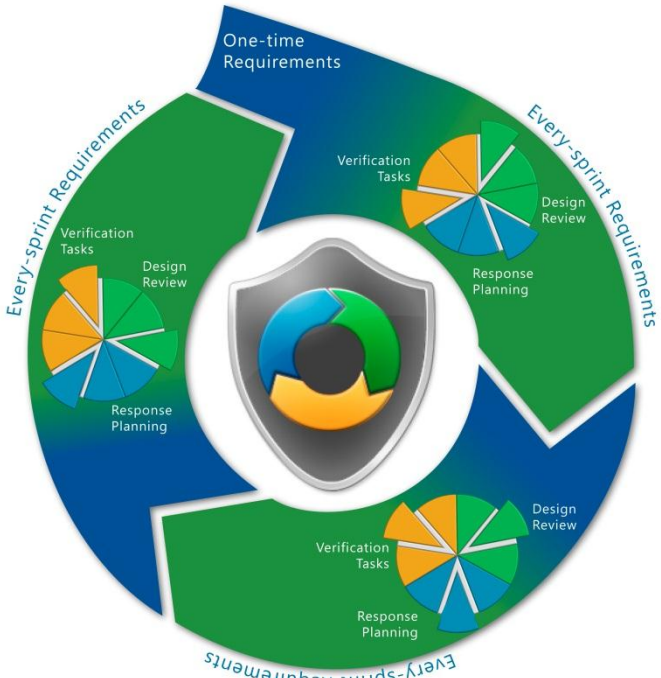
- Threat model new stored procedures
- Run static analysis
- …

Sprint exit criteria

- All every-sprint requirements complete
- No bucket items over six months/weeks old
- No expired one-time requirements
- No open security bugs over the bugbar

SDL Process Guidance for Agile Development Methodologies

- Requirements defined by frequency, not phase
  - Every-Sprint (most critical)
  - One-Time (non-repeating)
  - Bucket (all others)



- Great for projects without end dates, like cloud services

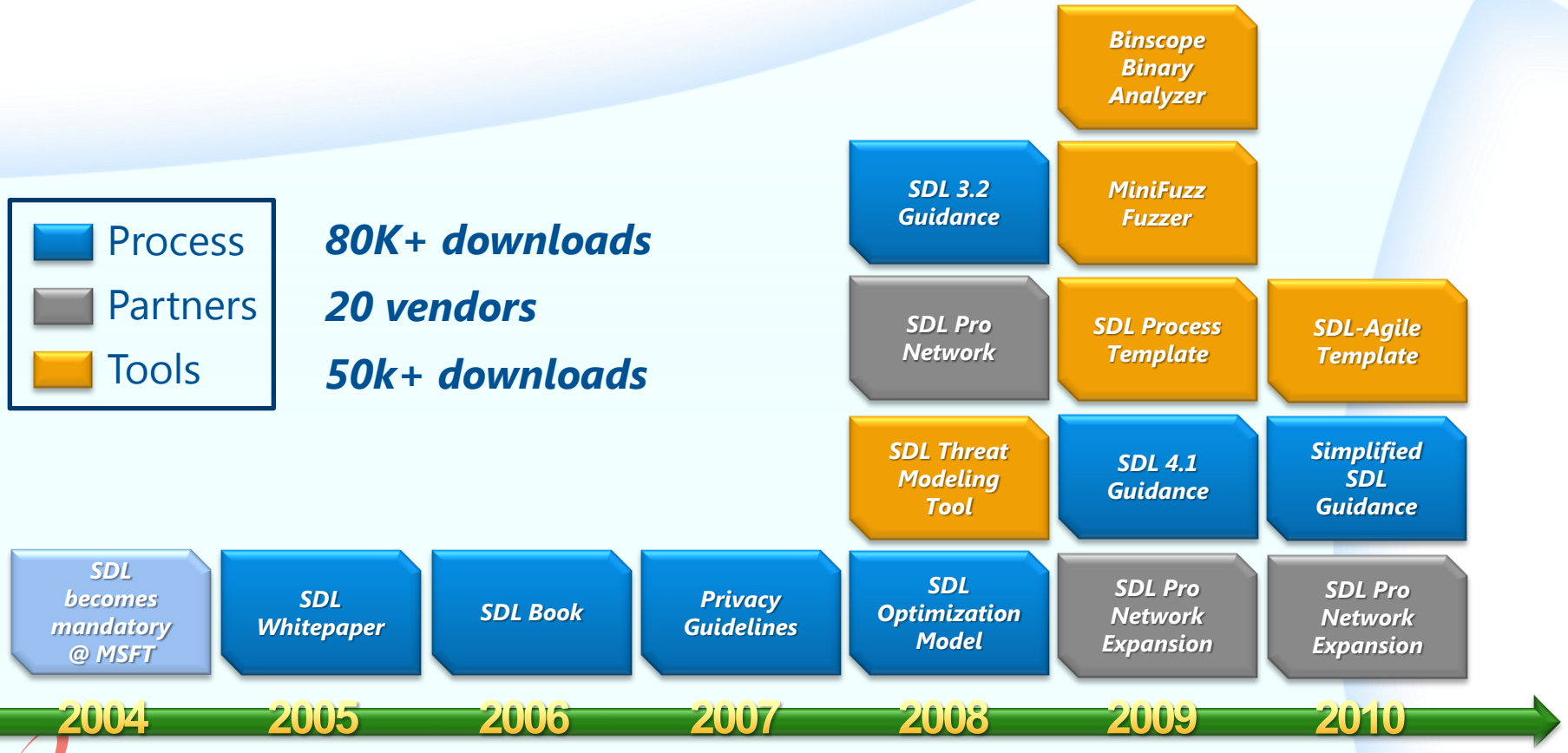
Demo: MSF Agile + SDL Process Template

SSW

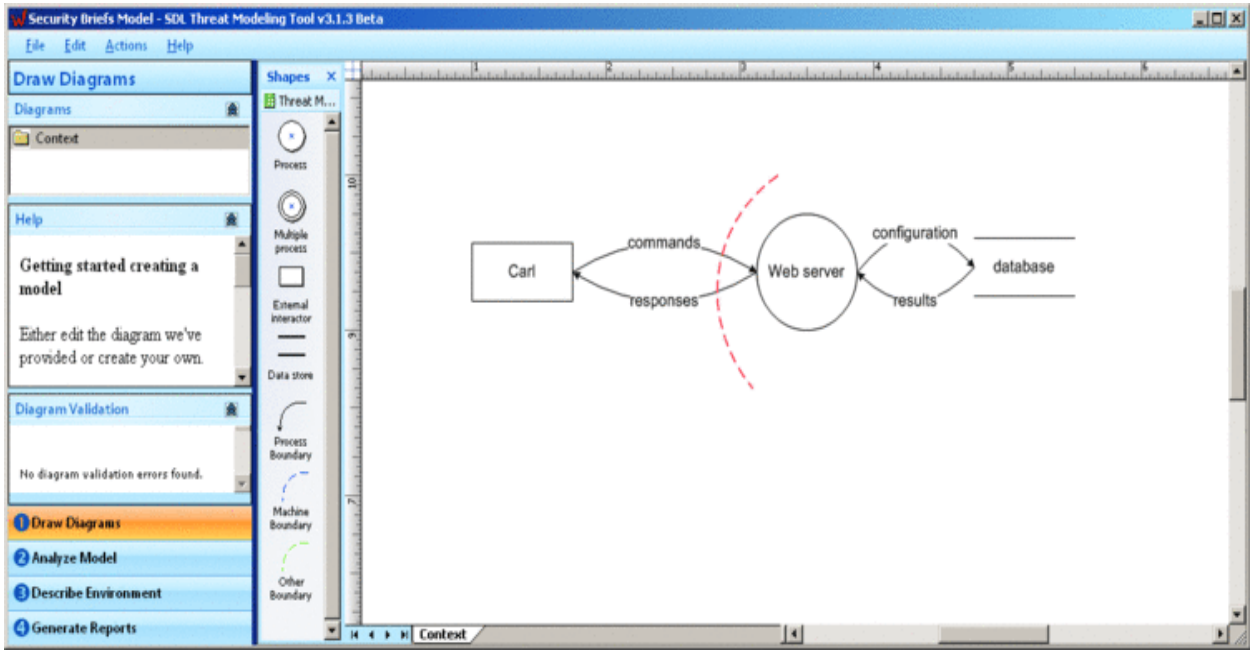
Resources for Development Organizations

SSW

Resources from Microsoft, at a glance…



SDL Threat Modeling Tool

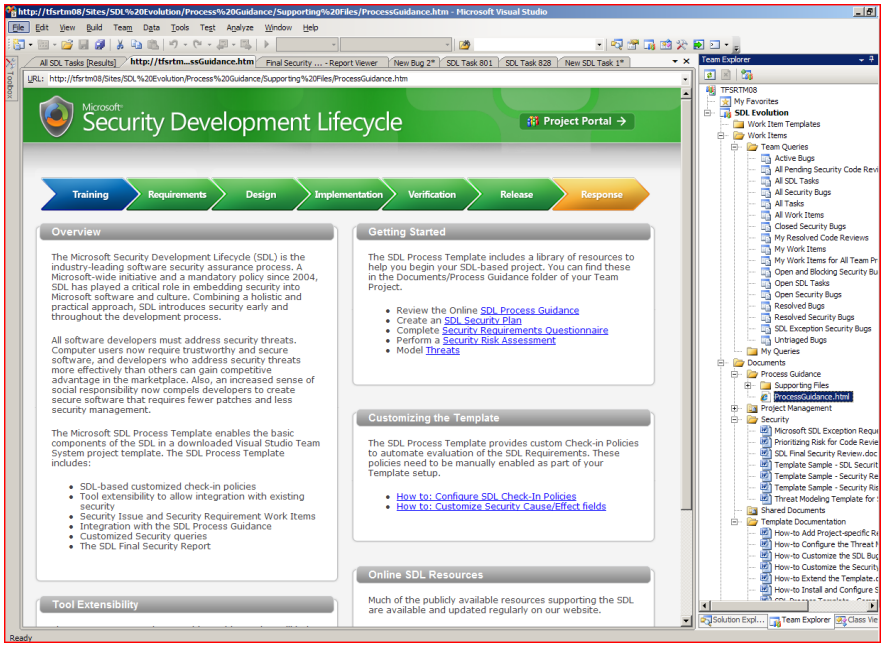


Transforms threat modeling from an expert-led process into a process that any software architect can perform effectively

Provides:

- Guidance in drawing threat diagrams
- Guided analysis of threats and mitigations
- Integration with bug tracking systems
- Robust reporting capabilities

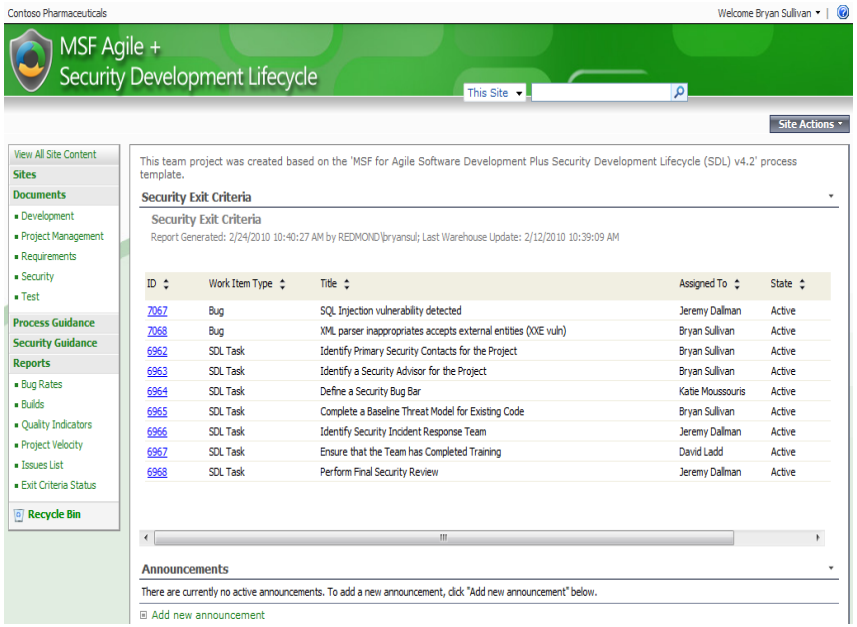
SDL Template for VSTS (Spiral)



The SDL Process Template integrates SDL 4.1 directly into the VSTS software development environment.

- Incorporates
  - SDL requirements as work items
  - SDL-based check-in policies
  - Generates Final Security Review report
  - Third-party security tools
  - Security bugs and custom queries
  - A library of SDL how-to guidance
- Integrates with previously released free SDL tools
  - SDL Threat Modeling Tool
  - Binscope Binary Analyzer
  - Minifuzz File Fuzzer

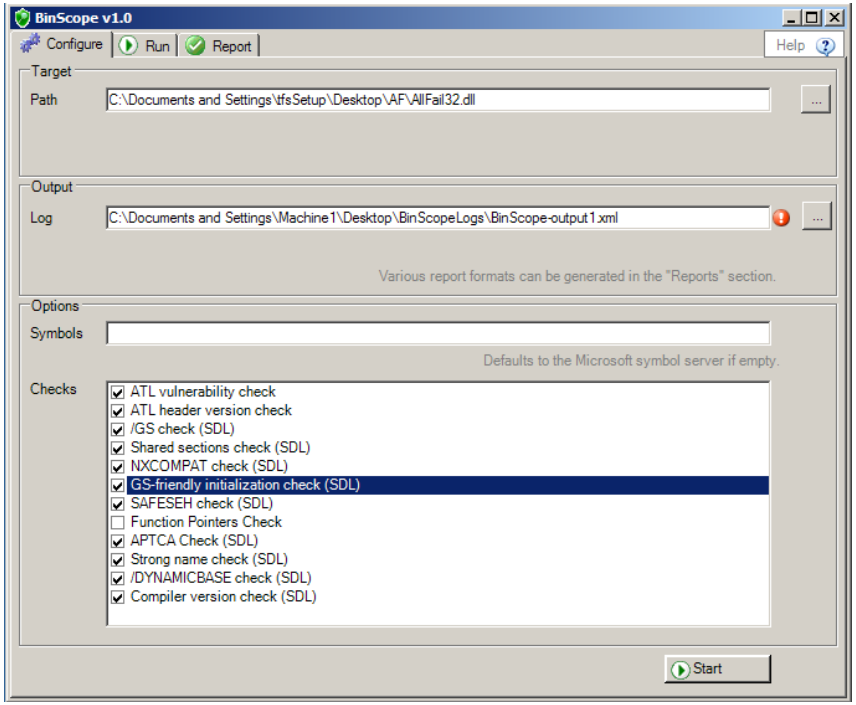
MSF Agile + SDL Template for VSTS



- Incorporates SDL-Agile secure development practices directly into the Visual Studio IDE - now available as beta (planned release at the end of Q2CY10)

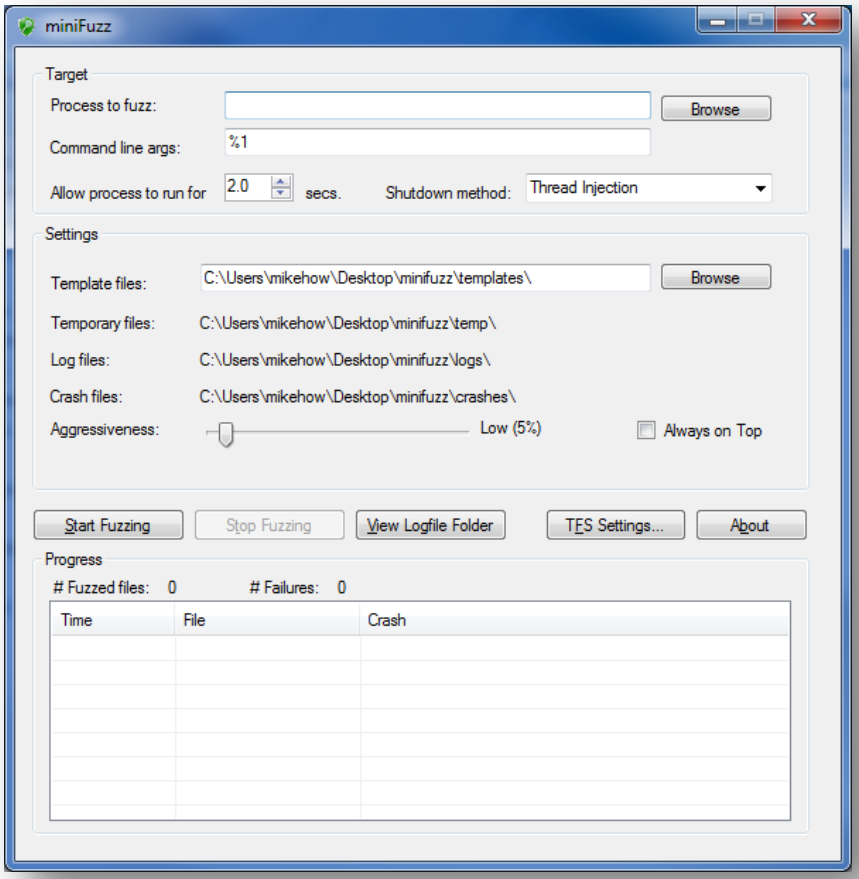
- Automatically creates new security workflow items for SDL requirements whenever users check in code or create new sprints
- Ensures important security processes are not accidentally skipped or forgotten
- Integrates with previously released free SDL tools
  - SDL Threat Modeling Tool
  - Binscope Binary Analyzer
  - Minifuzz File Fuzzer
- Will be updated for VS2010

Binscope Binary Analyzer



- Provides an extensive analysis of an application binary
- Checks done by Binscope
  - /GS - to prevent buffer overflows
  - /SafeSEH - to ensure safe exception handling
  - /NXCOMPAT - to prevent data execution
  - /DYNAMICBASE - to enable ASLR
  - Strong-Named Assemblies - to ensure unique key pairs and strong integrity checks
  - Known good ATL headers are being used
- Use either standalone or integrated with Visual Studio (VS) and Team Foundation Server (TFS)

MiniFuzz File Fuzzer



- MiniFuzz is a basic testing tool designed to help detect code flaws that may expose security vulnerabilities in file-handling code.
  - Creates corrupted variations of valid input files
  - Exercises the code in an attempt to expose unexpected application behaviors.
  - Lightweight, for beginner or advanced security testing
  - Use either standalone or integrated with Visual Studio (VS) and Team Foundation Server (TFS)

SDL Pro Network

Consulting Members

Booz | Allen | Hamilton
strategy and technology consultants

casaba

cigital

CONSULT COMPLY

IOActive™
COMPREHENSIVE COMPUTER SECURITY SERVICES

iSEC PARTNERS

leviathan security group

next generation security software
an ncc group company

nruns professionals

SAIC
From Science to Solutions

SecurityInnovation
THE SOFTWARE SECURITY COMPANY

verizonbusiness

Training Members

SANS

SAFELIGHT
SECURITY ADVISORS

Security University®
Where "Qualified" Happens!

Tools Members

Beyond Security

CODENOMICON

FORTIFY

Mu Dynamics

VERACODE

Summary

Attacks are moving to the application layer

SDL = embedding security into software and culture

Measurable results for Microsoft software

Microsoft is committed to making SDL widely available and accessible

SSW

Resources

SDL Portal

http://www.microsoft.com/sdl

SDL Blog

http://blogs.msdn.com/sdl/

SDL Process on MSDN (Web)

http://msdn.microsoft.com/en-us/library/cc307748.aspx

Simplified Implementation of the Microsoft SDL

http://go.microsoft.com/?linkid=9708425

Questions?

Lei Xu

Blog: http://www.almnetworks.net
Email: leixu@ssw.com.au
Phone: 6815 5515

Do what Microsoft did, not what they do...

- ***"...If you take the SDL Microsoft has described and try to implement it, you will fail. I am talking to the 99% of people out there who would think about implementing SDL and think "Hey, Microsoft published this new thingie for free; let's use it and save ourselves the time and money!" Wrong."***

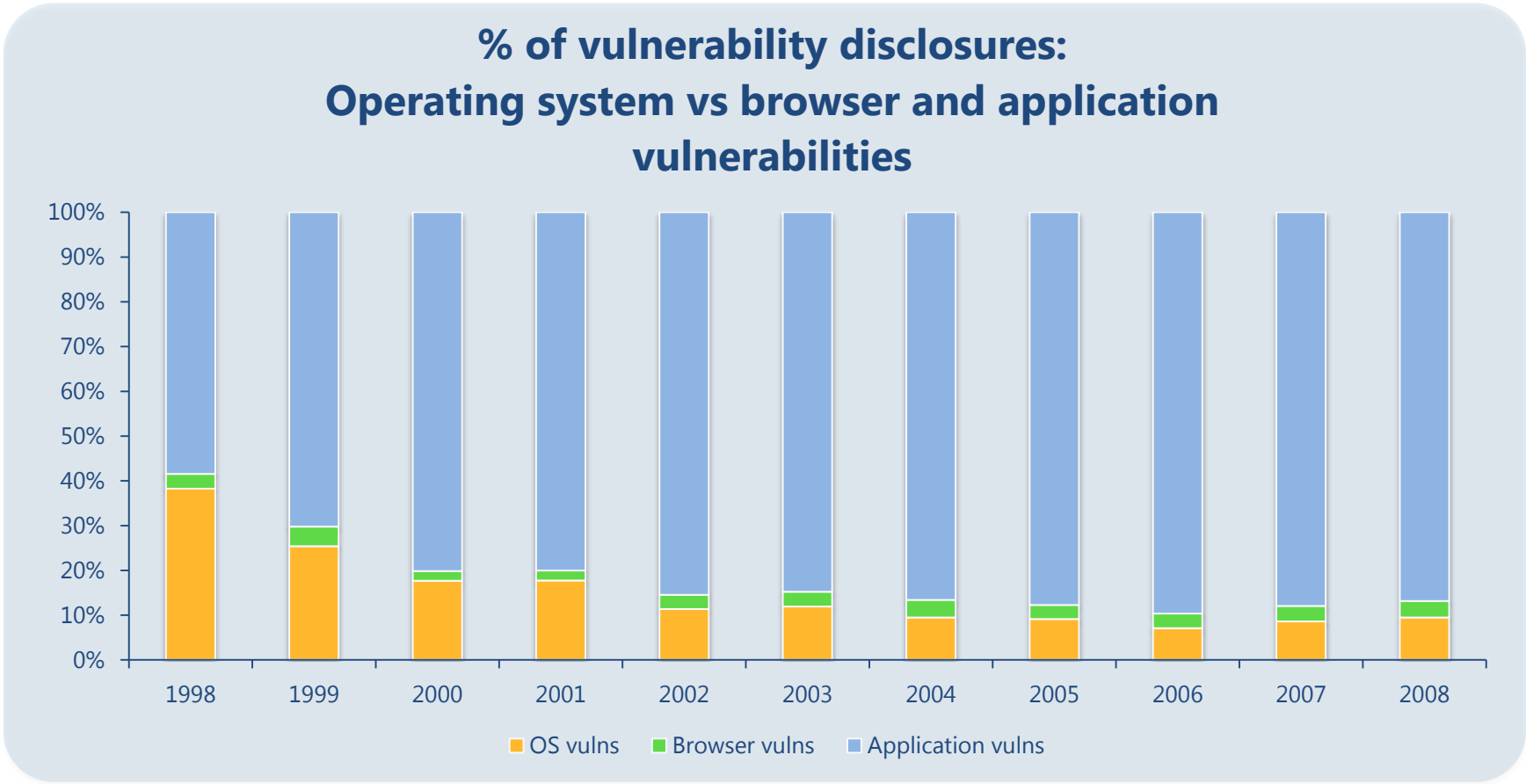
http://blogs.msdn.com/b/sdl/archive/2010/05/11/do-what-microsoft-did-not-what-they-do.aspx

BACKUP

Applications under attack...

SSW

Attacks are focusing on applications



Calculated from the Microsoft Security Intelligence Report V6

90% of vulnerabilities are remotely exploitable

Sources: IBM X-Force, 2008