Challenges and Opportunities in Implementing a Security Development Lifecycle

Rob Caldwell

imagination at work

GE
Digital Energy

Overview

The problem

Change is hard

Where do we go from here

Four initial focus areas

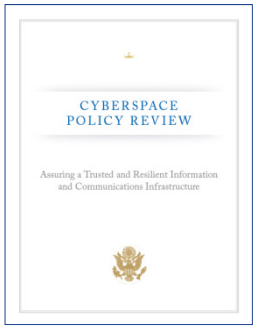
Other important needs

Measuring progress

Market drivers

Regulatory drivers

Current Landscape



NIST

CPNI
Centre for the Protection of National Infrastructure

NERC ISA
NORTH AMERICAN ELECTRIC RELIABILITY CORPORATION

Customer and regulatory requirements

- NERC CIP Reliability Standards, ISA99
- Department of Homeland Security: Cyber Security Procurement Language for Control Systems
- **Impact:** Increased development cost, potential revenue loss

Targeted cyber attacks & media interest

- Stuxnet malware targeting Siemens WinCC SCADA systems
- Security researcher interest in control systems security
- “Advanced Persistent Threat” & Google “Aurora” attacks
- **Impact:** Potential for litigation or reputational/brand injury

Product security certification & marketing

- Microsoft Trustworthy Computing and SDL
- Emerging cyber security certifications (ex. Wurldtech Achilles, ISASecure, SEI CERT SCALe)
- **Impact:** Security as a competitive factor

imagination at work

The Air Gap is gone…

We can no longer rely on network defenses

Inter-connection IS the key in Smart Grid

Application Security is absolutely critical!

Why transformational efforts fail

Stage	Action Needed	Pitfalls
Establish a sense of urgency	• Examine market and competitive realities for potential crises and untapped opportunities. • Convince at least 75% of your managers that the status quo is more dangerous than the unknown.	• Underestimating the difficulty of driving people from their comfort zones • Becoming paralyzed by risks
Form a powerful guiding coalition	• Assemble a group with shared commitment and enough power to lead the change effort. • Encourage them to work as a team outside the normal hierarchy.	• No prior experience in teamwork at the top • Relegating team leadership to an HR, quality, or strategic planning executive rather than a senior line manager
Create a vision	• Create a vision to direct the change effort. • Develop strategies for realizing that vision.	• Presenting a vision that's too complicated or vague to be communicated in five minutes
Communicate the vision	• Use every vehicle possible to communicate the new vision and strategies for achieving it. • Teach new behaviors by the example of the guiding coalition.	• Undercommunicating the vision • Behaving in ways antithetical to the vision

imagination at work

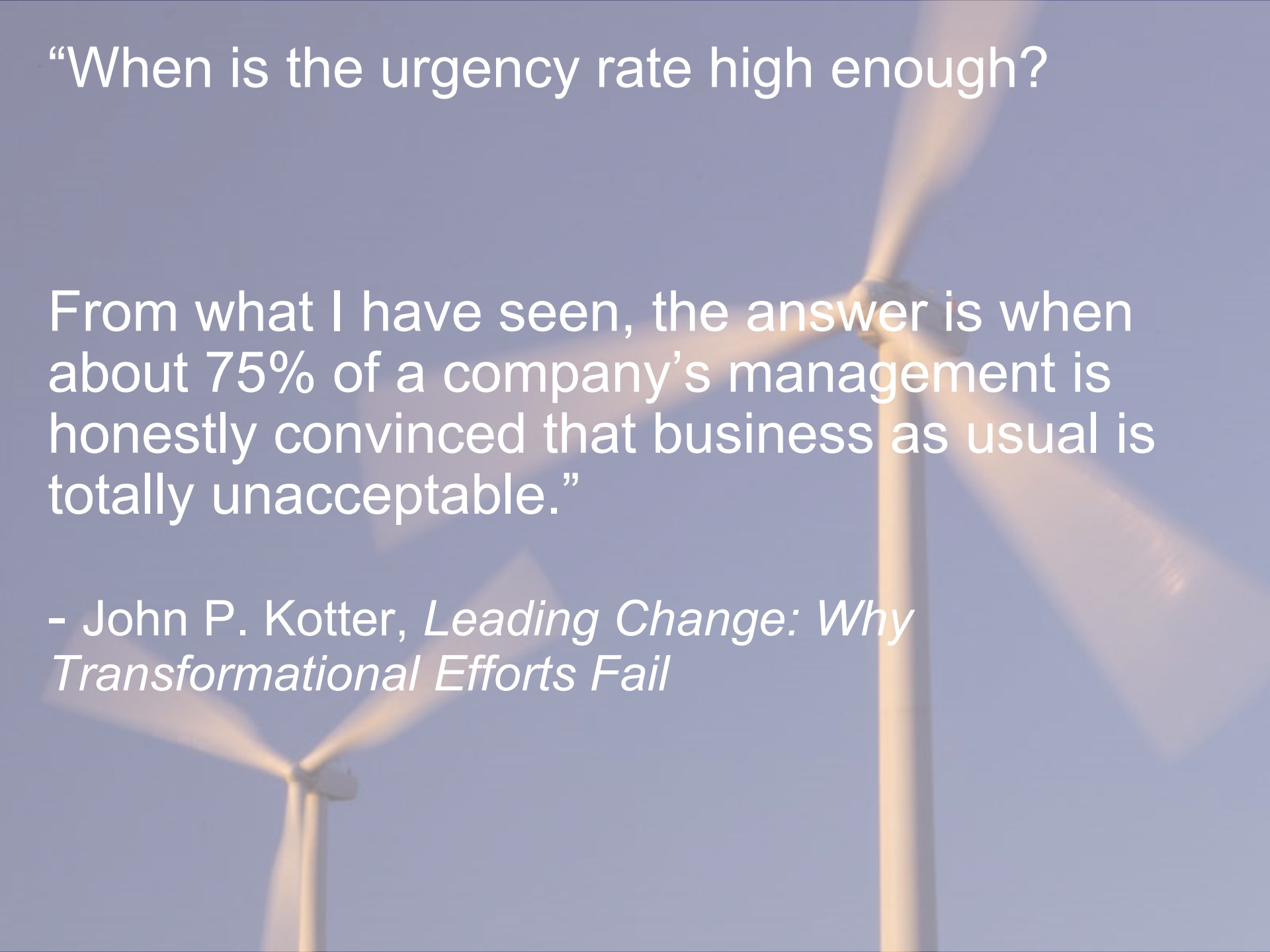
Source:
Leading Change: Why Transformational Efforts Fail
By John P. Kotter, From *Harvard Business Review*

Why transformational efforts fail (Continued)

Stage	Action Needed	Pitfalls
Empower others to act on the vision	• Remove or alter systems or structures undermining the vision. • Encourage risk taking and nontraditional ideas, activities, and actions	• Failing to remove powerful individuals who resist the change effort
Plan for and create short-term wins	• Define and engineer visible performance improvements. • Recognize and reward employees contributing to those improvements.	• Leaving short-term successes up to chance • Failing to score successes early enough (12-24 months into the change effort)
Consolidate improvements and produce more change	• Use increased credibility from early wins to change systems, structures, and policies undermining the vision. • Hire, promote, and develop employees who can implement the vision. • Reinvigorate the change process with new projects and change agents	• Declaring victory too soon with the first performance improvement • Allowing resistors to convince “troops” that the war has been won
Institutionalize new approaches	• Articulate connections between new behaviors and corporate success. • Create leadership development and succession plans consistent with the new approach	• Not creating new social norms and shared values consistent with changes • Promoting people into leadership positions who don’t personify the new Approach

imagination at work

Source:
Leading Change: Why Transformational Efforts Fail
By John P. Kotter, From *Harvard Business Review*



“When is the urgency rate high enough?

From what I have seen, the answer is when about 75% of a company’s management is honestly convinced that business as usual is totally unacceptable.”

- John P. Kotter, *Leading Change: Why Transformational Efforts Fail*

Microsoft Security Development Lifecycle

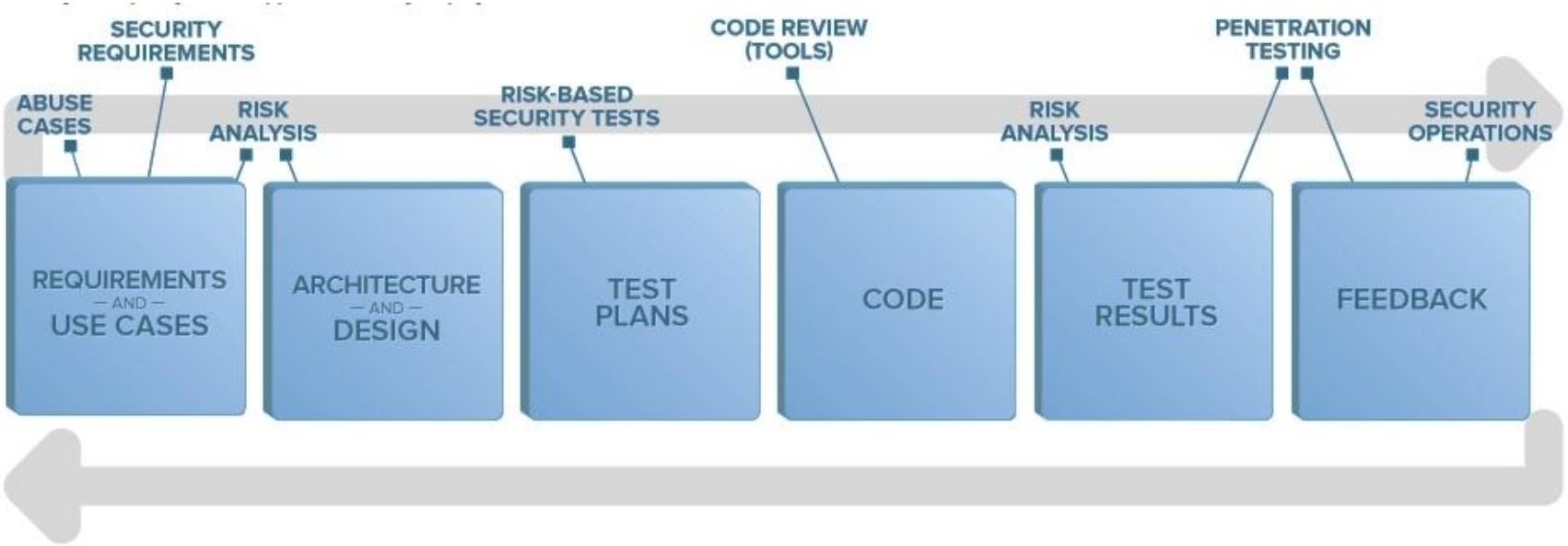
Huge investment in security after 2002 Trustworthy Computing memo

Very focused on the way Microsoft develops products

Training	Requirements	Design	Implementation	Verification	Release	Response
1. Core Security Training	2. Establish Security Requirements	5. Establish Design Requirements	8. Use Approved Tools	11. Perform Dynamic Analysis	14. Create an Incident Response Plan	
	3. Create Quality Gates/Bug Bars	6. Perform Attack Surface Analysis/ Reduction	9. Deprecate Unsafe Functions	12. Perform Fuzz Testing	15. Conduct Final Security Review	Execute Incident Response Plan
	4. Perform Security and Privacy Risk Assessments	7. Use Threat Modeling	10. Perform Static Analysis	13. Conduct Attack Surface Review	16. Certify Release and Archive	

Cigital Security Touchpoints

Development methodology agnostic



What direction were we going to take?

Challenge – Disparate develop methods, varying Agile process maturity

Opportunity – Widespread support from senior management on down

Software security maturity models

	Building Security In Maturity Model (BSIMM)	Software Assurance Maturity Model (OpenSAMM)
Background	BSIMM is a real-world set of software security activities organized so that you can determine where you stand with your software security initiative and how to evolve your initiative over time. Authored by Gary McGraw, Sammy Migues, & Jacob West	The Software Assurance Maturity Model (SAMM) is an open framework to aid organizations in: • Evaluating an organization's existing software security practices • Building a balanced software security program in well-defined iterations • Demonstrating concrete improvements to a security assurance program • Defining and measuring security-related activities within an organization OpenSAMM is an OWASP project led by Pravir Chandra.
Advantages	• Built-in benchmark against nine leading software organizations (Adobe, Google, Microsoft, QUALCOMM, EMC, DTCC, Wells Fargo, etc.) • Comprehensive and customizable	• Easy-to-interpret checklist • Limited effort required to get a basic measurement of program maturity • Clear baseline of basic activities organizations should practice
Disadvantages	• No simple "checklist" • Requires considerable interpretation and analysis effort http://www.bsi-mm.com	• Unclear which assurance maturity level is "enough" security • Does not provide a frame of reference to other organizations http://www.opensamm.org

Building Security In Maturity Model

The Software Security Framework (SSF)			
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

4 domains, each with 3 practices

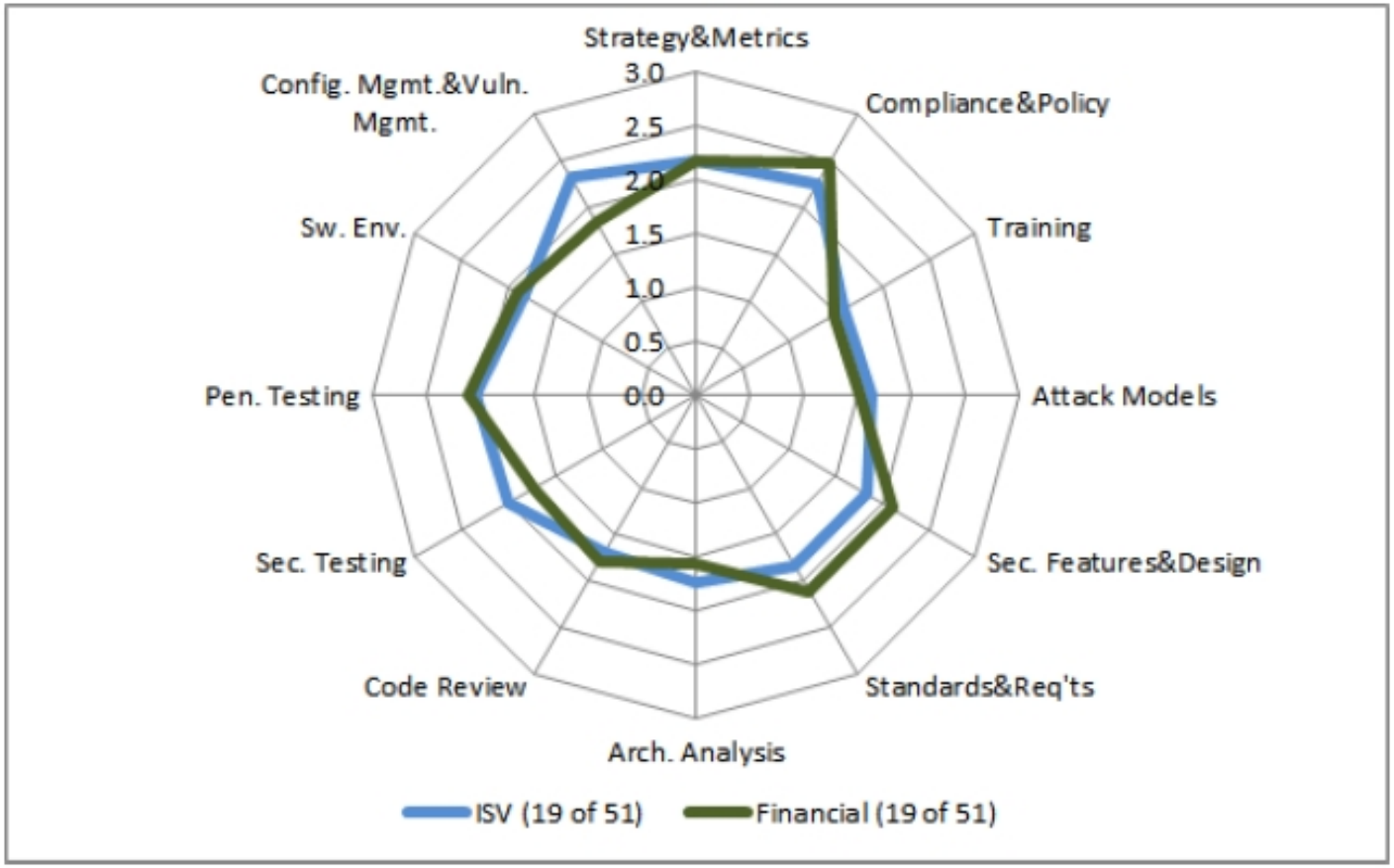
-Each Practice has 3 maturity levels

-Each Level has 1-5 activities

-111 activities in total

“Note that no organization carries out all of the activities described in the BSIMM.”

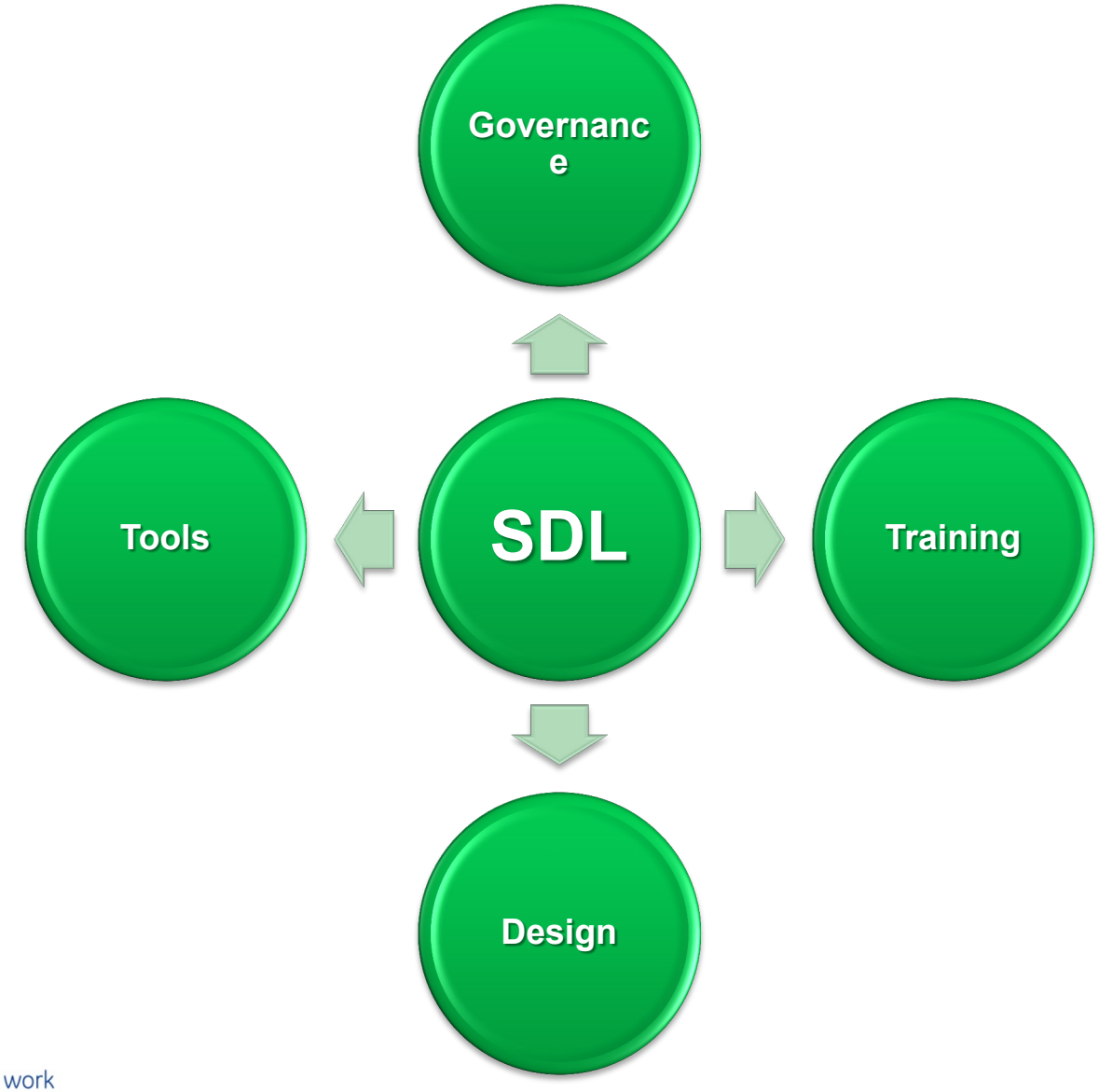
Building Security In Maturity Model



Our combined Kiviat diagram showed obvious areas for focus

imagination at work

SDL Focus Areas



imagination at work

Governance

Opportunity – existing robust design review process allowed for easy integration of security tollgates

Early win that was easy to demonstrate and measure

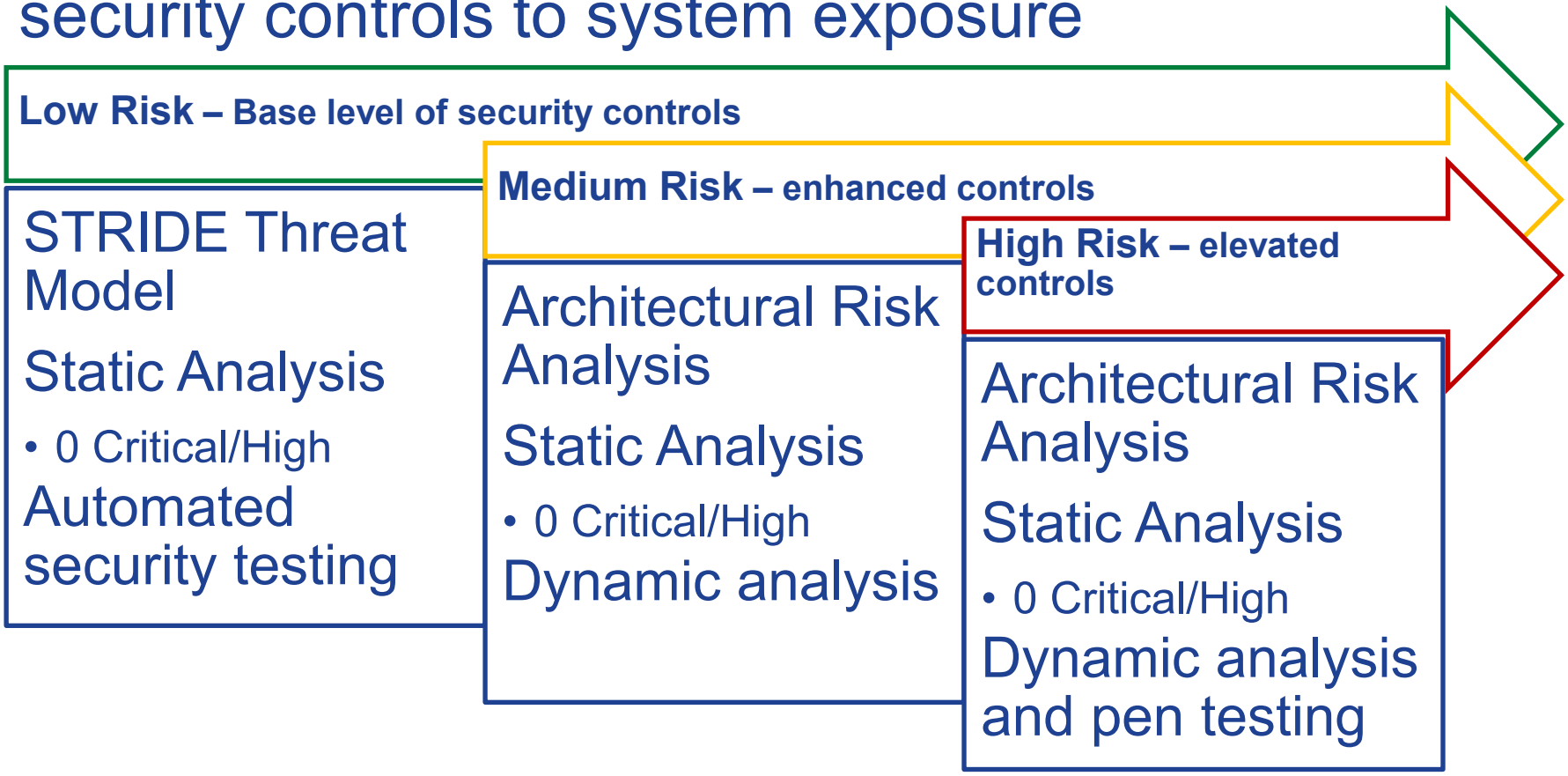
Buy-in from Chief Architects' team made this a sustainable, scalable process

Success is having the right decisions made when you aren't in the room.

imagination at work

Governance (continued)

Application risk levels – match the level of security controls to system exposure



Training

Challenge – personnel know security is important, but don't know what "security" means, or what to do

SDL implementation will fail unless underpinned by effective security training program

Computer based training works for fundamental concepts, however, instructor led training is most effective for advanced concepts (understanding exploits, defensive coding, threat modeling, etc.)

Training (continued)

Role-specific security training program

- Management
- Developers
- Business Analysts/Scrum Masters
- Test/Quality Assurance
- Architects

Focus is on building awareness and hands-on skills that feed into the other facets of the SDL

Examples: Identifying security requirements, Code review and static analysis, Architectural risk assessment, Defensive coding

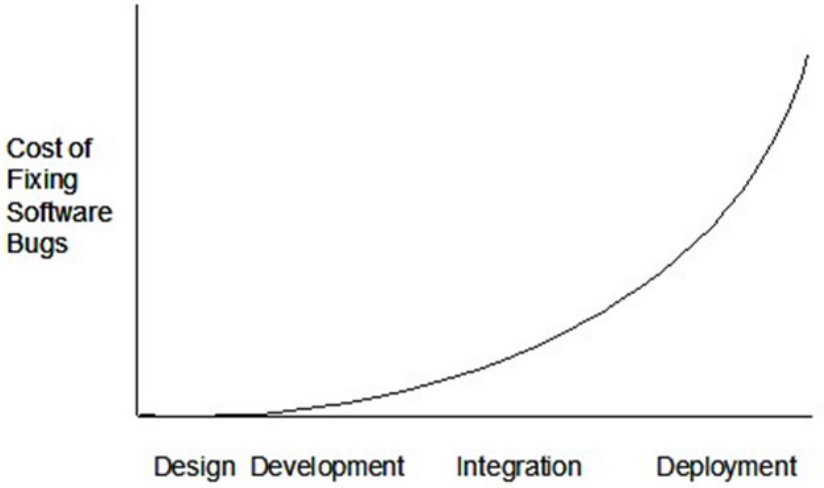
Design

Opportunity – leverage design review process/tollgates to enforce security at beginning of lifecycle

Two primary types of security issues in applications

- Flaws, driven by application design
- Bugs, driven by faults/errors in coding

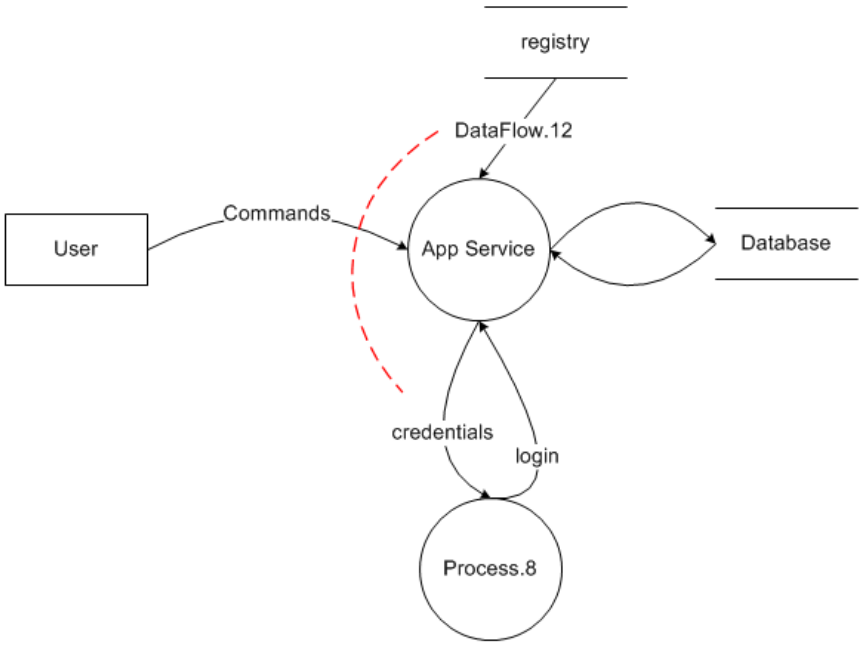
Goal is to identify and eliminate flaws before they get written into the application, and fix coding errors before product release



NIST *The Economic Impacts of Inadequate Infrastructure for Software Testing*, 2002

imagination at work

Design - Threat Modeling



Microsoft

Threat Modeling addresses finding application flaws and areas of concern early in the design process

Required to pass through the design review process (product security plan)

Depth of model dependent on application risk level

Interesting and informative team interactions when working on model

Design - Secure Coding Standards

Secure Coding standards provide a foundation developers refer to while coding

Compiled from industry sources and best practices, such as:

- CWE/SANS Top 25 Most Dangerous Software Errors
- Oracle's Secure Coding Guidelines for Java
- SEI CERT Secure Coding Standards for C
- And others…

Examples include input validation to prevent buffer overflow, prohibiting use of banned APIs, and using structured exception handlers

Tools – Code Review

Both manual code review and static code analysis required

- Manual code review necessary to catch logic issues, malicious code
- Static analysis useful for consistently catching common vulnerabilities, “heavy lifting”

Static code analysis tied to nightly build process, flagging errors for immediate fix, rather than waiting until testing to find them (or not!)

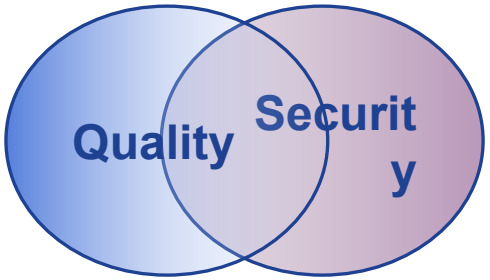
Opportunity – Organization-wide effort to standardize IDEs

Long-term goal of deploying SCA to developer IDE

Quick side trip – choosing an SCA tool

Security and quality teams partnered to choose tool

SCA toolsets available in the industry cover both aspects



Unified objective: Change the quality conversation

Evaluation Criteria

CTQ	Typical Measurements
Quality of Results	• Quality metrics (cyclomatic complexity, cohesion, etc.) • Security metrics (SANS Top 25, OWASP, etc.) • Action-ability of results (categorization/severity) • Action-ability of results (remediation)
Technology Support	• Languages • Platforms • Frameworks • IDEs
Automation	• Command line interface, enabling automation and incorporation into build process. • Pre-build scanning by development staff (scheduled vs. on-demand)
Customization	• Customization (rules/severity)
Developer Focus	• Development team opinions of tools
Cost Estimate	• License model, deployment footprint, etc.
Intangibles	• “What makes your product better than your competitors?”
Ease of Use & Setup	• Hours to configure/produce results • Required training

imagination at work

Observations

Two types of static analysis tools

- Quality driven
- Security auditor driven

Markedly different philosophies in handling findings

Most tools produce similar results, but differ in implementation/architecture

Licensing models are drastically different

Tool Maxims

1. Tools usage ≠ irrefutable assurance
2. Great auditor tools ≠ great developer tools
3. Tools are not “point and scan”
4. Tools are platform and language-specific
5. Tools cannot solve process problems

-GE Security COE

Beyond the four initial SDL focus areas – Security Testing

Opportunity – GE's skilled internal security assessment team located in IT

Opportunity – Mature commercial control system security assessment vendors

Opportunity – Much easier to get approval for security testing tools, IT's permission to set up security labs

Goal to use QA for some security testing, but we do not want to make them pen testers

Growing interest amongst developer community

Another critical process - Vulnerability Management

Opportunity – Strong customer support and defect resolution processes provide infrastructure for vulnerability management

Challenge – Support personnel also need to receive security training

Challenge – Support processes differ between product lines, due to different market needs

Measuring progress

Goals set for product lines/development groups

- Buy-in on implementation dates

Teams without formal security personnel nominate a "security representative"

Monthly status calls as a group, individual follow-up calls more frequent

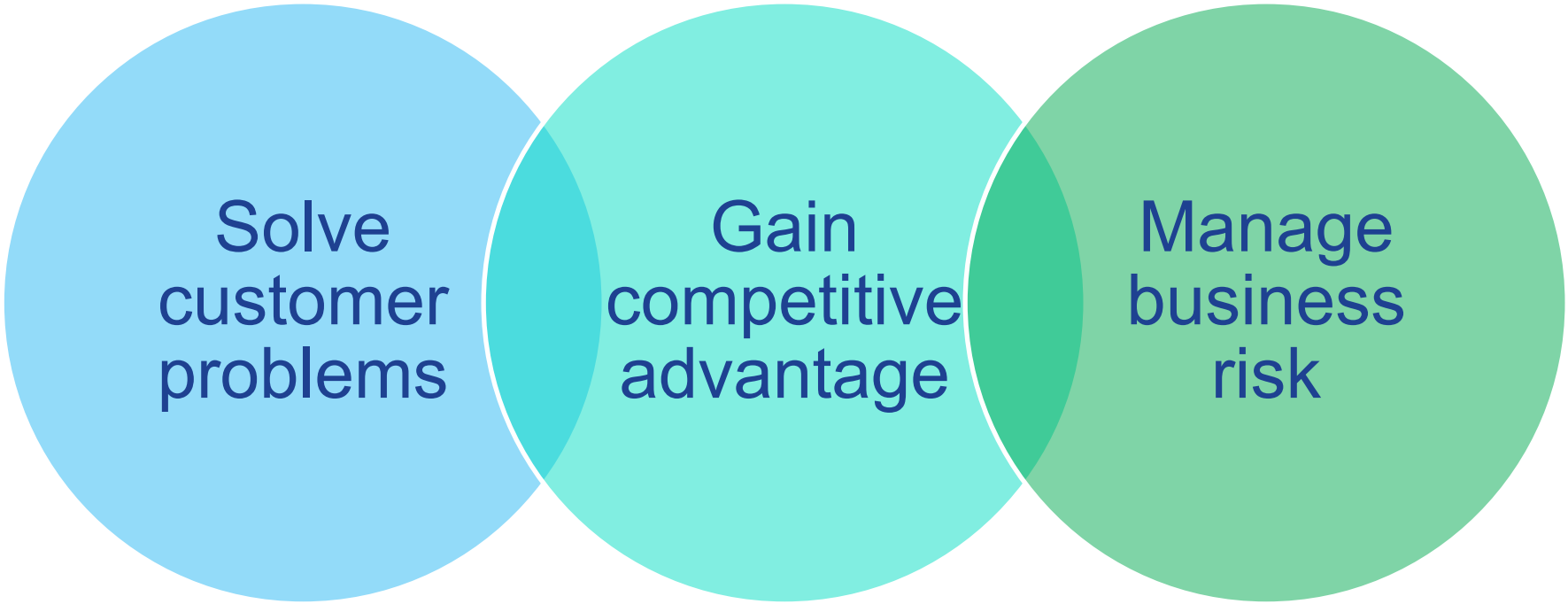
Monthly report-out to management of Security Scorecard

Stoplight metrics – very "corporate", but they actually work

	Risk Categorization	Training	Threat Model	SCA on CI	Vuln Response Process	Vuln Process Exercise
Product 1	✓	100%	30-Aug-13	30-Sep-13	✓	15-Aug-13
Product 2	27-Sep-13	100%	31-Dec-13	29-Nov-13	27-Sep-13	27-Sep-13
Product 3	✓	100%	1-Oct-13	✓	✓	1-Nov-13
Product 4	✓	96%	30-Sep-13	30-Sep-13	31-Dec-13	31-Dec-13
Product 5	✓	98%	27-Dec-13	27-Dec-13	✓	✓
Product 6	✓	100%	27-Sep-13	✓	29-Nov-13	29-Nov-13
Product 7	✓	96%	23-Dec-13	✓	23-Dec-13	23-Dec-13

	Complete or planned date greater than four weeks
	Planned date within four weeks
	Planned date within two weeks
	Planned date is late

Market drivers for an ISV



- Address security in RFPs
- Enable customer regulatory requirements
- Conform to industry security standards

- Differentiate products with security/robustness
- Include innovative security features
- Faster to market with secure products – at a lower cost

- Protect the brand
- Ensure product quality and reliability
- Respond promptly to security issues

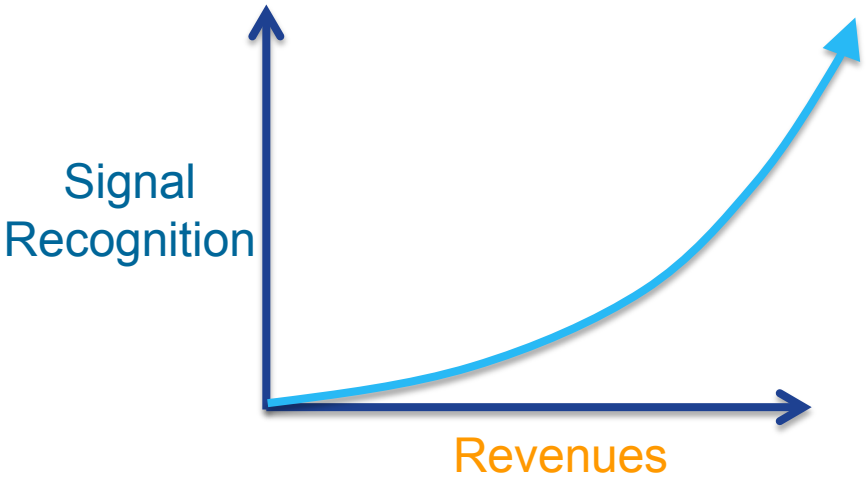
imagination at work

Security product labeling would be valuable

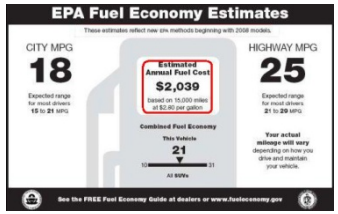
When security is viewed as a business risk, economic incentives don't necessarily drive quality and security controls during development

However, when viewed as a product capability, economic incentives naturally drive new software features & capabilities

Product Labeling democratizes product **intangibles**, making them **visible** to all buyers - not just technical experts



Real-world labeling examples:



Sources: David Rice, Presentation to GE Product Security Working Group, 7/27/2010

Wikipedia - http://en.wikipedia.org/wiki/Signalling_%28economics%29

ecomagination

imagination at work

Multitude of regulatory and standards drivers

NIST SP series and NISTIR 7628

ISO 27001

ISO 15408 “Common Criteria”

ISA-99, now ISA/IEC-62443

SEI CERT SCALe for Energy Delivery Systems

CIS/DISA/FISMA….

Our customers are most concerned about NERC CIPS

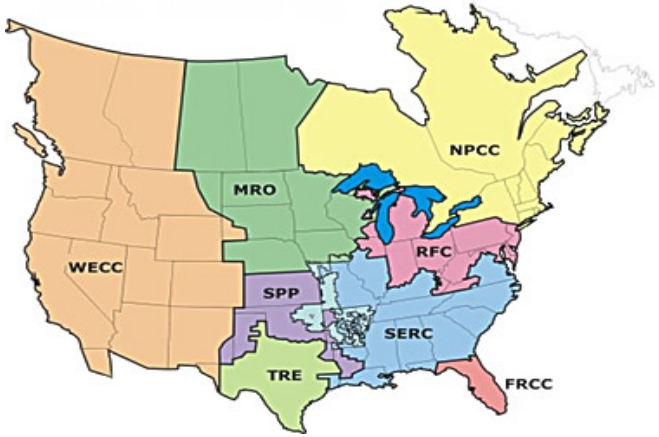
NERC CIPS

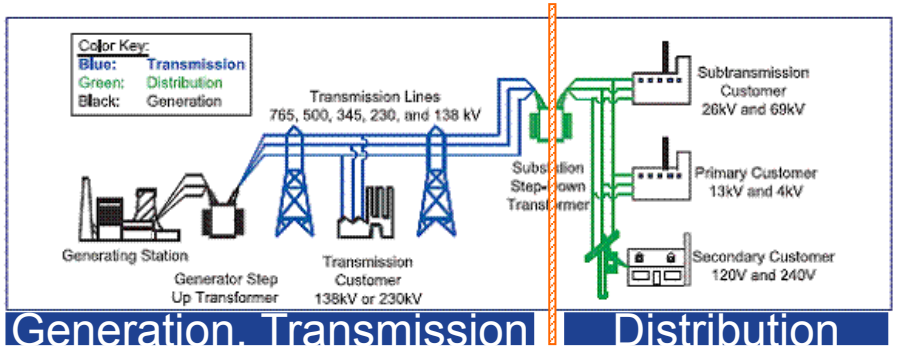
Every utility has a slightly different interpretation

Enforcement has varied among regions

Line between Transmission and Distribution is clear now, but less so in the future

This makes it difficult for both utilities and vendors





imagination at work

NERC compliance penalties are real…

…but are sometimes used to drive FUD – fear, uncertainty, and doubt ($1,000,000 per day per incident)

However, the $ amount of enforcement actions is growing

~$150,000 max in 2011 to $350,000 in July, 2013

	Violation Severity Level							
Violation Risk Factor	Lower		Moderate		High		Severe	
	Range Limits/Day		Range Limits/Day		Range Limits/Day		Range Limits/Day	
	Low	High	Low	High	Low	High	Low	High
Lower	$1,000	$3,000	$2,000	$7,500	$3,000	$15,000	$5,000	$25,000
Medium	$2,000	$30,000	$4,000	$100,000	$6,000	$200,000	$10,000	$335,000
High	$4,000	$125,000	$8,000	$300,000	$12,000	$625,000	$20,000	$1,000,000

Questions