
# Operational Security in Software Development

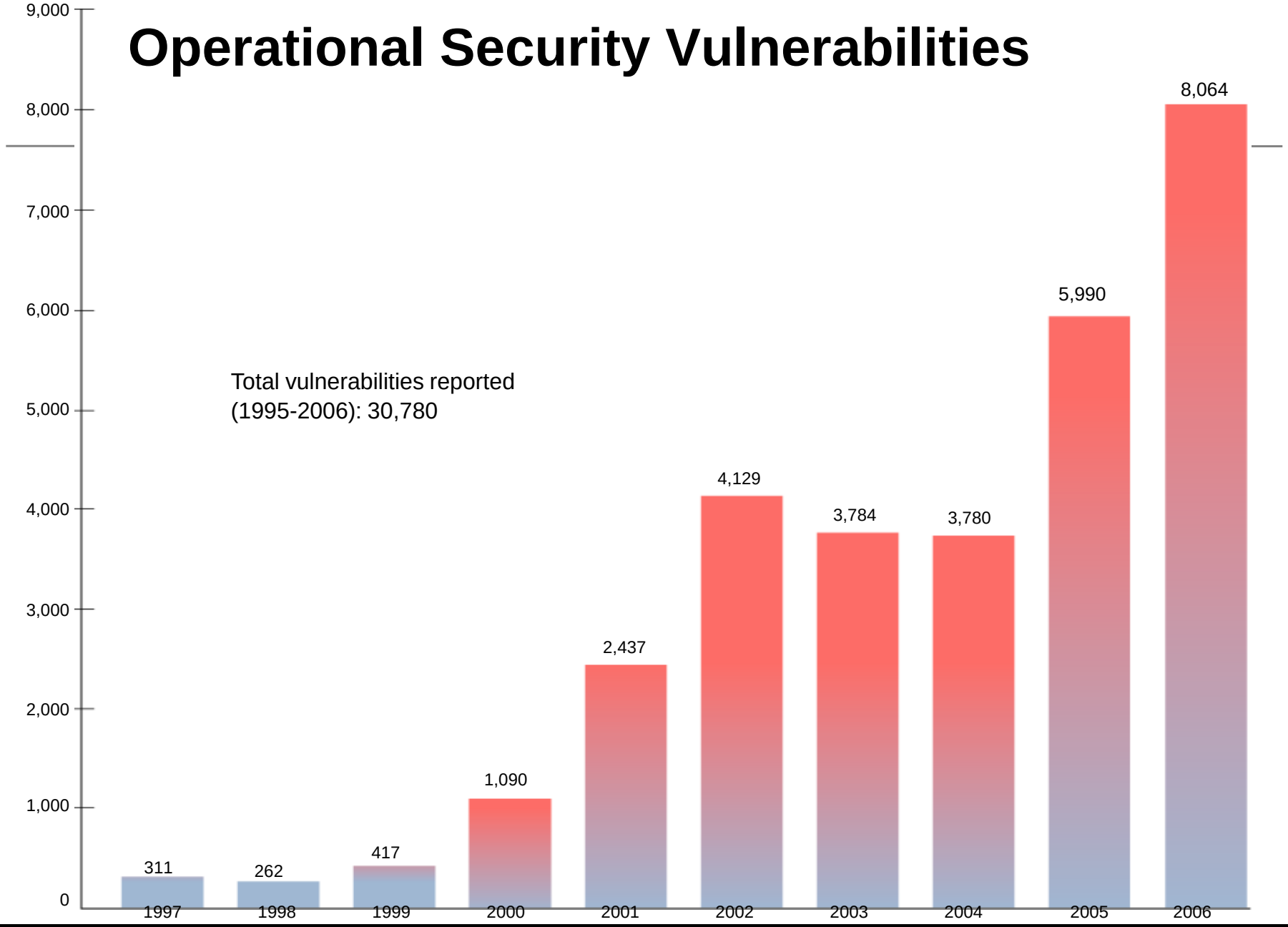
Carol Woody, Ph.D.

# Presentation Contents

- Embedding Security in the Current System Development Life Cycle (SDLC)
- SEPG 07 Security Track Panel Report – Process Improvement Should Link to Security
- The Next Wave is Already Here – Assurance for Systems of Systems


# Embedding Security in the SDLC

# Operational Security Vulnerabilities



# Top Web Application Vulnerabilities

Unvalidated input

Broken access control

Broken authentication and session management

Cross site scripting

Buffer overflow

Injection flaws

Improper error handling

Insecure storage

Application denial of service

Insecure configuration management

Reference: Open Web Application Security Project (OWASP), www.owasp.org/index.php/Top_Ten

# Problems in the Application Layer

SECURITY REQUIREMENTS: Systems are built to requirements and security (as well as quality in general) is frequently poorly defined.

OWNERSHIP: Stakeholders do not champion security. Security is frequently in conflict with other qualities (performance, safety, flexibility) and trade-off decisions are poorly articulated.

CODE QUALITY: Software developers average a defect every 7-10 instructions.

TESTING:

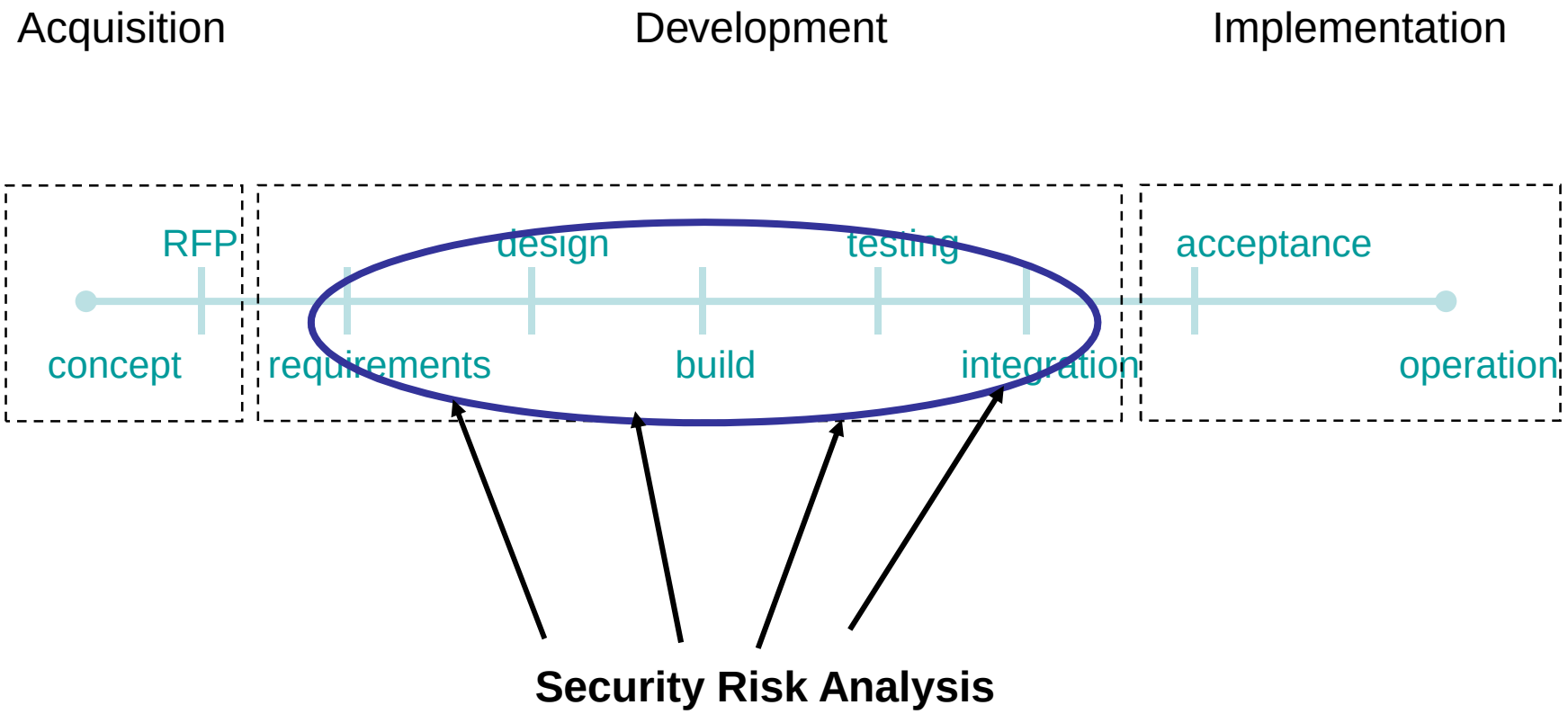
- Insufficiently planned,
- under funded and limited by project cost and schedule,
- infrequently includes evaluation of “what can go wrong”, and integration areas (component interactions), and
- rarely simulates live usage.

ACCREDITATION AND CERTIFICATION: Focused primarily on software components and not on integrated results

# Everything Has to Work Together


Establish effective operational security risk analysis during development

# Check and Adjust the Operational Security Balance During Development



# Operational Security Balance

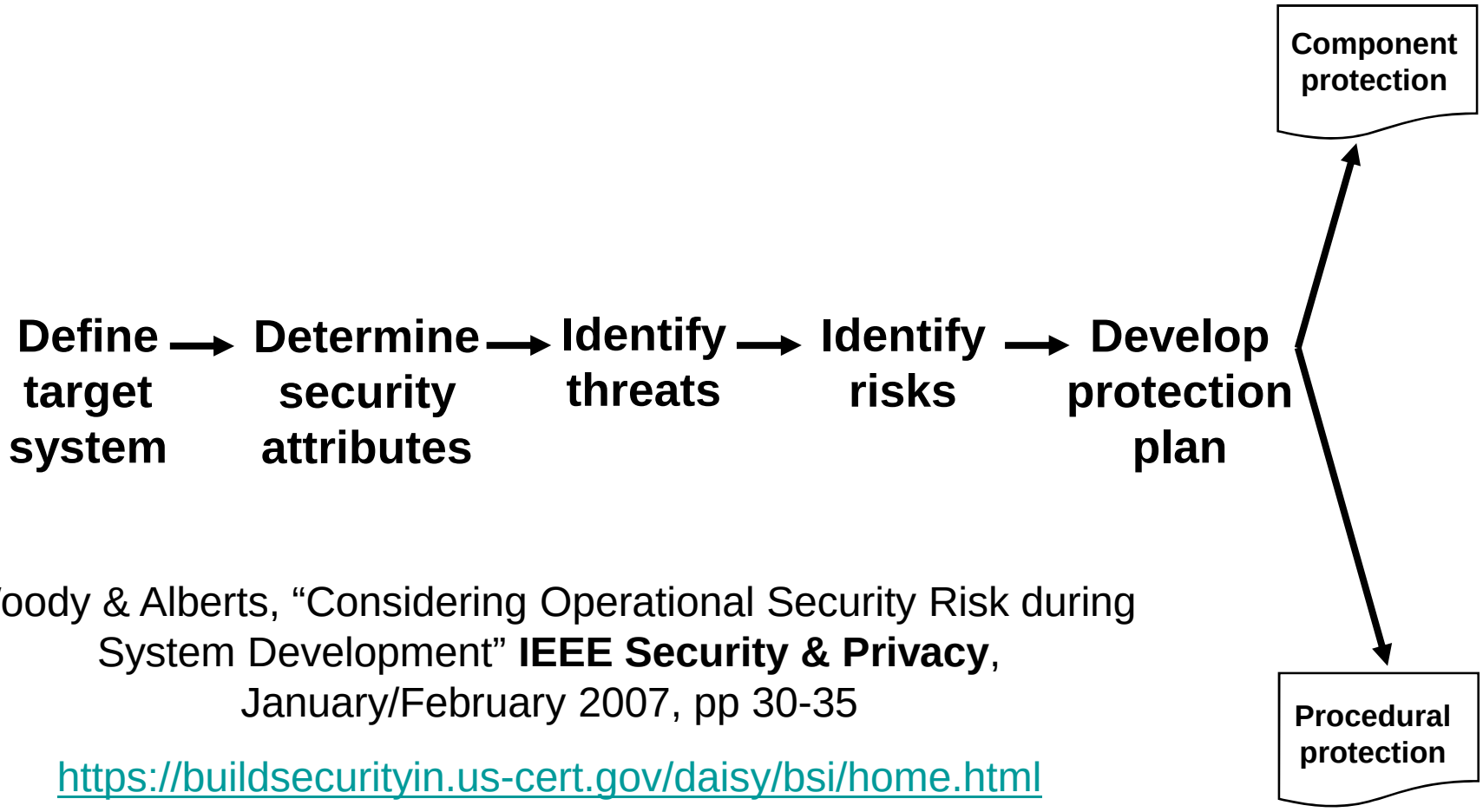
Adjust the security balance at key milestones during a project to accommodate the realities of change

Waiting until the end (ST&E) is too late

Blend information from everything that must work together to validate that the operational security is sufficient

- Application security and functional design
- Tools and products selected for implementation
- Operational security practices
- Organizational policies and practices
- User security practices

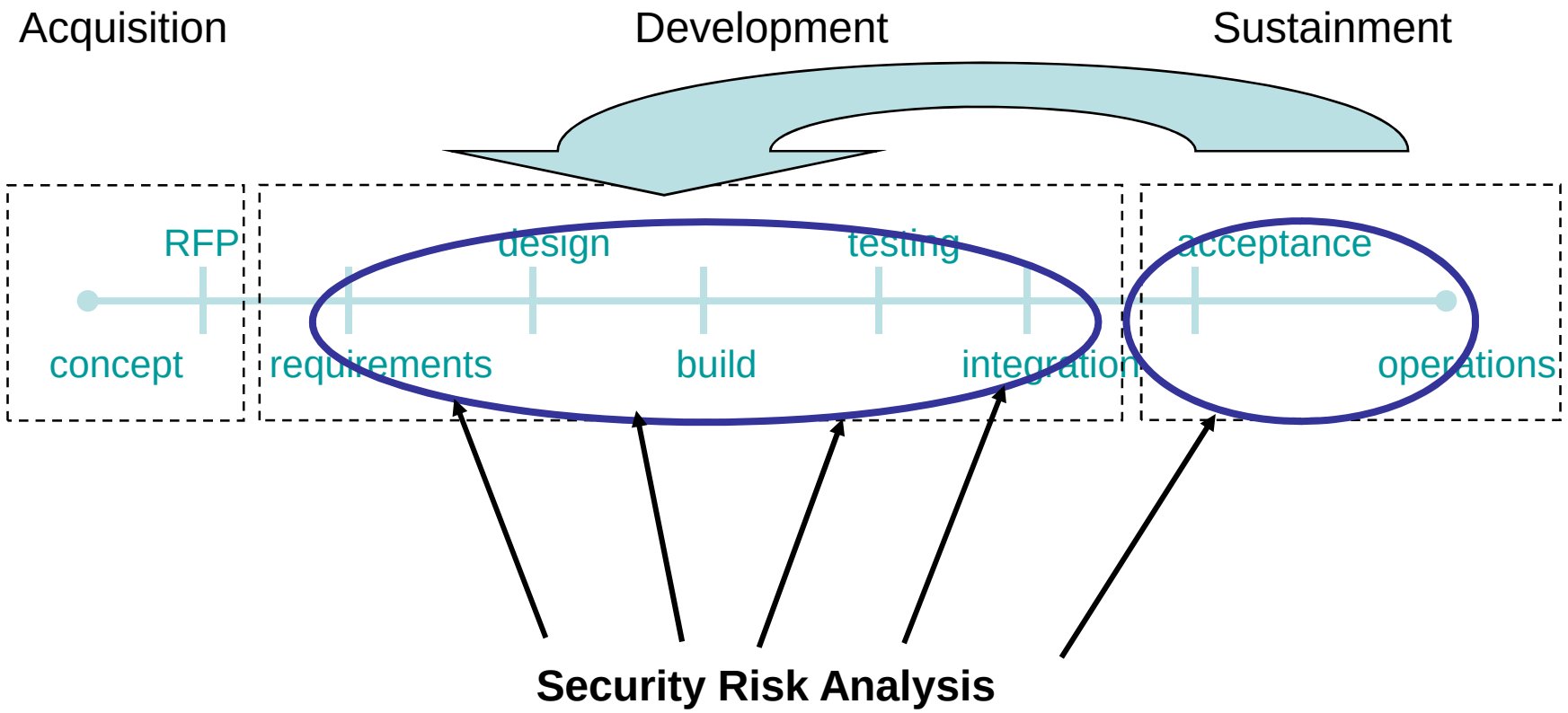
# Steps for Security Risk Analysis of a System During Development




Woody & Alberts, “Considering Operational Security Risk during System Development” **IEEE Security & Privacy**, January/February 2007, pp 30-35

https://buildsecurityin.us-cert.gov/daisy/bsi/home.html

# Monitor Security Risk Balance Across the SDLC and into Sustainment




# Process Improvement Should Link to Security

Report from SEPG 07 Panel

Software Engineering Institute | Carnegie Mellon

# Getting the Conversations Started

## SEPG 2007 Security Track

- Focused security speakers on process links
- Panel with audience participation (range of perspectives)
  - DHS
  - Motorola
  - Booz Allen Hamilton
  - SEI CERT
- Conference interest was much greater than in the past
  - Attendance doubled from prior years, but still not high
  - Extensive informal feedback around this topic

## SEPG 2008 Security Track – consider being a presenter

# SEP07 Selected Audience Responses

- Process is the security enabler – getting the right people at the right place at the right time
- Security is a separate discipline that must collaborate with existing process areas but should not be assumed to fully blend with existing processes
- Security must be a part of the normal organizational information flow and not an add-on when it suits
- Lots of awareness of the need to do something but little understanding of how to go about it effectively
- Do not need more standards and regulations – need to implement what we have throughout the life cycle
- Evidence that an organization is meeting a standard should be part of the process measurement
- Existing practices support security but do not promote it
- Security capabilities must be assessed to assure improvement to meet the needs of today's environment.

# Security Birds of a Feather at SEPG07

Need:

- Clear definition of what trying to achieve
- Identification of gaps with current model
- Definition of additions needed
- Making security mandatory

Have:

- One organization's application of selected standards in specific projects
- Projected extensions to existing practices and a few additional practices to address one organization's needs

Moving Forward:

- Publication of security track summary report for broader distribution (draft available for 8-7-7)
- Continued expansion of Build Security In - Call for Authors and Reviewers https://buildsecurityin.us-cert.gov/daisy/bsi/900.html


# The Next Wave is Already Here

## Assurance for Systems of Systems

Software Engineering Institute | Carnegie Mellon

# Solving Current Security Issues is Not Enough

Organizations require integrating system and people activities across a *constantly evolving* mix of changing systems and people to meet capability needs

Increased reliance on shared technology/services requires establishing *operational trust* among systems, software components, and services.

Establishing and maintaining organizational capabilities requires *traceability* between technical decisions and capability requirements

Reliability of organizational capabilities can be affected by the *interactions* of software systems, hardware systems, and human operations

Organizations must continuously adjust technology and people to meet immediate critical ad hoc needs. This level of *flexibility* contributes to system *fragility*.

# Operational System of System Challenges

## Characteristics of large networked systems

- Heterogeneous, potentially inconsistent, and changing elements (hardware, software, systems)
- Continuous evolution of functionality and usage (perpetual beta)
- Erosion of the people/system boundary (each influences the other)
- Independently developed and managed systems integrated into a system of systems

## Mitigating component failure is not sufficient

- Increasingly failures result from a group of errors (operator, unexpected software state, and user) that can be handled individually but not collectively
- Increasing dependencies among development, deployment, and operations (complexity hides risks until deployment)

## Complexity is addressed with segregation, decomposition, and simplifying assumptions

- Hides risks making them difficult to observe until deployment.
- Conflicting assumptions among components leads to mismatches

# Complex Failure: 2003 Power Blackout [1]

On August 14, 2003, approximately 50 million electricity consumers in Canada and the northeastern U.S. were subject to a cascading blackout. There was not a single cause for this event.

The failures occurred over a four hour period

- Tree trimming procedures were not followed
- Race condition disabled alarm system that provided the only effective means for grid operators to identify problems. The corruption of the data stream caused the backup server to fail also.
- Alarm subsystem could only be restarted by restarted full control system – sixty minutes. Without the aid of the alarms, grid operators were not aware of affects of the loss of the lines.
- IT confused by initial symptoms. Did not notify grid operations of the alarm subsystem failure.

# Complex Failure: 2003 Power Blackout [2]

The power failure demonstrates the need for a system assurance case to include not only the computing systems but also business operations, training, and IT operations.

- Issues with operator training managing emergency conditions.
- Operational and system analysis should have identified a system requirement to automatically notify grid controllers when the alarm system or other critical subsystems fail.
- An analysis of software faults in addition to hardware faults might have lead to a business continuity requirement to be able to restart a service such as alarm notification without having to restart the entire system.

The events leadings to the blackout had non-malicious intent, but could have been exploited especially with some insider knowledge.

# Expect Mismatches

Inconsistencies (mismatches) must be assumed as we compose systems.

- Components are developed at different times with variances in technology and expected usage
- A system will not be constructed from uniform parts: always some misfits as the components are extended and repaired

Human interactions frequently bridge mismatched components

- Erosion of the people/system boundary --- people are part of the system
- Failure response and its effects on component interactions is not designed – steps to recognize, resist, and recover must look across the interactions

High Risk for Power Grid: Lack of real-time analysis of component failure induced by mismatches and delayed human intervention to reset the systems and restore component functionality led to cascading operational failure

# New Analysis Approach Required

Existing mechanisms do not link technology to complex organizational activities:

- focus on a single system or highly integrated group of systems
- consider only single system architecture trade-off decisions
- do not consider business process dependencies on multiple systems
- do not identify the relationship of a component's properties to the completion of a business process that crosses systems
- consider only a point in time for usage, attack patterns and system configuration – reactive

Systems of systems are developed to provide organizational capabilities and assurance must be evaluated for a mission that relies on the system of systems

# Assurance for Systems of Systems

**Research Challenge:** How do we consider assurance as we acquire, design, build, and compose software components and systems to function within a system of systems to provide for the survivability of a mission?

**Research Projects:** JBMC2, Airborne Networks, FCS, VA (planned)

**Research Results:**

Survivability Analysis Framework (SAF): Approach for constructing an operational model of an end-to-end mission for stakeholders to reason about failures and identify survivability gaps

# SAF: Building the Operational Model

## Mission Thread Approach

An end-to-end view of the groups of activities (steps) that link people and resources (systems, software, connectivity, training, policy, practices, experience, authentication, authorization, etc.) to complete an organizational goal

- JBMC2, Airborne, and FCS: Time-sensitive targeting
- Airborne (in process): Close Air Support
- VA (planned): patient scheduling, patient enrollment

## Mission Thread Analysis

Identification of the linkages of assurance responsibilities among people and resources to show the impact of failures in one area on mission

Identification of stresses (aka threats) that would be the visible indicators of possible failures

# Sample Stresses (Threats) to a Mission

## Interaction (data)

- Missing, Inconsistent, Incorrect, Unexpected, Incomplete, Unintelligible, Out of date, Duplicate

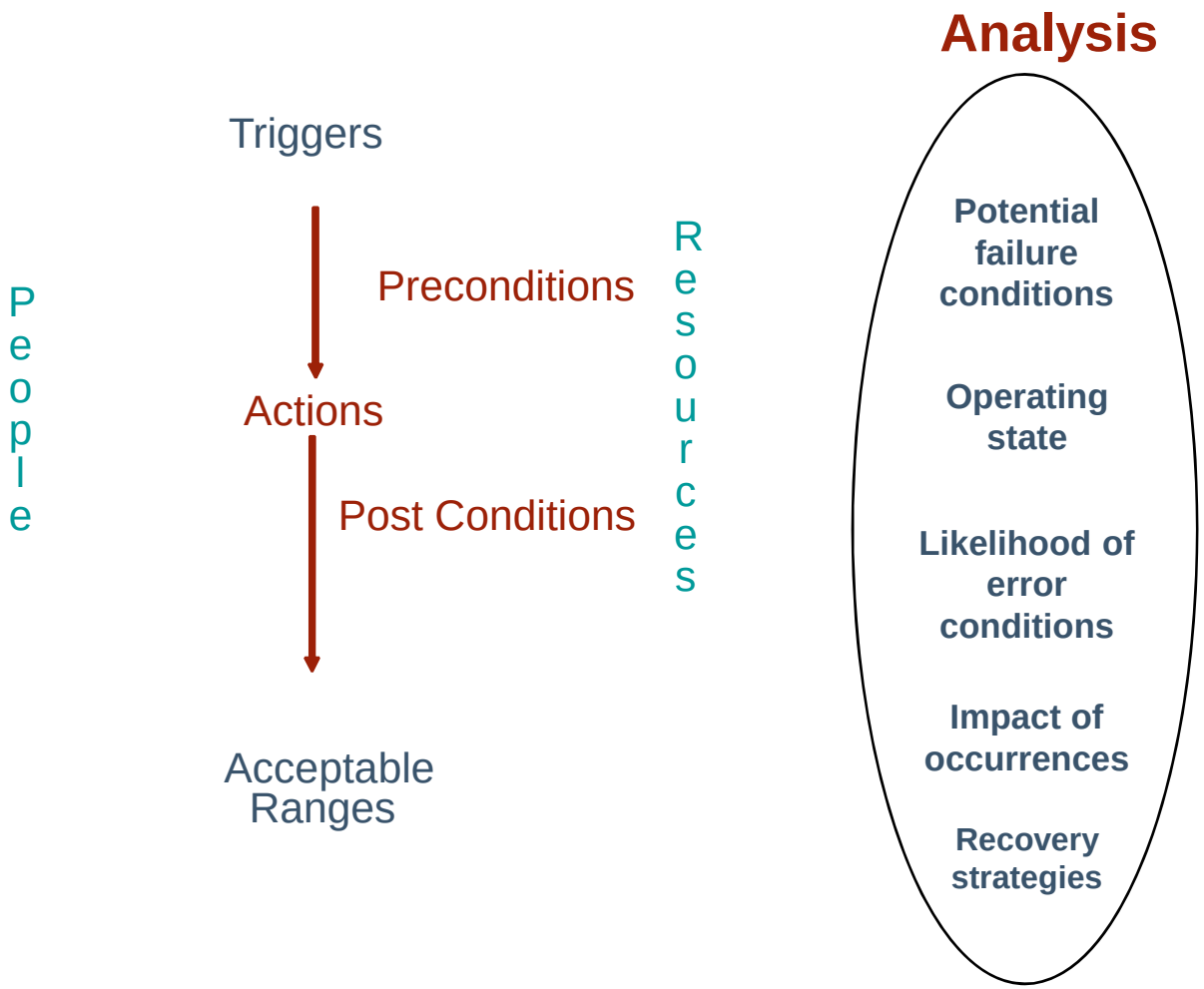
## Resource

- Insufficient, Unavailable, Excessive, Latency, Inappropriate, Interrupted

## People

- Information overload, Analysis paralysis, Fog of war, Distraction (rubber necking), Selective focus (only looking for positive reinforcement), Diffusion of responsibility (e.g. not my job), Spurious correlations

# Mission Step Analysis



# Potential Survivability Analysis Outcomes

From initial use of the framework:

- Potential points of failure (stress analysis)
- Survivability gaps (step interactions)
- Mitigation strategies for a business process
- Gaps in current component requirements
- Better quality specifications for component requirements
- Better quality specifications for shared services

Application of the framework to a business thread periodically as systems and services change:

- Changes in survivability capabilities over time
- Opportunities for survivability improvement

# Lessons Learned so Far

Exposing developers to the operational realities increases consideration of those issues during design and implementation.

Operational personnel view this as an effective means of communicating their challenges to management

Characterizing the interactions of people, resources, and activities provides a structured way of describing the complexity

Identification of potential failures requires detail knowledge of how activities are actually performed

Analysis steps are not sufficiently structured for repeatability

# Assurance for Systems of Systems

Establish an approach for the construction of assurance cases and identification of evidence that assurance is provided based mission thread analysis data

- Claims, arguments, and evidence
  - Addresses physical system, software, and operational procedures
- Addresses potential threats leading to
  - Unauthorized disclosure, modification, or loss of sensitive information
  - Denial of access
  - Unauthorized actions by authorized users

Technical note to be published Fall 2007 (Ellison, Goodenough, Weinstock, Woody)