SECURITY METRICS for Software Development in a Emerging Economy

Sanjay Misra
Prof of Computer Engineering
Head ,Department of Cyber Security Science
Federal University of Technology,Minna,Nigeria

A Main Challenge to Cyber Security Science

"*A major difference between a "well developed" science such as physics and some of the less "well-developed" sciences such as psychology or sociology is the degree to which things are measured.*"

: Fred S. Roberts

Metrics Defined

- The National Institute of Standards and Technology (NIST) define metrics as tools designed to facilitate decision-making and improve performance and accountability through collection, analysis, and reporting of relevant performance-related data.
- Metrics are simply a standard or system of measurement.

Security Metrics Defined

- Therefore, security metrics is a standard for measuring security
- Security metrics has become a standard term when referring to security level, security performance, security indicators or security strength

Why Security Metrics?

- Capability Maturity Model for Software Engineering used to measure quality fails to address security issues
- Consequently, security flaws are identified only at the later stages of the application lifecycle
- And thus much greater cost to fix and high maintenance cost
- With the Emerging ICT-based Economy, there is greater need for Security Metrics fully Integrated into Software Developmental Stages for Secured Deliverables

Security Metrics Benefits

- The benefits involves:
- risk management,
- software security assurance,
- security testing,
- security performance,
- adaptive security monitoring and
- intrusion detection and prevention

Categories of Security Metrics

- **Strategic support** :- Decision making, such as program planning, resource allocation, and product and service selection.
- **Quality assurance** :- Elimination of vulnerabilities, particularly during code production
- **Tactical oversight** :- Monitoring and reporting of the security status

Aspects of Security Measurement

- Correctness and Effectiveness
- Leading Versus Lagging Indicators
- Organizational Security Objectives
- Qualitative and Quantitative Properties

Secure Software means

- *Secure software cannot be intentionally subverted or forced to fail.*
- *It remains correct and predictable in spite of intentional efforts to compromise that dependability.*

Secure Software means…

- *Continue operating correctly in the presence of most attacks*
- *Isolate, contain, and limit the damage resulting from any failures*

Attributes of Secure Software

- *Exploitable faults and other weaknesses are avoided*
- *The likelihood is greatly reduced or eliminated that malicious developers can intentionally implant exploitable faults and weaknesses or malicious logic into the software.*
- *Attack-resistant or attack-tolerant, and attack-resilient.*
- *The interactions among components within the software-intensive system, and between the system and external entities, do not contain exploitable weaknesses.*

Metrics Vs Measurement

Measurement

- Measurements provide single-point-in-time views of specific, discrete factors
- Measurements are generated by counting
- Measurements are objective raw data

Metrics

- Metrics are derived by comparing to a predetermined baseline two or more measurements taken over time
- Metrics are generated from analysis
- Metrics are either objective or subjective human interpretations of those data

GOOD METRICS

- Good metrics are those that are SMART, i.e.
- Specific,
- Measurable,
- Attainable,
- Repeatable,
- Time-dependent

Metric Types/examples

Process Metrics

Information about the processes themselves. Evidence of maturity.

→

Examples

- Secure coding standards in use
- Avg. time to correct critical vulnerabilities

Vulnerability Metrics

Metrics about application vulnerabilities themselves

→

Examples

- By vulnerability type
- By occurrence within a software development life cycle phase

Management

Metrics specifically designed for senior management

→

Examples

- % of applications that are currently security "certified" and accepted by business partners
- Trending: critical unresolved, accepted risks

Examples of Application Security Metrics

Process Metrics

- Is a SDL Process used? Are security gates enforced?
- Secure application development standards and testing criteria?
- Security status of a new application at delivery (e.g., % compliance with organizational security standards and application system requirements).
- Existence of developer support website (FAQ's, Code Fixes, lessons learned, etc.)?
- % of developers trained, using organizational security best practice technology, architecture and processes

Management Metrics

- % of applications rated “business-critical” that have been tested.
- % of applications which business partners, clients, regulators require be “certified”.
- Average time to correct vulnerabilities (trending).
- % of flaws by lifecycle phase.
- % of applications using centralized security services.
- Business impact of critical security incidents.

Examples of Application Security Metrics

Vulnerability Metrics

- Number and criticality of vulnerabilities found.
- Most commonly found vulnerabilities.
- Reported defect rates based on security testing (per developer/team, per application)
- Root cause of "Vulnerability Recidivism".
- % of code that is re-used from other products/projects*
- % of code that is third party (e.g., libraries)*
- Results of source code analysis**:
  - Vulnerability severity by project, by organization
  - Vulnerabilities by category by project, by organization
  - Vulnerability +/- over time by project
  - % of flaws by lifecycle phase (based on when testing occurs)

Value of Security Metrics

- Accepted Management principle says that an activity cannot be managed if it cannot be measured.
- Metrics can be an effective tool for security managers

Security Managers…

- Security managers can use metrics to
- discern the effectiveness of various components of their security programs,
- the security of a specific system,
- product or process,
- and the ability of staff or departments within an organization to address security issues for which they are responsible.
- identify the level of risk in not taking a given action, and in that way provide guidance in prioritizing corrective actions.

Security Managers…

- to raise the level of security awareness within the organization
- With knowledge gained through metrics, security managers can better answer hard questions from their executives and others, such as:
- Are we more secure today than we were before?
- How do we compare to others in this regard?
- Are we secure enough?

Security Metrics Development

- Define the metrics program goal(s) and objectives
- Decide which metrics to generate
- Develop strategies for generating the metrics
- Establish benchmarks and targets
- Determine how the metrics will be reported
- Create an action plan and act on it, and
- Establish a formal program review/refinement cycle

Security-aware Software Industry

- For the software industry, the key to meeting demand for improved security, is to implement repeatable processes that reliably deliver measurably improved security
- Thus, there must be a transition to a more stringent software development process that greatly focuses on security
- Goal: minimize the number of security vulnerabilities in design, implementation, and documentation
- Identify and remove vulnerabilities in the development lifecycle ***as early as possible!!!***

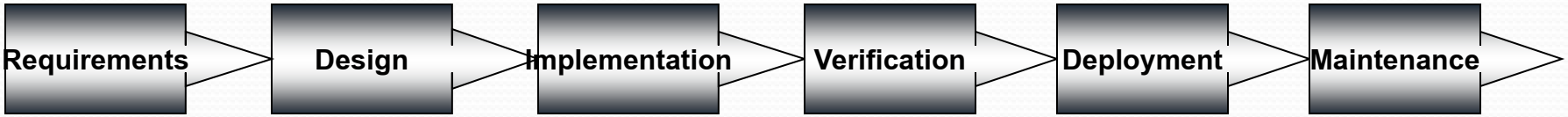
Secure Software Development

Three essential components

- Repeatable process
- Engineer Education
- Metrics and Accountability

- SDL – Secure Development Lifecycle
  - Used along with traditional/current software development lifecycle/techniques in order to introduce security at every stage of software development

Secure Dev Lifecycke - PHASES



SDL – Requirements Phase

- Develop Security Requirements
  - Security Requirements of a system/application must be developed along with any other requirements requirements (e.g. functional, legal, user, etc)
- Risk analysis
  - Identify all the assets at risk
  - Identify all the threats
- Develop security policies
  - Used as guidelines for requirements
- Develop security metrics

SDL – Design Phase

- At this stage all design decisions are made, about
  - Software Architecture
  - Software components
  - Programming languages
  - Interfaces
  - …
- Develop documentation
- Confirm that all requirements are followed and met

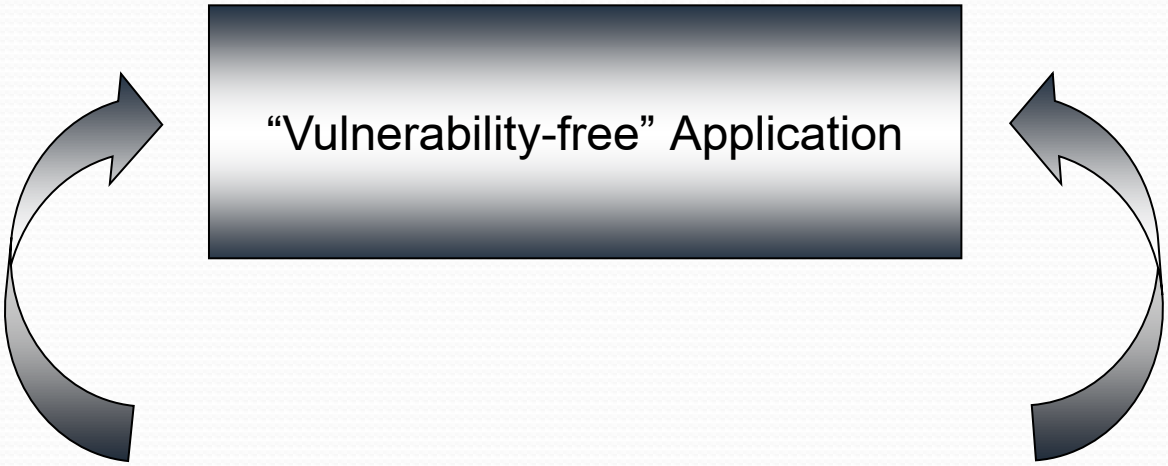
SDL – Design Phase…

- Treat Models
- Input Data Types
- Security Use Cases
- Security Architecture
- Defense in Layers / Separate Components / Least Privilege
- Tool
  - SecureUML – Secure Unified Modeling Language

SDL – Implementation Phase

- This is the stage where coding is done.
- To produce secure software
  - Coding Standards
  - Centralized Security Modules
  - Secure builds and configurations
    - Known security vulnerabilities - use good programming practices. Be aware of
      - Race conditions
      - Buffer overflow
      - Format string
      - Malicious logic
      - …
- Follow Design & Develop Documentation

SDL – Implementation Phase…



Design and implementation of security features.
From the Requirements

Robust Programming Practices
Good design and coding practices

SDL – Verification Phase

- Testing of the code developed in the previous stage
- Cleared security tests
- Security vulnerability tracking
- Code Reviews
- Documentation

SDL – Release Phase

- Secure Management Procedures
- Monitoring Requirements
- Security Upgrade Procedures

SDL – Response Phase

- Causes:
  - Costumer feedback
  - Security incident details and vulnerability reports
  - …
- Types of maintenance
  - Need to introduce new functionality
  - Need to upgrade to keep up with technology
  - Discovered vulnerability

Reality

- Every security vulnerability / flaw overlooked in an earlier phase will end-up at later phase[s]

- Resulting into greater
  - Cost
  - Time

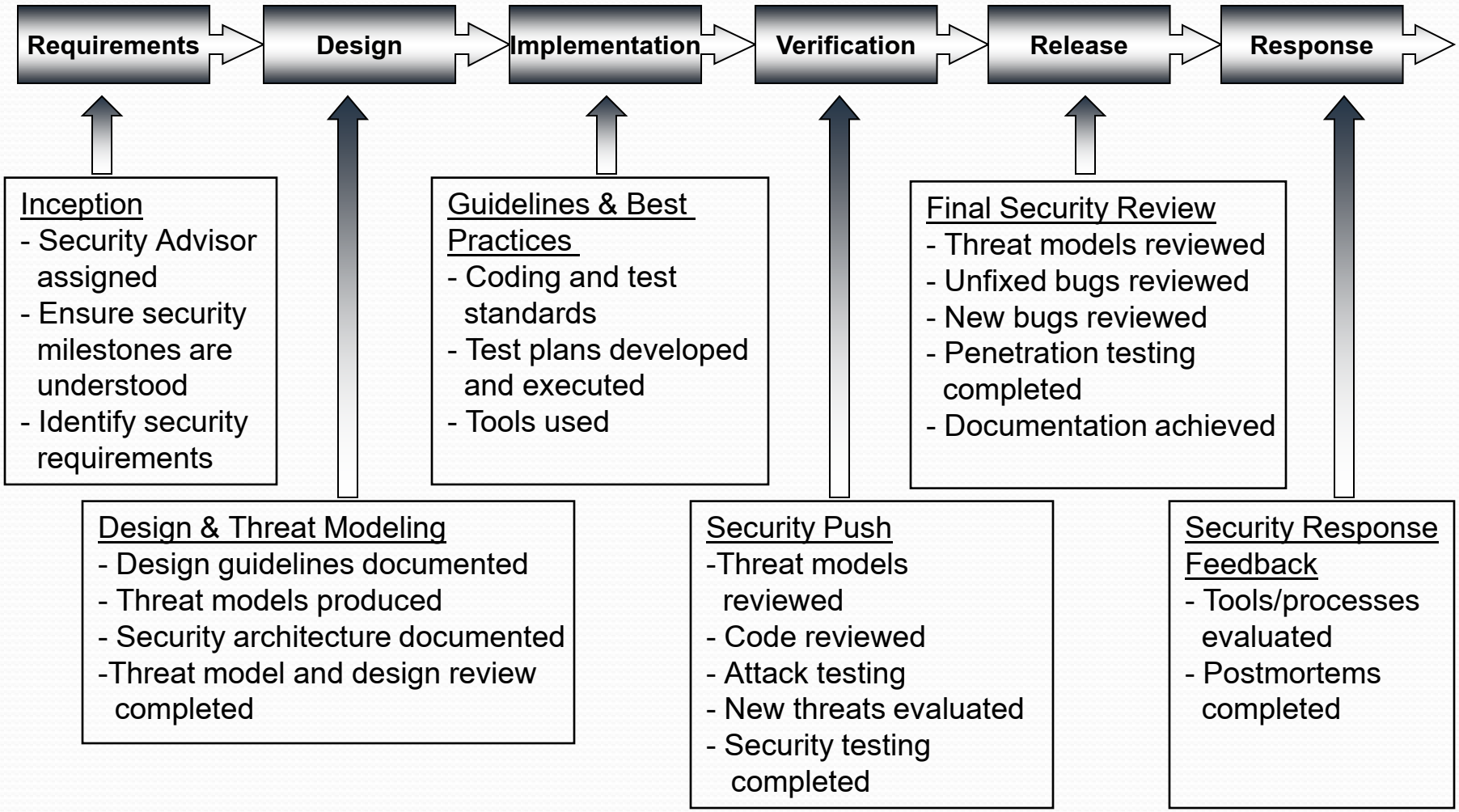
  of the software development and/or maintenance

Microsoft – Case Study

$SD^3 + C$

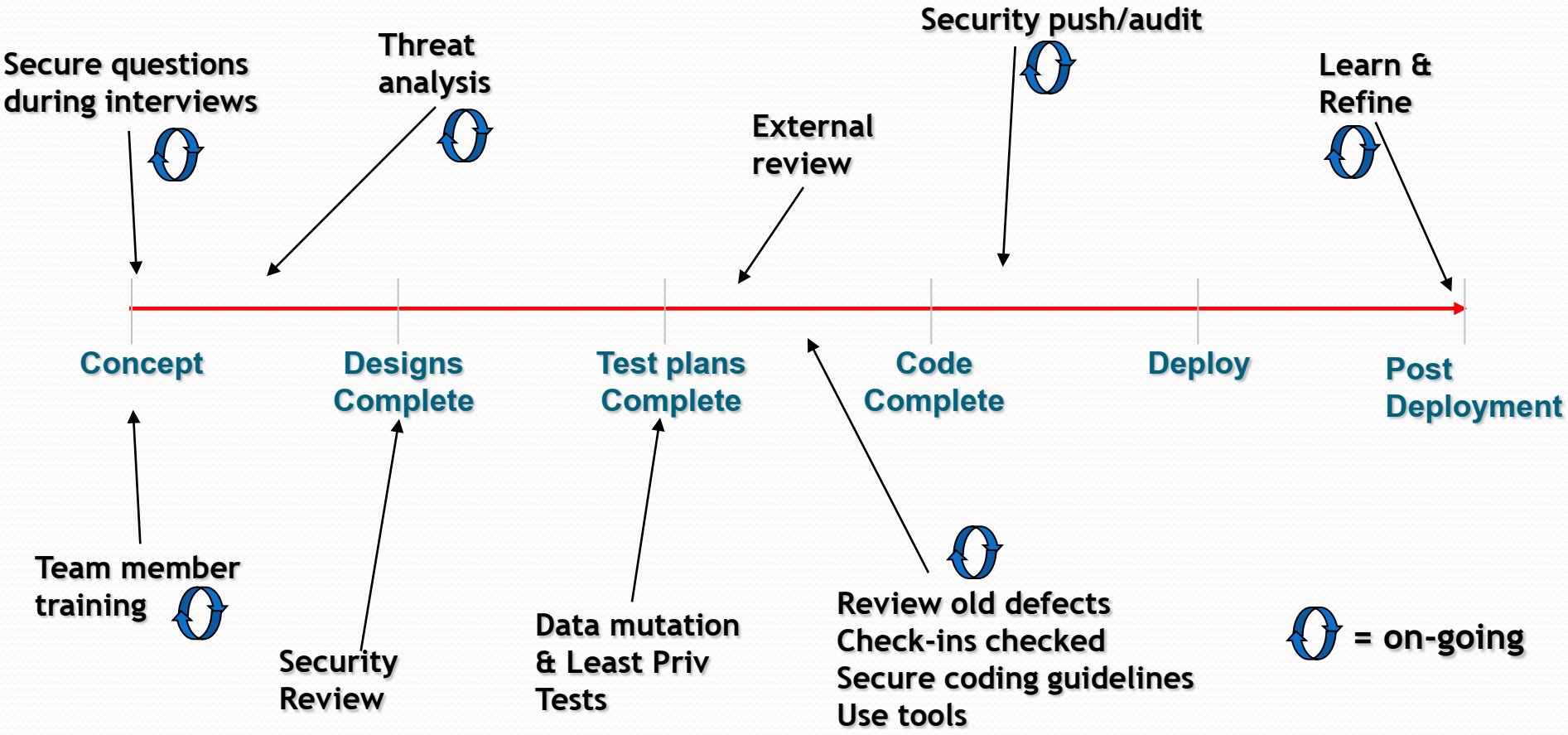
- Secure by Design
  - Software designed and implemented to "protect" itself and its information
- Secure by Default
  - Accept the fact that software will not achiever perfect security
  - To minimize the harm when vulnerabilities exploited, software's default state should promote security (ex. least necessary privileges)
- Secure in Deployment
  - Software accompanied by tools and guidance to assist secure use

SDL @ Microsoft



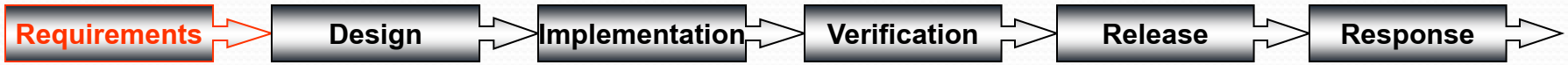
Opportunities for Metrics - Secure Development Life Cycle (SDL)

Were software assurance activities conducted at each lifecycle phase?



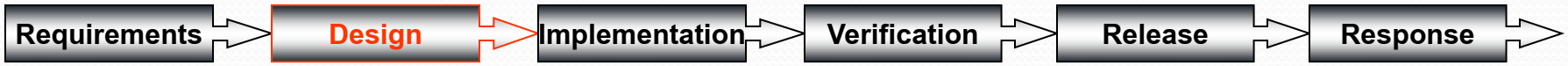
Source: Microsoft

SDL – Requirements Phase @ Microsoft



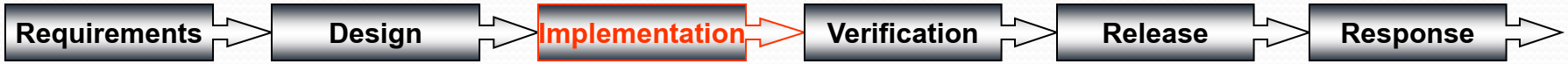
- Product and central security teams assign "security buddy" – security advisor
  - Point of contact / resources / guide
  - Review plans / recommendations / resources
- Product team considers
  - How security will be integrated into the development process
  - Key security objectives
- Documentation

SDL – Design Phase @ Microsoft



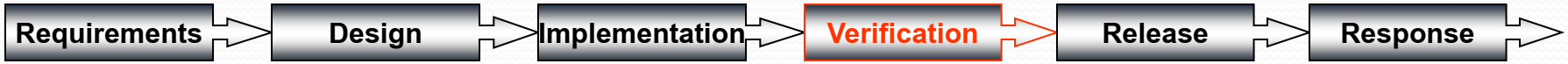
- Define security architecture and design guidelines
- Document the elements of the software attack surface
- Conduct threat modeling
- Define supplemental ship criteria

SDL – Implementation Phase @ Microsoft



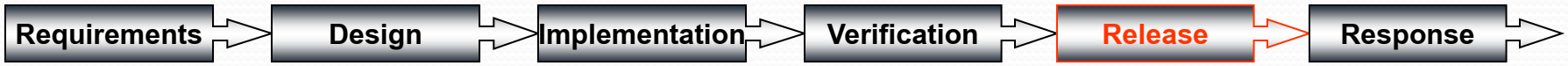
- Apply coding and testing standards
- Apply fuzzing tools
  - Supplies structured but invalid inputs
- Apply static-analysis code scanning tools
- Conduct code reviews

SDL – Verification Phase @ Microsoft



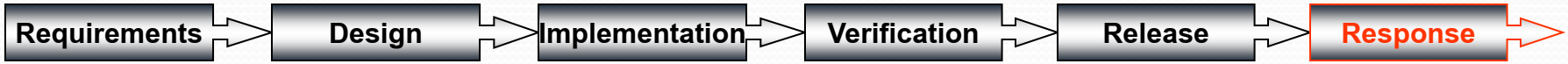
- "Beta" testing stage
- "Security push"
  - security code reviews beyond ones completed in implementation phase
  - Testing of high priority code
  - Trying to "break" the code

SDL – Release Phase @ Microsoft



- During the release, software is subject to Final Security Review [FSR]
- The goal of FSR is to determine whether, from security viewpoint, the software is ready to be delivered to costumers
- Not pass / fail
- Goal is to find every remaining security vulnerability in software
  - If found, revisit all the preceding phases and fix the root problem
- Conducted by central security team

Microsoft



- Despite use of SDL, resulting software is not vulnerability free; and even if it could be so, new attacks would be possible
- Evaluation of reports
- Development of patches and security updates

SDL @ Microsoft

- Mandatory Application of the SDL
- Mandatory Education
- Metrics for Product Teams
- The Central Security Team

Thank you very much

Questions?????

Contact Information:

mail: ssopam@gmail.com

smisra@futminna.edu.ng

cell number:07030851086