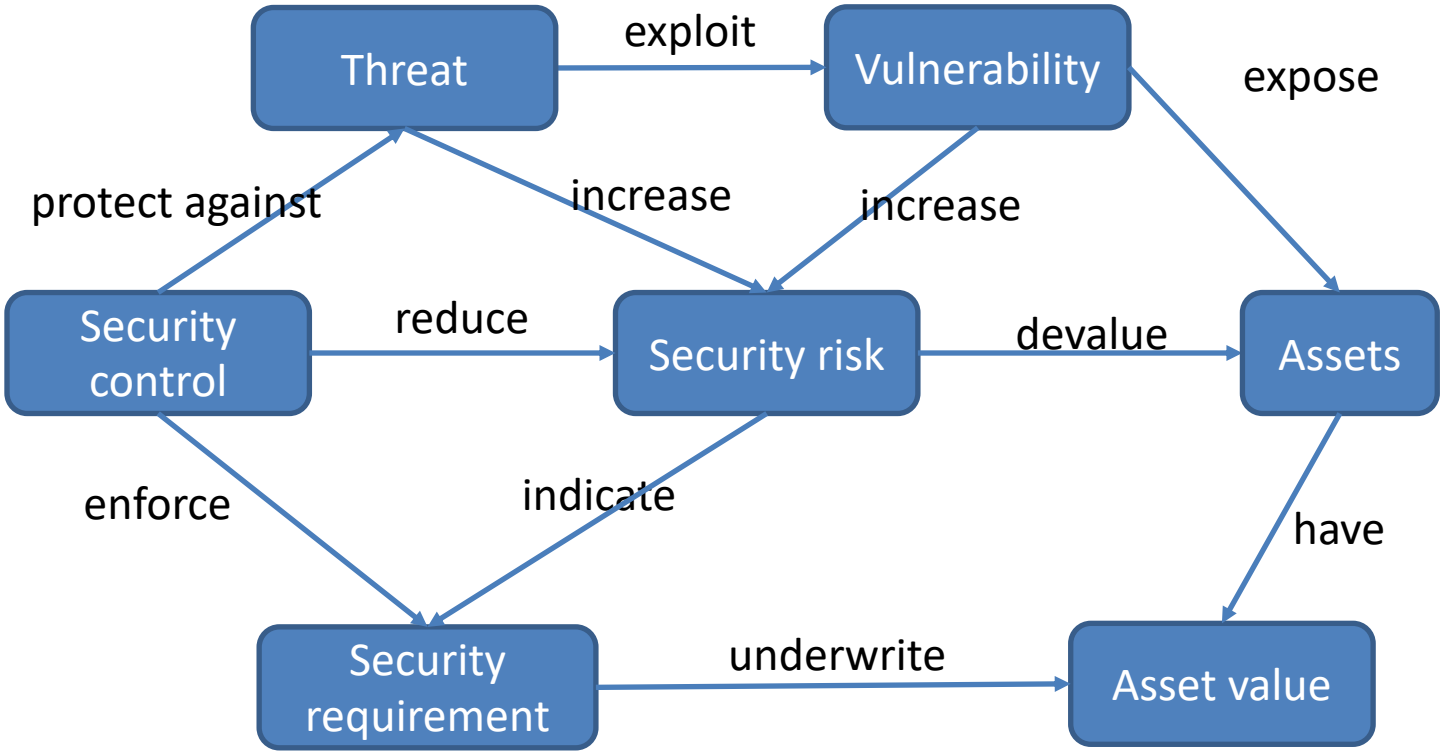
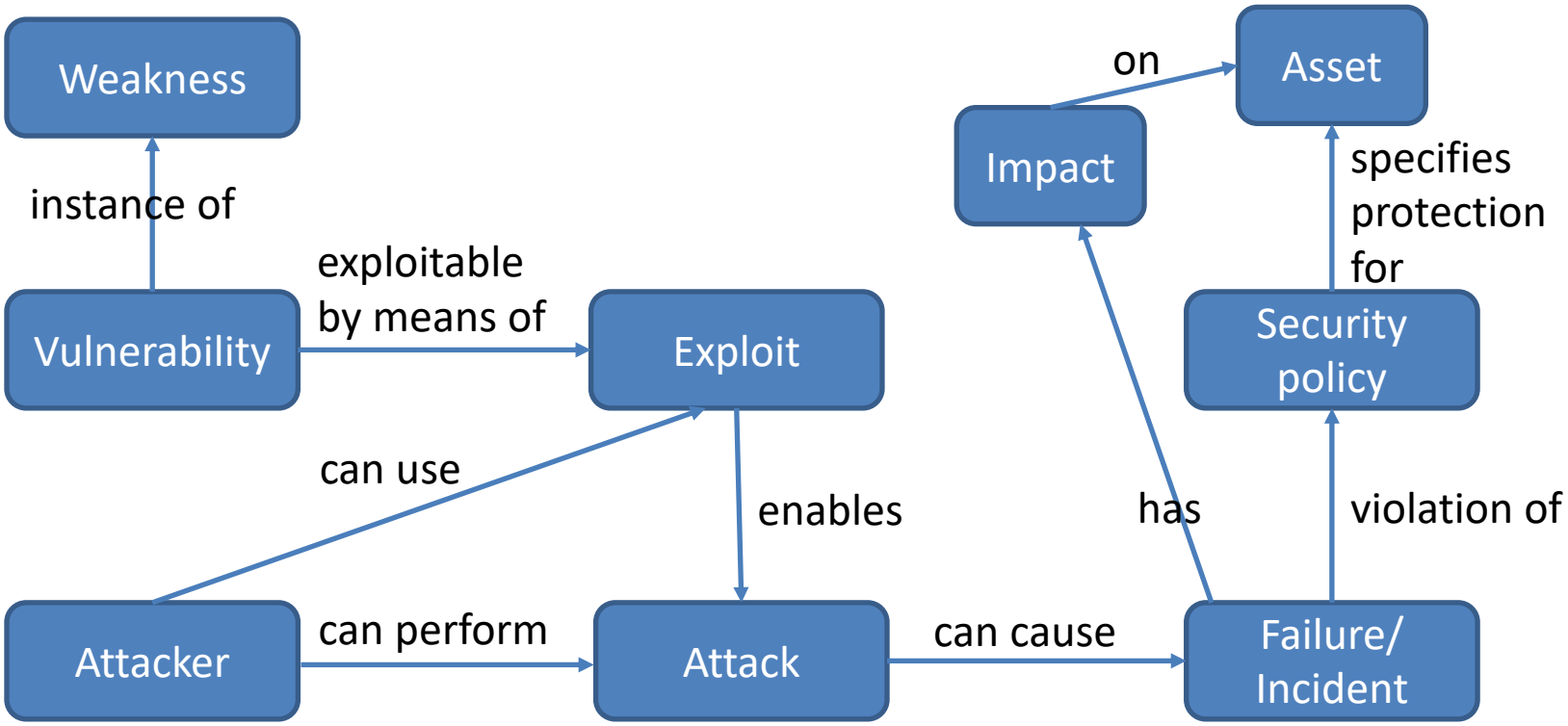
# Security Assessment Techniques: Definitions and Classification

# Recalling Some (Cyber)security Key Concepts



# Recalling Some (Cyber)security Key Concepts



# Vulnerability

- Bug or flaw in
  - Design
  - Specification
  - Implementation (hardware, software)
  - Configuration

  of a specific system component

can also be a third-party library (exposed logic)

  which could be exploited to compromise system security

# Vulnerability: Events

- Vulnerability is created (creation)
- Vulnerability is discovered (discovery)
- Vulnerability is disclosed (disclosure)
- Exploit is created (exploit)
- Exploit is disclosed (exploit disclosure)
- Patch is created (patch)
- Patch is made available (patch publication)
- Patch has been applied to most of the systems

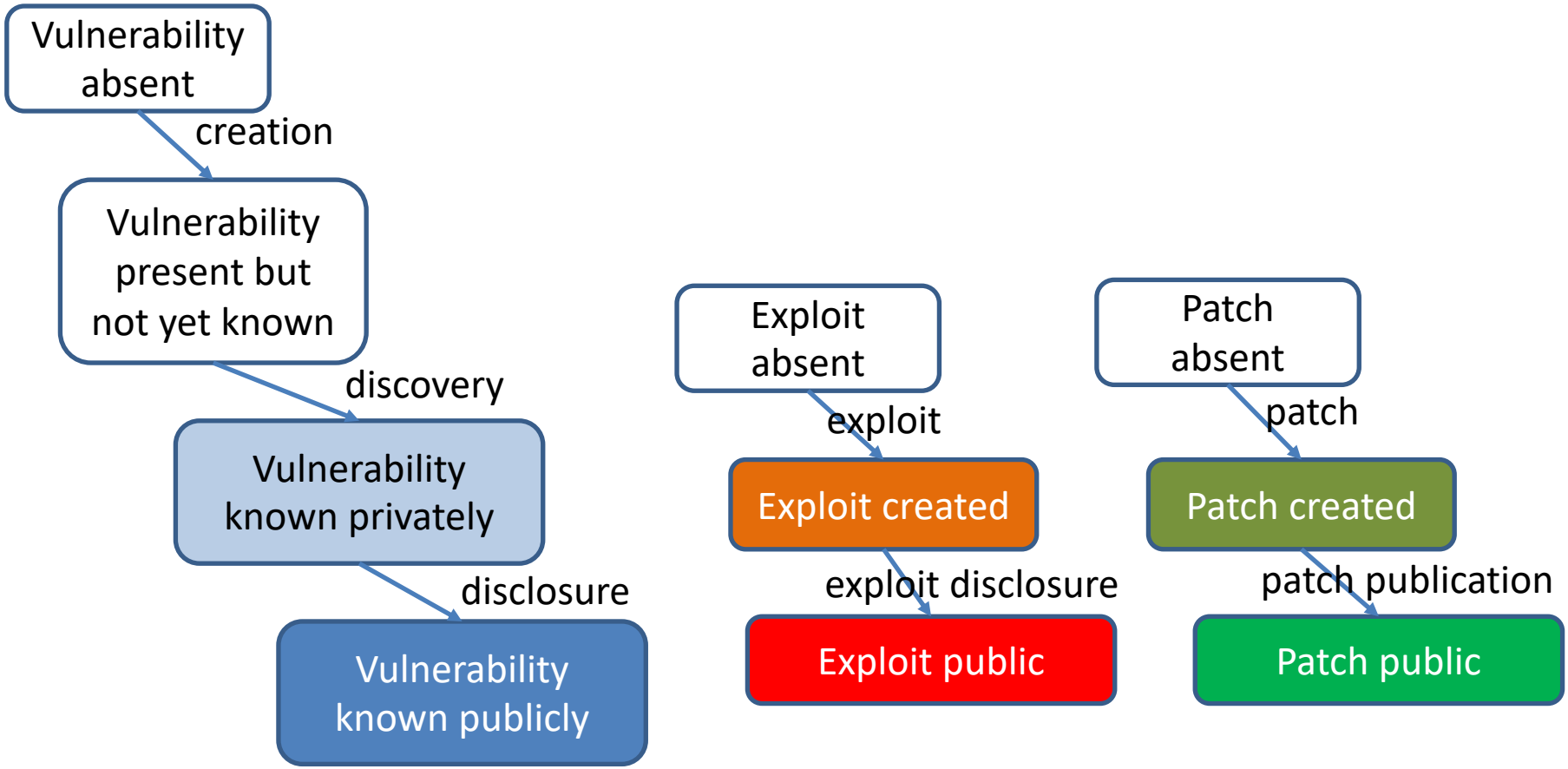
# Vulnerability Repositories

- Public
  - MITRE Common Vulnerabilities and Exposures (https://cve.mitre.org)
  - NIST National Vulnerability Database (http://nvd.nist.gov)
  - CERT vulnerability notes ( http://www.kb.cert.org/vuls https://www.cisa.gov/uscert/ncas )
  - …
- Private
  - Exodus Intelligence (https://www.exodusintel.com)
  - Zerodium (https://zerodium.com/)
  - …

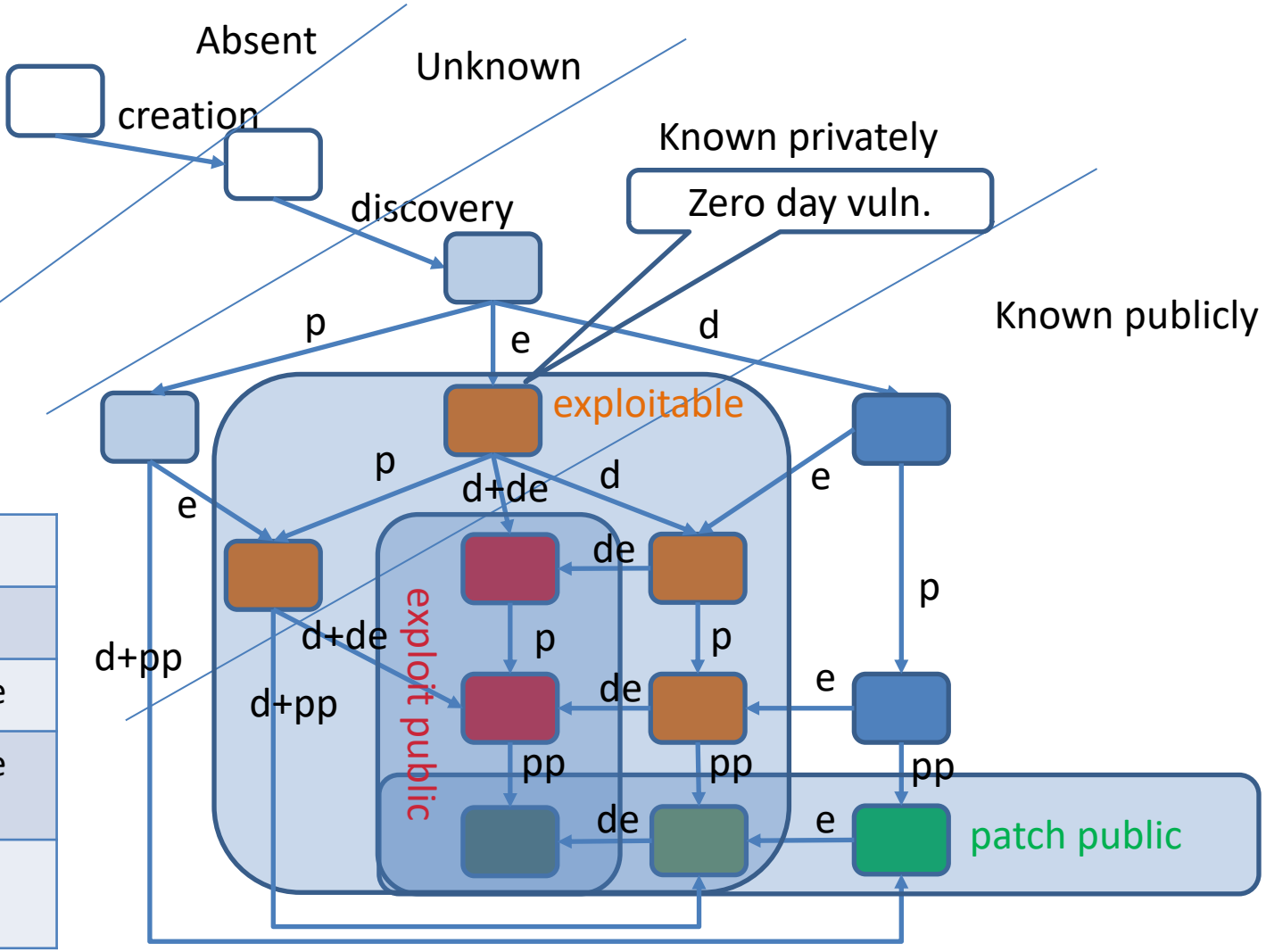
# Examples: MITRE CWE and CVE

- Common Vulnerabilities and Exposures (https://cve.mitre.org)
- Common Weakness Enumeration (https://cwe.mitre.org)

# Vulnerability Life Cycle



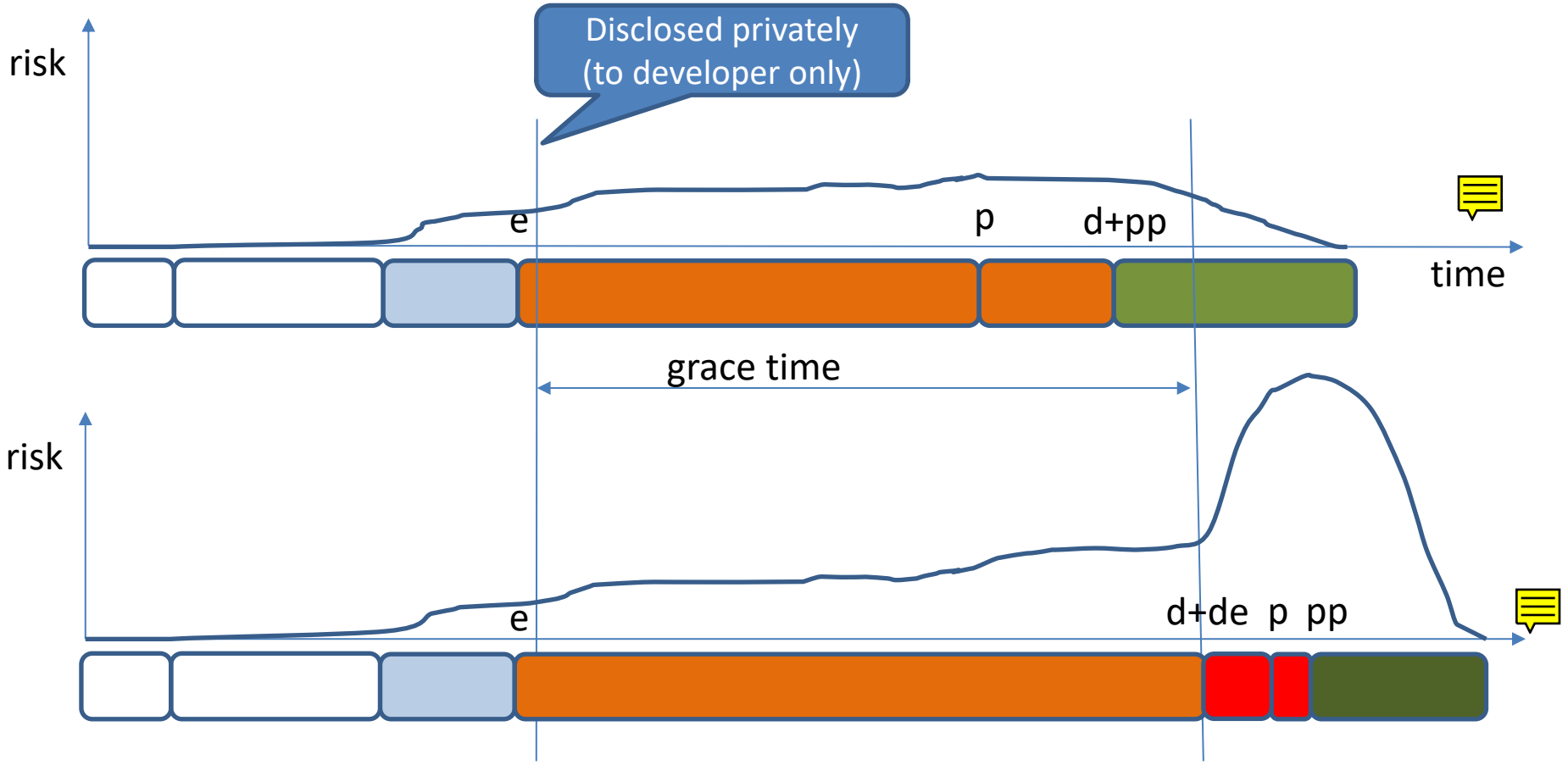
# Vulnerability Life Cycle



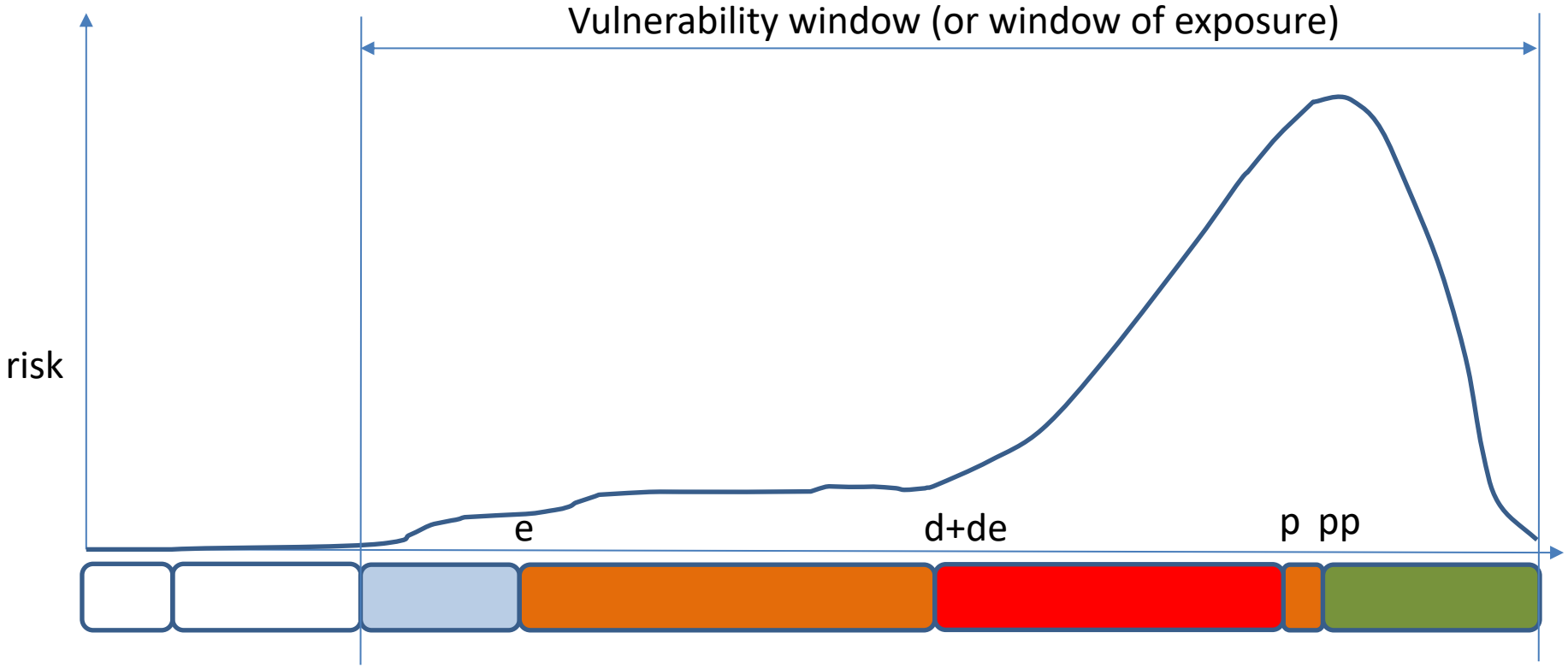

Legenda

| | |
|---|---|
| p | patch |
| e | exploit |
| d | disclose |
| de | disclose exploit |
| pp | publish patch |

# Timing Example: Responsible Disclosure



# Timing Example: Full Disclosure



# Security Assessment

- In order to perform a security assessment of a system it is necessary to
  - Look for its vulnerabilities and their related exploits
    - Like the task of an attacker, but
    - may use more information than available to an attacker
- A mix of different techniques can be used
  - **Vulnerability Assessment**, **Penetration Testing**
  - **Code Analysis**, **Formal Verification**, **Auditing**
- Let's classify and define them

# Types of Systems that can be the target of Security Assessment

- Any computer-based system
  - Hardware
  - Software
  - System
    - distributed systems
      - Network infrastructure
      - Distributed application
    - cyber-physical systems

# Different Objectives/Perspectives

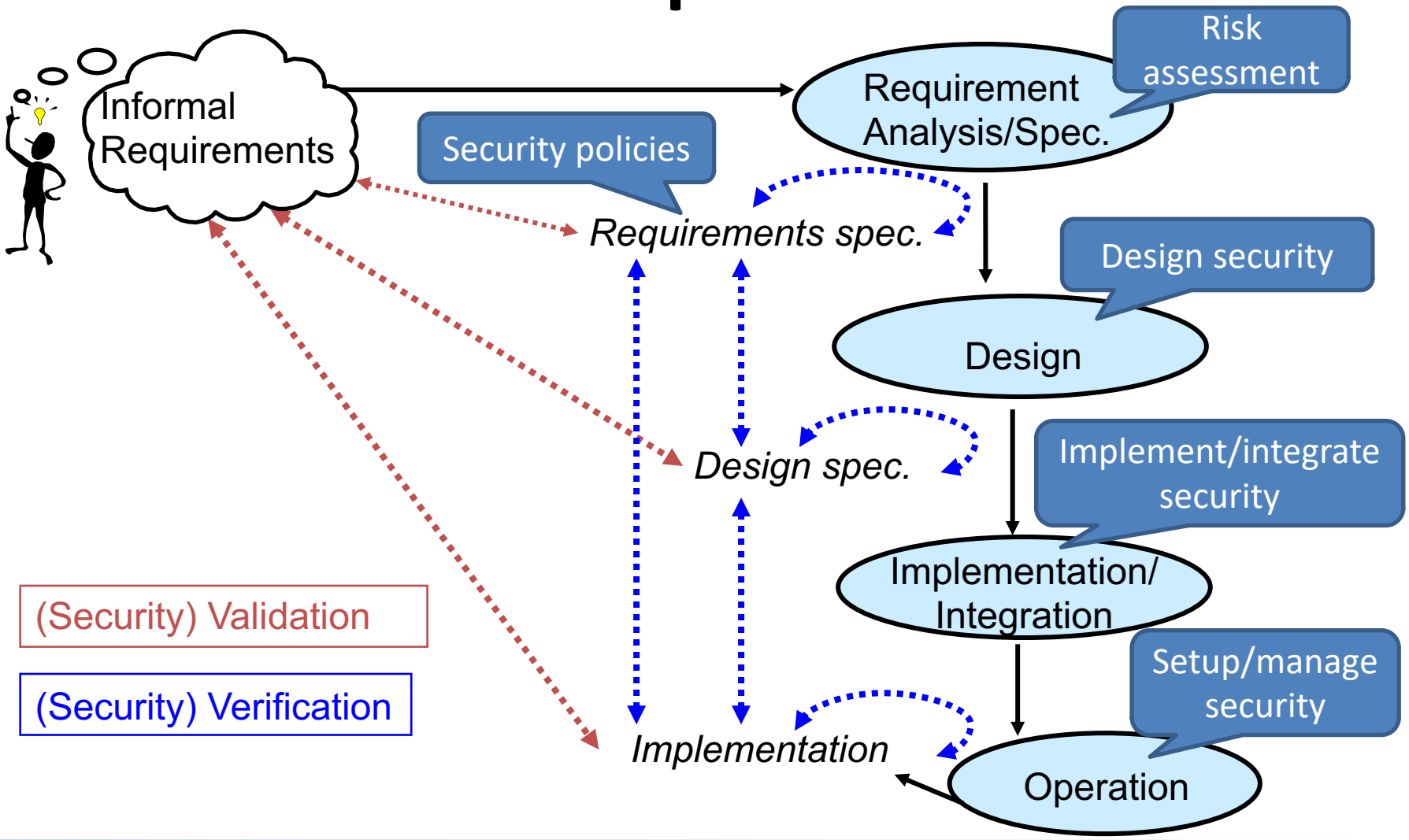
- A software vendor wants to assess the security of the developed software product
- A system administrator wants to assess the security of the administered system
- A software vendor or administrator wants to assess the security of a software product developed by a third party
- A software vendor or system administrator wants to provide a security certification of the developed software product or administered system

Assessment during development

Assessment done by user

Certification

# (Security) Assessment in System and Software Development Processes



# Cost of Fixing Defects

- According to a study of NIST:

| Phase | Cost to fix the same defect |
|---|---|
| Requirements analysis and architectural design | 1x |
| Coding/Unit testing | 5x |
| System Integration | 10x |
| Beta test programs | 15x |
| Post-release | 30x |

# (Security) Certification

- A formal attestation of some (security) properties of a system or of (security) capabilities
  - Produced by an accredited third independent party
  - Includes some form of evidence
  - Is performed according to some recognized criteria (certification standard)
    - Example: Common Criteria (CC) (ISO/IEC 15408)
    - Example: System Security Engineering Capability Maturity Model (SSE-CMM) (ISO/IEC 21827)

# Static versus Dynamic Analysis

- Static Analysis
  - Analyze system (documentation/code etc.)
  - No need to have a running/concrete system
  - Examples: auditing, formal verification
- Dynamic Analysis
  - Test a running instance of the system (testing)
  - Examples: penetration testing

# Discussion

- Static analysis more expensive
  - The most expensive analyses are applied only to selected critical components
- Dynamic analysis less exhaustive
  - Testing can show the presence of errors, not their absence
- Static Analysis can be done earlier in the development cycle

  => when fixing bugs is less expensive

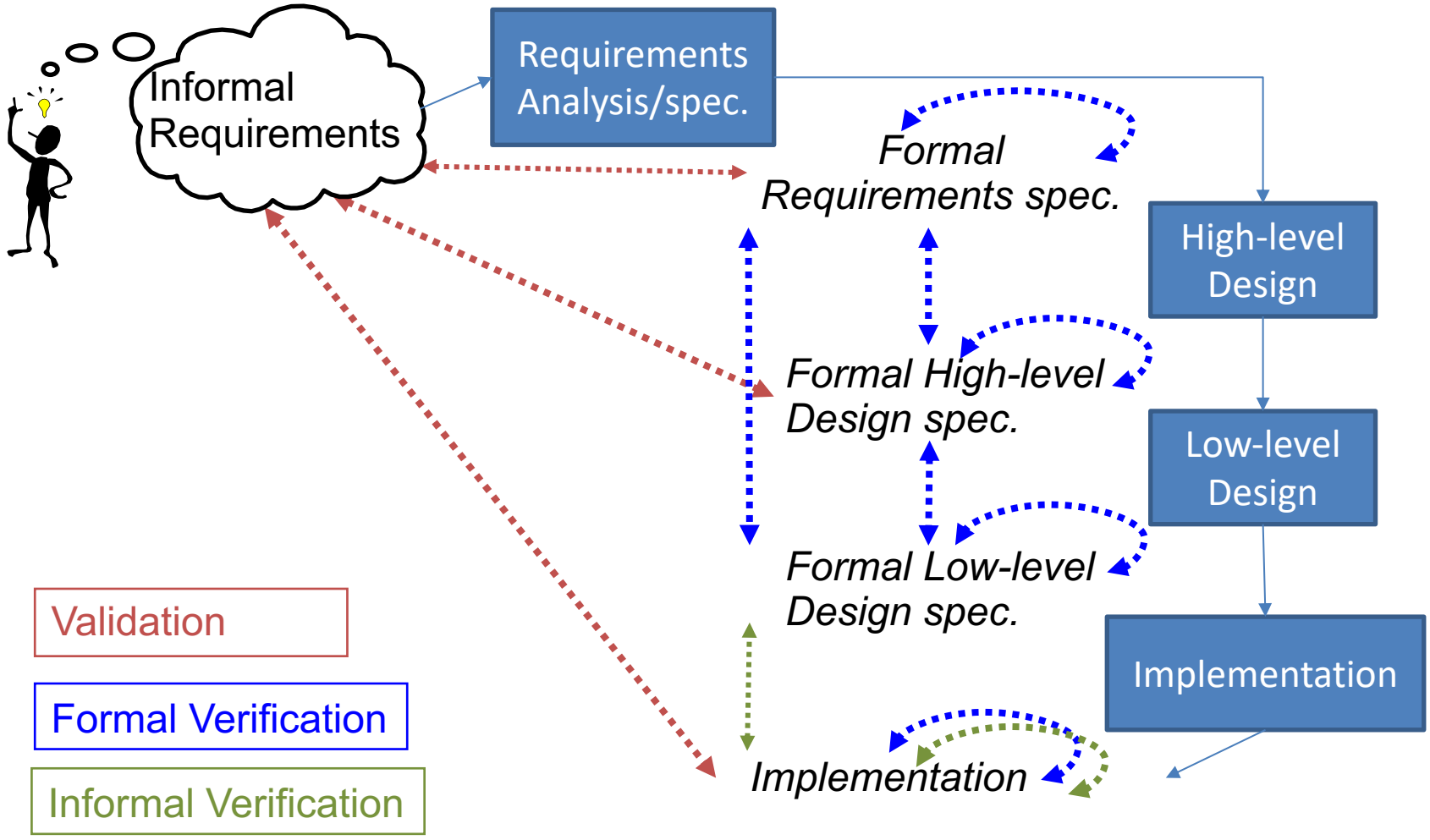
# White-Box vs Black-Box

- White-Box Techniques
  - all information about the target system is available
    - Specifications, design documentation, source code, etc.
- Black-Box Techniques
  - no information about the target system is available
    - information must be extracted from the system itself
  - Assessor behaves like a real attacker
- Grey-Box Techniques

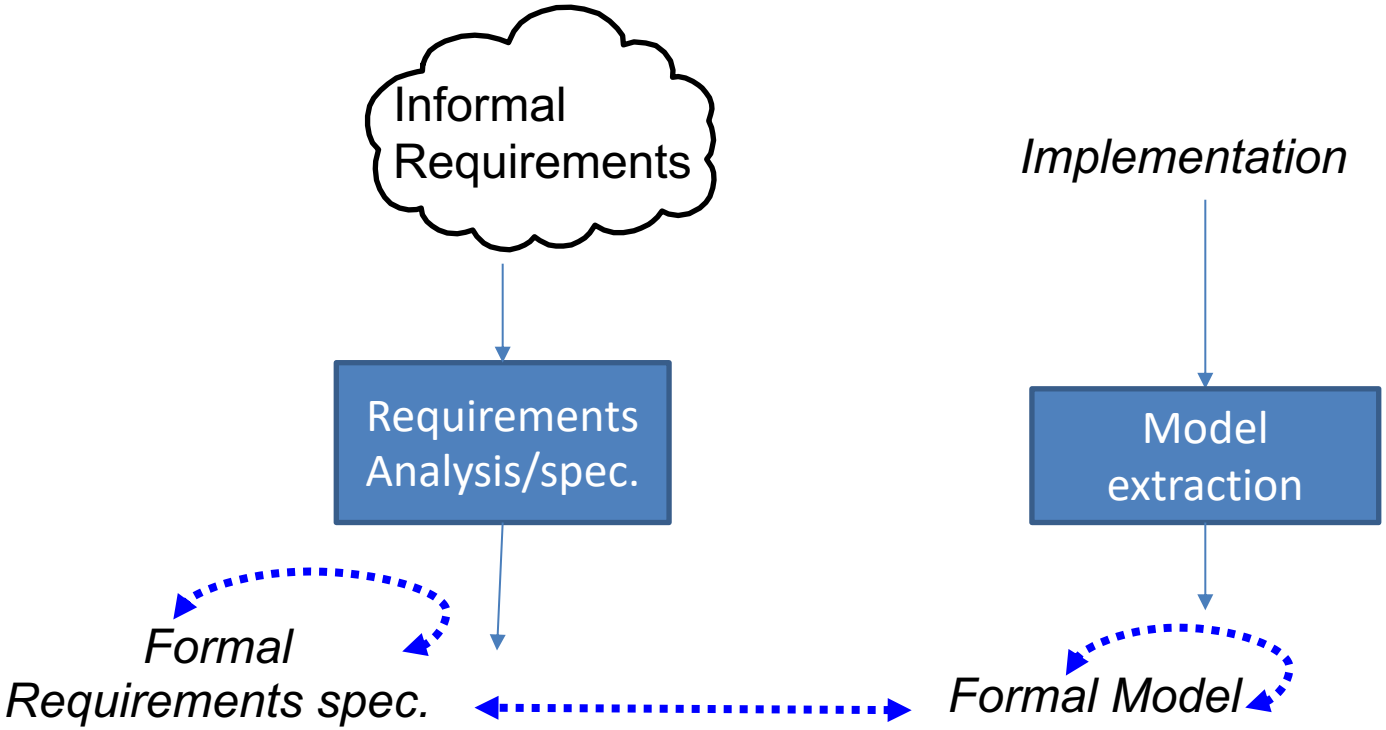
# Formal Verification

- Static Analysis of a System Formal Model
  - **Formal Model**: mathematically-based rigorous and unambiguous model
    - Examples: a state machine, a graph
- Used to provide very high **security assurance**
- Can even provide correctness proofs (about the model, not about the real system!)
- Can be applied to any type of system (hardware, software, system)

# System Development Based on Formal Methods



# Model Extraction for A-Posteriori Formal Verification



# Security Auditing/Reviews

- (Sets of) Formal Meetings aiming at evaluating security, finding vulnerabilities, proposing fixes
  - Basically manual (but can be supported by automatic tools)
  - White-box
  - Performed in various development stages (policy review, design review, code review)
  - Performed by independent auditors/reviewers (with the help of developers)

# Security Auditing/Reviews

- Typically go through several stages
  - Goal definition and planning
  - Preparation
  - Execution
  - Results reporting and sharing

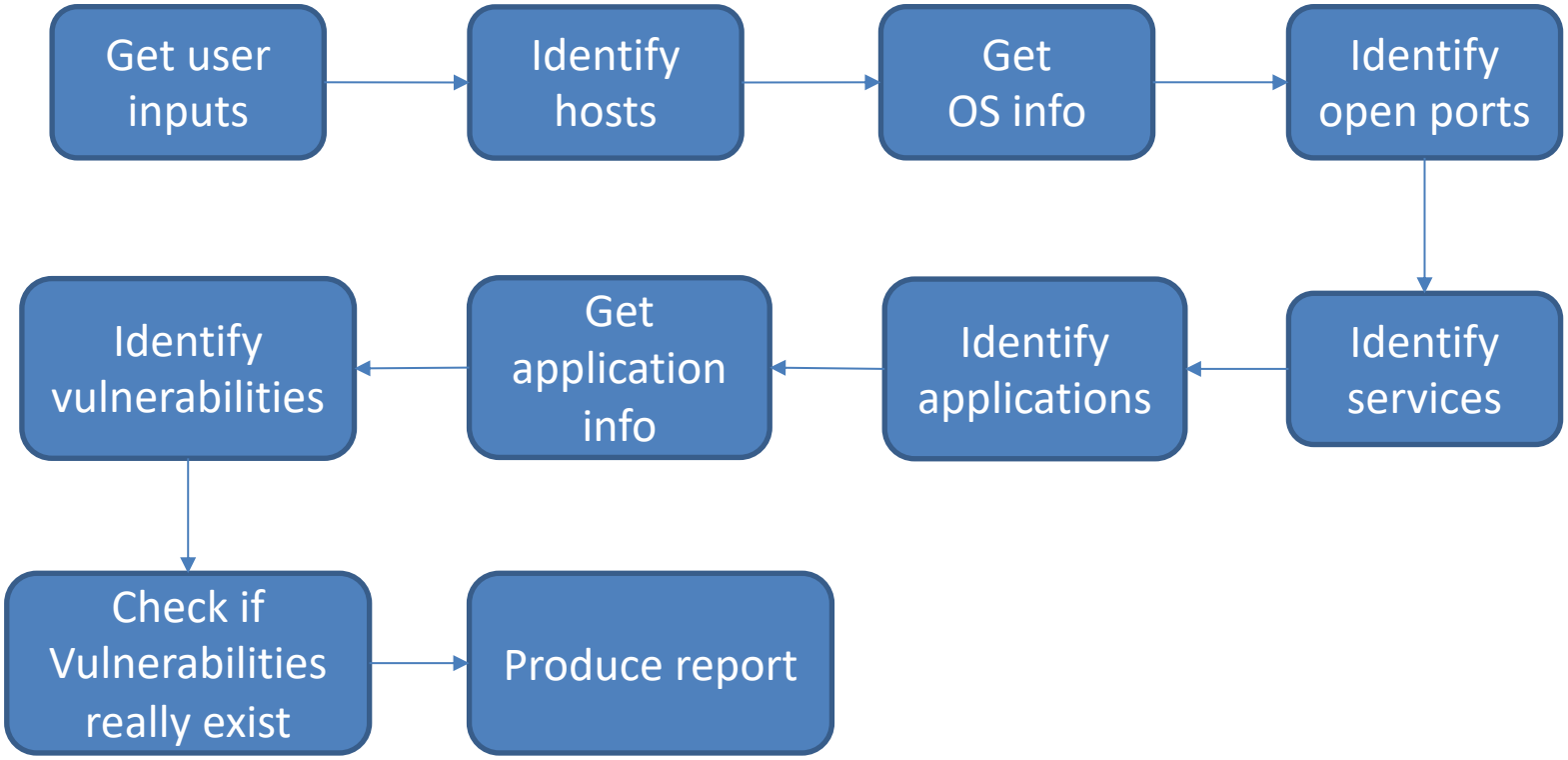
# Security Testing Techniques for Networked Systems

- Vulnerability Assessment
- Penetration Testing

# Vulnerability Assessment (VA)

- Identification (and reporting) of vulnerabilities in a system
  - Often identified with **network vulnerability assessment** (and automated scanners like Nessus)
  - May include host assessment (e.g., COPS)
  - May include static/dynamic white/black/grey-box analyses
  - May include manual auditing

# Network Vulnerability Assessment Typical Flow of Automated Tools



# White Box
# Network Vulnerability Assessment

- Exploits full knowledge of the network (insider's view/ AKA administrative approach)
  - Real topology including hosts and middleboxes (firewalls, NATs, etc)
    - May analyse parts that are usually invisible to attackers
  - User credentials for login (including admin logins)
    - May analyze host configuration files, registries, permissions, database contents, etc.

# Black Box
# Network Vulnerability Assessment

- Analyze the system as a real hacker could do (outsider's view)
  - Real topology unknown, only public IP addresses known, only DMZ initially accessible
    - Must find ways to get into the system from the outside
  - No privileges

# Discussion

- White-box and black-box approaches provide different types of information
  - White-box solutions may fail to get info about some hosts (e.g. some credentials are not available)
  - Black-box solutions are more aggressive and may disrupt the normal system behavior
  - Hybrid solutions exist (e.g. Nessus)
    - Can use both approaches based on configuration and available inputs

# False Positives

- Automated tools may report vulnerabilities that really do not exist (false positive)
  - A flaw or bug exists but it cannot be exploited
  - A patched software is not distinguished from the unpatched one (network scanners)
- Reports must be analyzed critically
- Penetration Testing is a way to discriminate real and false positives of VA

# False Negatives

- Automated tools may not report some existing vulnerabilities (false negative)
- A way to limit false negatives is to use multiple different analysis techniques and tools
- We must be aware that false negatives cannot be eliminated completely

# Penetration Testing (PT)

- Identification of vulnerabilities in a system **and** attempt to exploit them to assess what an attacker can gain from an attack (black-box)
  - First phases similar to black-box vulnerability assessment. Can use static and dynamic techniques
  - Continues with exploiting found vulnerabilities
  - Reports:
    - what vulnerabilities could be exploited?
    - what was gained by these exploits?
    - How could security be improved?

# VA vs PT

| VA | PT |
|---|---|
| Reports all found vulnerabilities | Reports vulnerabilities that can be exploited and what can be gained |
| Mostly based on automated tools | Mostly based on manual activities |
| Easy to do | Requires high expertise |
| Often done by internal personnel | Often outsourced |
| Cheap | Expensive |
| Performed frequently (whenever significant changes occur) | Performed more rarely or when more accurate analysis is needed |

# The Stages of a PT (http://www.pentest-standard.org)

- Pre-Engagement
- Information Gathering
- Threat Modeling
- Vulnerability Analysis
- Exploitation
- Post-Exploitation
- Reporting

# Techniques for Software

- Applied throughout the Development Process
  - Code Reviews and Static Code Analysis
  - Security Testing (Dynamic Security Analysis)
- May be used as well as part of system-wide assessments

# Static Code Analysis

- Static analysis of software code
- White-box => source code
  - Typically used to support code reviews
  - Examples: PVS Studio, FindSecBugs
- Black-box => binaries
  - E.g. Decompilers, Angr

# Static Code Analysis Flavors

- Type Checking
- Style Checking
- Program understanding and navigation
- Automated Formal Verification
  - Model checkers
  - Theorem provers
  - Control/Data-Flow Analyzers
- Symbolic Execution

# Software Security Testing Techniques

- Software security testers (application vulnerability scanners) use several techniques
  - Fuzzing
    - Generate randomized inputs trying to trigger unexpected software behavior (e.g., crash or error)
  - Proxies
    - Emulate man-in-the-middle attacks
    - Example: web application vulnerability scanners

# Software Security Testing Techniques

- Debuggers offer functionalities useful for vulnerability analysis and exploiting
- Security Testing tools may use static analysis techniques to find good test inputs
  - Symbolic and concolic execution

# Terminology: Analysis Techniques for Applications

- SAST (Static Application Security Testing)
  - White-box static analysis of source code
  - Examples: PVS Studio, Coverity, FindSecBugs
- DAST (Dynamic Application Security Testing)
  - Black-box dynamic analysis (vulnerability scan)
  - Examples: OWASP ZAP, Acunetix
- IAST (Interactive Application Security Testing)
  - Examples: Acusensor, Contrast Assess, Glass Box Appscan

# SAST vs DAST

| SAST | DAST |
|---|---|
| Find more vulnerabilities | Find less vulnerabilities |
| More false positives | Less false positives |
| Used in all development stages | Used in the last development stages |
| Can provide info on the cause of vulnerability (code line) | Cannot provide info on the cause of vulnerability (black-box) |
| Coverage of libraries is an issue | |
| Each tool applies to only some languages/frameworks | Independent of language/technology used to develop application |
| May be time-consuming | |

# IAST

- Recent concept that tries to combine static and dynamic analyses
  - Motivation: complexity of applications (especially web apps), made of many third-party components
  - Main ideas
    - Agent that runs inside a running application (code instrumentation)
    - Agent has access to code and all execution details, but only the code that is triggered is analyzed
    - Agent can run continuously (even during operation)
    - Reports are live (continuously generated/updated)

# IAST Pros and Cons

- Pros
  - Find many vulnerabilities (like SAST) but with few false positives
  - Very fast and scalable (good for dev-sec-ops)
- Cons
  - Not yet available for any language/framework
  - Not yet widely known and adopted

# Tool Benchmarking

- Different initiatives for benchmarking
  - OWASP Benchmark Project
    - https://owasp.org/www-project-benchmark/
  - The Web Application Vulnerability Scanner Evaluation Project (WAVSEP)
    - https://sectooladdict.blogspot.com/