Software Security Engineering Lecture 3

Nancy R. Mead, SEI
nrm@sei.cmu.edu

Outline

I. An Assurance Ecosystem (carried over from Lecture 2)

II. Requirements Engineering

III. Introduction to SQUARE

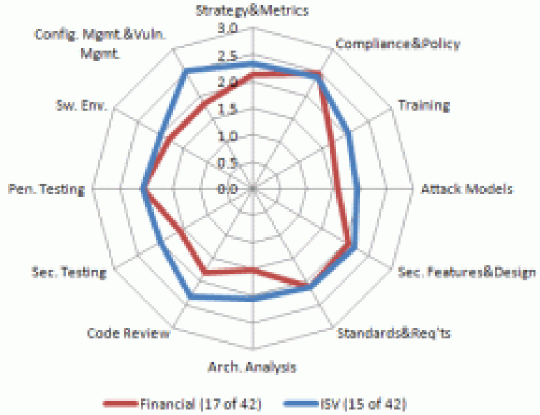
IV. Questions

An Assurance Ecosystem

Developed by Dan Reddy EMC-2

One view as to how the pieces fit

BSIMM



Shows data congruence of security activities found in companies that were analyzed

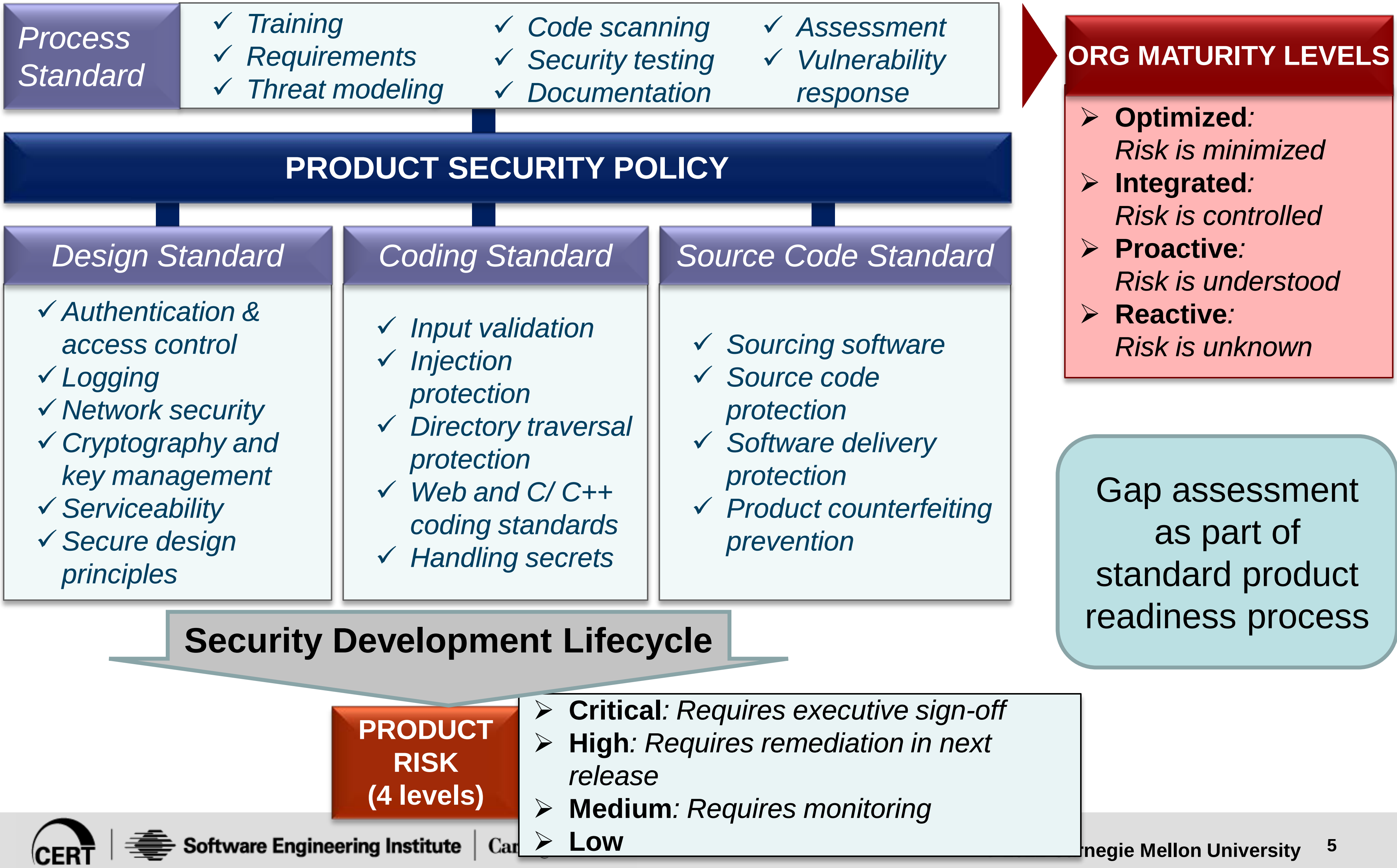
THE Open GROUP
Making standards work®

- Standard that outlines best practices of ICT Providers to mitigate vs *tainted* & *counterfeit* products.
- Method to accredit Trusted Technology Providers.

SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

- Building secure products
- Prescriptive.
- How should I do it?
- Where should I start?

EMC-wide Standard with focus on Risk and Organization Maturity



Process Standard

- ✓ *Training*
- ✓ *Requirements*
- ✓ *Threat modeling*
- ✓ *Code scanning*
- ✓ *Security testing*
- ✓ *Documentation*
- ✓ *Assessment*
- ✓ *Vulnerability response*

PRODUCT SECURITY POLICY

Design Standard

- ✓ *Authentication & access control*
- ✓ *Logging*
- ✓ *Network security*
- ✓ *Cryptography and key management*
- ✓ *Serviceability*
- ✓ *Secure design principles*

Coding Standard

- ✓ *Input validation*
- ✓ *Injection protection*
- ✓ *Directory traversal protection*
- ✓ *Web and C/ C++ coding standards*
- ✓ *Handling secrets*

Source Code Standard

- ✓ *Sourcing software*
- ✓ *Source code protection*
- ✓ *Software delivery protection*
- ✓ *Product counterfeiting prevention*

Security Development Lifecycle

PRODUCT RISK (4 levels)

- ➢ **Critical**: *Requires executive sign-off*
- ➢ **High**: *Requires remediation in next release*
- ➢ **Medium**: *Requires monitoring*
- ➢ **Low**

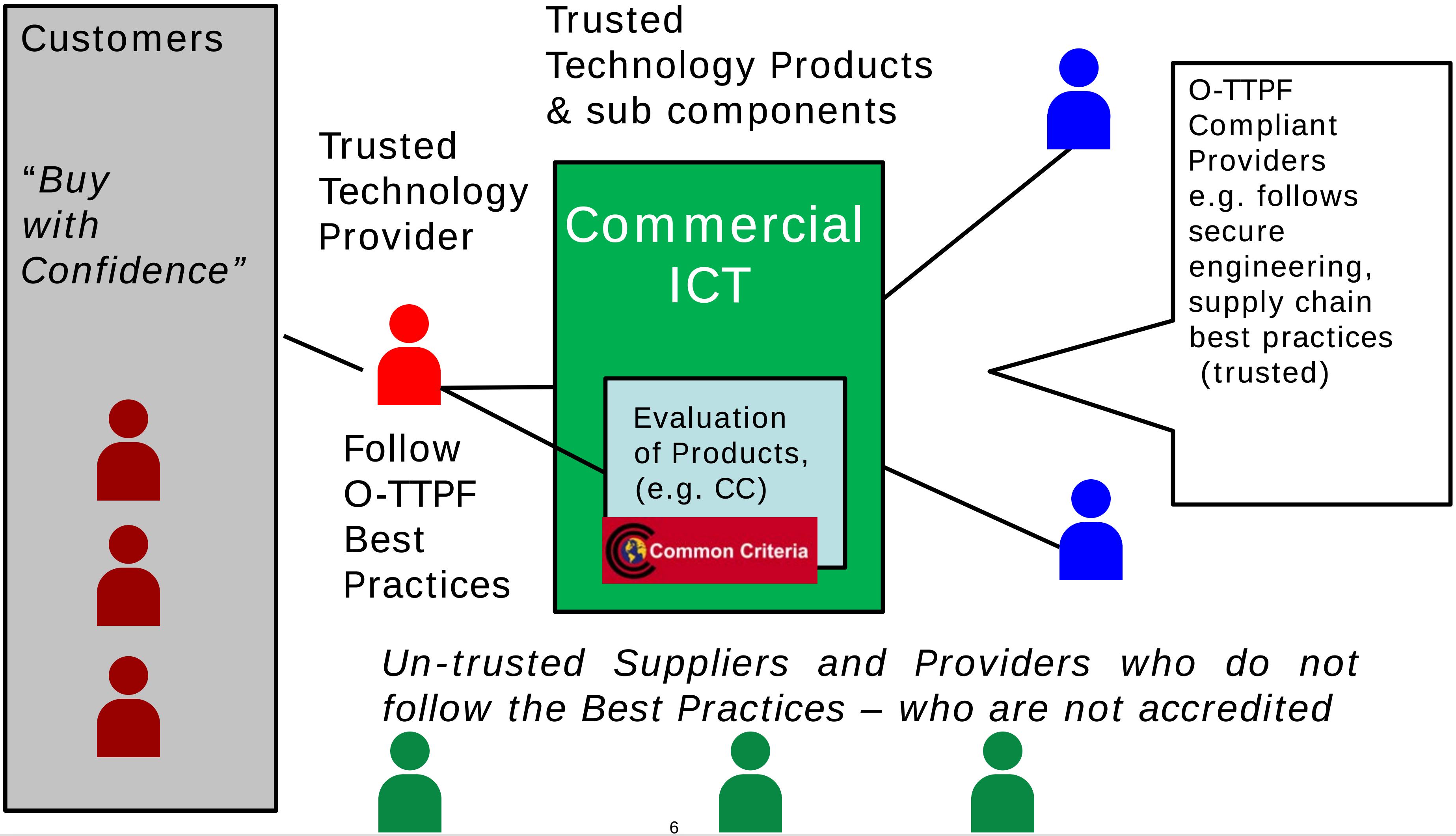
ORG MATURITY LEVELS

- ➢ **Optimized**: *Risk is minimized*
- ➢ **Integrated**: *Risk is controlled*
- ➢ **Proactive**: *Risk is understood*
- ➢ **Reactive**: *Risk is unknown*

Gap assessment as part of standard product readiness process

Customers Buy with More Confidence:

Providers & Suppliers Can Extend Supply Chain Integrity



Customers

“Buy with Confidence”

Trusted Technology Provider

Trusted Technology Products & sub components

Commercial ICT

Evaluation of Products, (e.g. CC)

Common Criteria

Follow O-TTPF Best Practices

O-TTPF Compliant Providers e.g. follows secure engineering, supply chain best practices (trusted)

Un-trusted Suppliers and Providers who do not follow the Best Practices – who are not accredited

Classifying Vulnerabilities: Some Useful Resources

- CVE: Common Vulnerabilities & Exposures Database
  - http://cve.mitre.org
- CWE: Common Weakness Enumeration
  - A community-developed dictionary of software weakness types
  - http://cwe.mitre.org/
- NVD: National Vulnerability Database
  - http://nvd.nist.gov
- Bugtraq mailing list: how to exploit & fix vulnerabilities
  - http://www.securityfocus.com/archive/1

Secure Coding: Some Useful Resources

- CERT Secure Coding Initiative
  http://www.cert.org/secure-coding/
- SANS Software Security Institute
  - http://www.sans-ssi.org/
- Open Web Application Security Project (OWASP)
  - http://www.owasp.org/
- Web Application Security Consortium (WASC)
  - http://www.webappsec.org/

Requirements Engineering

Requirements Engineering Issues

- RE defects cost up to 200 times more once fielded than if caught in requirements engineering
- Reworking defects consumes >50% of project effort
- >50% of defects are introduced in requirements engineering
- **Takeaway: Errors during requirements engineering are costly!**

Requirements Engineering Issues – Example

Cost of Fixing Vulnerabilities Later

Stage	**Critical Bugs Identified**	**Cost of Fixing One Bug**	**Cost of Fixing All Bugs**
Requirements		$139	
Design		$455	
Coding		$977	
Testing	50	$7,136	$356,800
Maintenance	150	$14,102	$2,115,300
Total	**200**		**$2,472,100**

Cost of Fixing Vulnerabilities Early

Stage	**Critical Bugs Identified**	**Cost of Fixing One Bug**	**Cost of Fixing All Bugs**
Requirements		$139	
Design		$455	
Coding	150	$977	$146,550
Testing	50	$7,136	$356,800
Maintenance		$14,102	
Total	**200**		**$503,350**

As can be seen, identifying defects early in the life cycle reduced costs by nearly $2 million.

Microsoft Security Lifecycle Results

- **Microsoft Windows: 45% Fewer Vulnerabilities in Windows Vista**
- Windows Vista was the first Microsoft operating system to benefit from the SDL. After the first year, Windows Vista had 45% fewer vulnerabilities than Windows XP. In a comparison of security vulnerabilities, Windows Vista also fares better than competing operating systems
- **Microsoft SQL Server: 91% Fewer Vulnerabilities in SQL Server 2005**
- SQL Server serves as an excellent example for security improvements resulting from incorporating the SDL. Within the three years after release, Microsoft has issued three security bulletins for the SQL Server 2005 database engine

Reference: <http://www.microsoft.com/security/sdl/learn/measurable.aspx>

Requirements Problems

- Requirements identification may not include relevant stakeholders
- Requirements analysis may or may not be performed
- Requirements specification are typically haphazard

Effects of Requirements Problems

Bad requirements cause projects to:

- exceed schedule
- exceed budget
- have significantly reduced scope
- deliver poor-quality applications
- deliver products that are not significantly used
- be cancelled

Security Requirements

- Address security in a particular application
- Are often ignored in the requirements elicitation process
- Incur high costs when incorporated later
- Must be addressed early

Security Requirements Methods

- **SQUARE**
- CLASP
- Core Security Requirements Artifacts
- SREP
- Security Patterns
- TROPOS
- Others

Security Requirements Methods

SQUARE

- Security Quality Requirements Engineering
- Nine-step process
- SQUARE-Lite
- SQUARE for Privacy
- SQUARE for Acquisition
- Can be used with existing requirements engineering process

SQUARE Methodology

What is it? Who is involved?

SQUARE

- Developed by the Networked Systems Survivability program at the SEI, Carnegie Mellon University.
- Stepwise methodology for eliciting, categorizing, and prioritizing security requirements for information technology systems and applications
- Security requirements are quality attributes.

SQUARE

Who is involved?

- stakeholders of the project
- requirement engineers with security expertise

In SQUARE, security requirements are:

- treated at the same time as the system's functional requirements, AND
- specified in the early stages of the SDLC
- specified in similar ways as software requirements engineering and practices
- determined through a process of nine discrete steps

SQUARE Steps

The Nine Steps

SQUARE Steps

1. Agree on definitions.
2. Identify assets and security goals.
3. Develop artifacts to support security requirements definition.
4. Assess risks.
5. Select elicitation technique(s).
6. Elicit security requirements.
7. Categorize requirements.
8. Prioritize requirements.
9. Inspect requirements.

Step 1

1	2	3	4	5	6	7	8	9
Def.	*Goals*	*Artifacts*	*Risk*	*Technique*	*Elicit*	*Categorize*	*Prioritize*	*Inspect*

Agree on Definitions

- Requirements engineers and stakeholders agree on a set of definitions.
- Process is carried out through interviews.
- Exit criteria: documented set of definitions
- Examples: non-repudiation, denial-of-service (DoS), intrusion, malware

Step 2

1	**2**	3	4	5	6	7	8	9
Def.	**Goals**	*Artifacts*	*Risk*	*Technique*	*Elicit*	*Categorize*	*Prioritize*	*Inspect*

Identify Assets and Security Goals

- Identify assets to be protected in the system.
- Goals are required to identify the priority and relevance of security requirements.
- Security goals must support the business goal.
- Goals are reviewed, prioritized, and documented.
- Exit criteria: one business goal, several security goals

Step 3

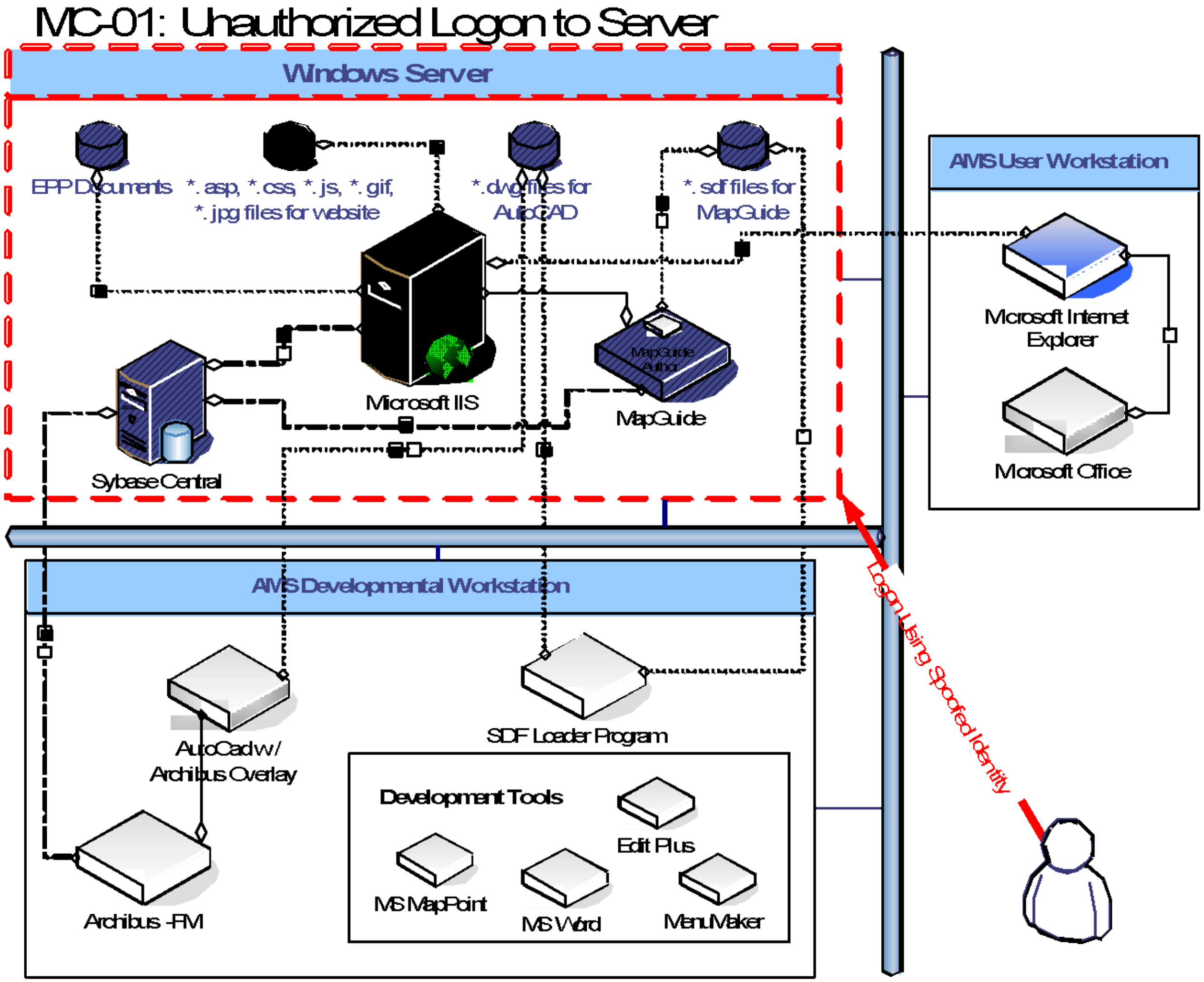
1	**2**	**3**	4	5	6	7	8	9
Def.	**Goals**	**Artifacts**	*Risk*	*Technique*	*Elicit*	*Categorize*	*Prioritize*	*Inspect*

Develop Artifacts

- Collect or create artifacts that will facilitate generation of security requirements.
- Jointly verify their accuracy and completeness.
- Examples: system architecture diagrams, use/misuse case scenarios/diagrams, attack trees, templates and forms

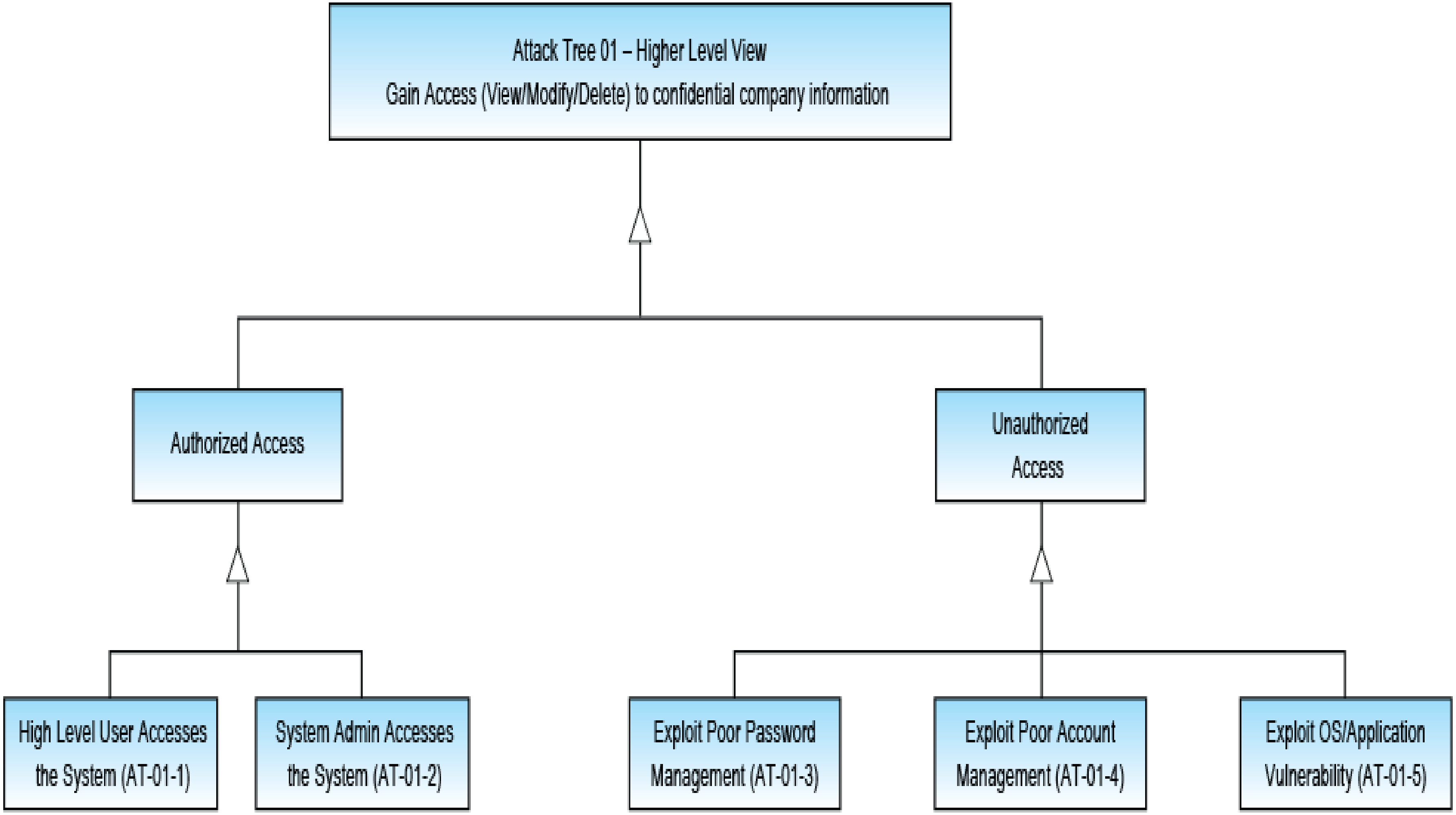
Examples of Artifacts

Misuse Case Diagram



Examples of Artifacts

Attack Tree



Attack Tree 01 – Higher Level View

Gain Access (View/Modify/Delete) to confidential company information

Authorized Access

Unauthorized Access

High Level User Accesses the System (AT-01-1)

System Admin Accesses the System (AT-01-2)

Exploit Poor Password Management (AT-01-3)

Exploit Poor Account Management (AT-01-4)

Exploit OS/Application Vulnerability (AT-01-5)

Step 4

1	**2**	**3**	**4**	5	6	7	8	9
Def.	**Goals**	**Artifacts**	**Risk**	*Technique*	*Elicit*	*Categorize*	*Prioritize*	*Inspect*

Perform Risk Assessment

- Identify threats to the system and its vulnerabilities.
- Calculate likelihood of their occurrence. Classify them. This will also help in prioritizing requirements later.
- Risk expert might be required.
- Exit criteria: documentation of all threats, their likelihood and classifications

Step 5

1	**2**	**3**	**4**	**5**	6	7	8	9
Def.	**Goals**	**Artifacts**	**Risk**	**Technique**	*Elicit*	*Categorize*	*Prioritize*	*Inspect*

Select Elicitation Technique

- Select appropriate technique for the number and expertise of stakeholders, requirements engineers, and size and scope of the project.
- Techniques: structured/unstructured interviews, **accelerated requirements method** (ARM), soft systems methodology, issue based information systems (IBIS), Quality Function Deployment

Step 6

1	**2**	**3**	**4**	**5**	**6**	7	8	9
Def.	**Goals**	**Artifacts**	**Risk**	**Technique**	**Elicit**	*Categorize*	*Prioritize*	*Inspect*

Elicit Security Requirements

(Heart of SQUARE)

- Execute the elicitation technique.
- Avoid non-verifiable, vague, ambiguous requirements.
- Concentrate on what, not how.
  Avoid implementations and architectural constraints.
- Exit criteria: initial document with requirements

Step 7

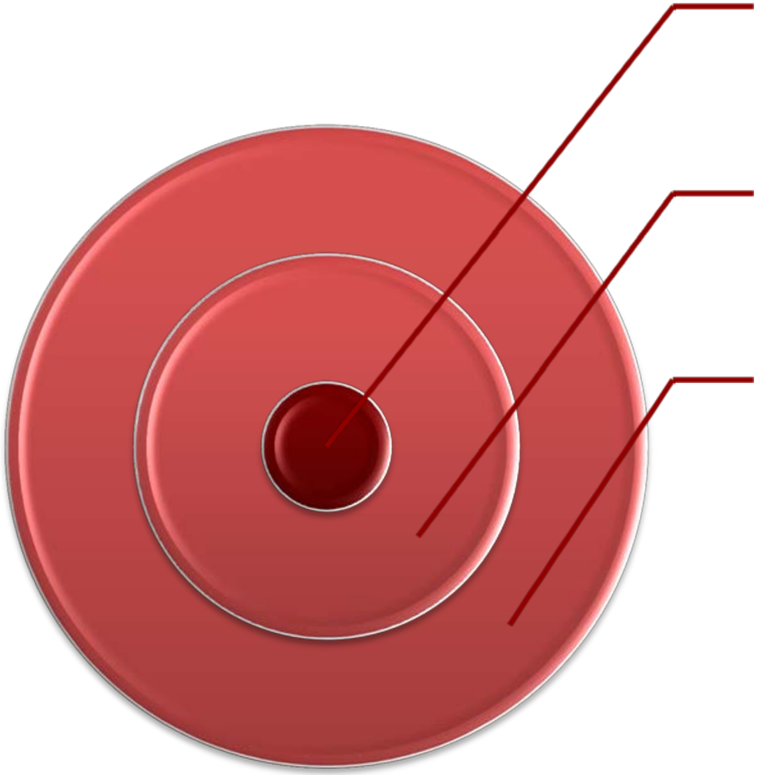
1	**2**	**3**	**4**	**5**	**6**	**7**	8	9
Def.	**Goals**	**Artifacts**	**Risk**	**Technique**	**Elicit**	**Categorize**	*Prioritize*	*Inspect*

Categorize Requirements

- Classify requirements into essential, non-essential, system, software, or architectural constraints.
- Sample table:

	System level	Software level	Architectural constraint
Reqt. 1			
Reqt. 2			

Step 7- Categorize Requirements Examples



Software Level: Users cannot exceed their access privileges.

System Level: The system is required to have strong authentication measures in place at all system gateways/entrance points.

Architectural Constraints: The system should be able to support the capabilities of a distributed network.

Step 8

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	9
Def.	**Goals**	**Artifacts**	**Risk**	**Technique**	**Elicit**	**Categorize**	**Prioritize**	*Inspect*

Prioritize Requirements

- Use risk assessment and categorization results to prioritize requirements.
- Prioritization techniques: Triage, Win-Win, Analytical Hierarchy Process
- Requirements engineering team should produce a cost-benefit analysis to aid stakeholders.

Step 9

1	2	3	4	5	6	7	8	9
Def.	**Goals**	**Artifacts**	**Risk**	**Technique**	**Elicit**	**Categorize**	**Prioritize**	**Inspect**

Requirements Inspection

- Inspection aids in creating accurate and verifiable security requirements.
- Look for ambiguities, inconsistencies, mistaken assumptions.
- Fagan inspections / peer reviews
- Exit criteria: all requirements verified and documented

Approach

The SQUARE process

- takes about three months calendar time to complete
- has been implemented in several case studies

SQUARE-Lite

- Agree on definitions.
- Identify assets and security goals.
- Perform risk assessment
- Elicit security requirements.
- Prioritize requirements.

SQUARE-Lite has been implemented in one case study.

Conclusion

Summary

- SQUARE – Security Quality Requirements Engineering

- Nine steps:
  - (1) agree on definitions
  - (2) identify assets and security goals
  - (3) develop artifacts
  - (4) assess risks
  - (5) select elicitation technique(s)
  - (6) elicit security requirements
  - (7) categorize requirements
  - (8) prioritize requirements
  - (9) inspect requirements

- SQUARE-Lite, P-SQUARE, A-SQUARE

Additional Resources

- R. Anderson – Home Page <http://act-r.psy.cmu.edu/people/ja/>
- Dr. Haralambos Mouratidis – Brief Biography <http://www.uel.ac.uk/cite/staff/haralambosmouratidis.htm#Biography>
- Mary Shaw – Research Activities <http://spoke.compose.cs.cmu.edu/shaweb/r/research.htm>

Additional Resources

- BSI content on requirements engineering
  <https://buildsecurityin.us-cert.gov/>
- SQUARE Technical Report – SEI web site
  <www.sei.cmu.edu/pub/documents/05.reports/pdf/05tr009.pdf>
- SQUARE Case Study Reports – SEI web site
- “Integrating Security and Software Engineering”
  IDEA Group Publishing
  <www.idea-group.com>
- SQUARE-Lite
  <http://www.cert.org/sse/square.html>

SQUARE Demo Videos

<http://www.cert.org/sse/square/square-tool.html>

Questions?

Looking Ahead: Lecture #4

I. Recap of SQUARE

II. SQUARE for Acquisition

Reading Assignment

- Chapter 3 in textbook
- Beckers paper on requirements engineering process: http://link.springer.com/chapter/10.1007/978-3-642-28166-2_2
- Khan/Zulkernine paper on selecting requirements engineering processes: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=5254051
- BSI content on requirements engineering <https://buildsecurityin.us-cert.gov/>
- SQUARE Technical Report – SEI web site <www.sei.cmu.edu/pub/documents/05.reports/pdf/05tr009.pdf>

Homework Assignment # 2

1) (25%) You are working on a project where you can select a security requirements engineering process. First you want to decide on some criteria for selection. What criteria do you pick (refer to the Khan/Zulkernine and Beckers papers for a start)?

2) (50%) Using those criteria, which existing process is the best fit (you can use the list of processes on slide 16 as a start)?

3) (25%) Does the selected process need to be modified for your project?

- Turn this in on Blackboard BEFORE the next class.