

Specifying Security Goals of Component based Systems: An End-User Perspective

Khaled Khan

**Qatar University
Qatar**

Jun Han

**Swinburne University of
Technology, Australia**

Overview

- **Context**
- **Security Perspectives**
- **Our Approach**
- **Security characterization**
- **An example**
- **Conclusion**

Software Component, Composition and Security

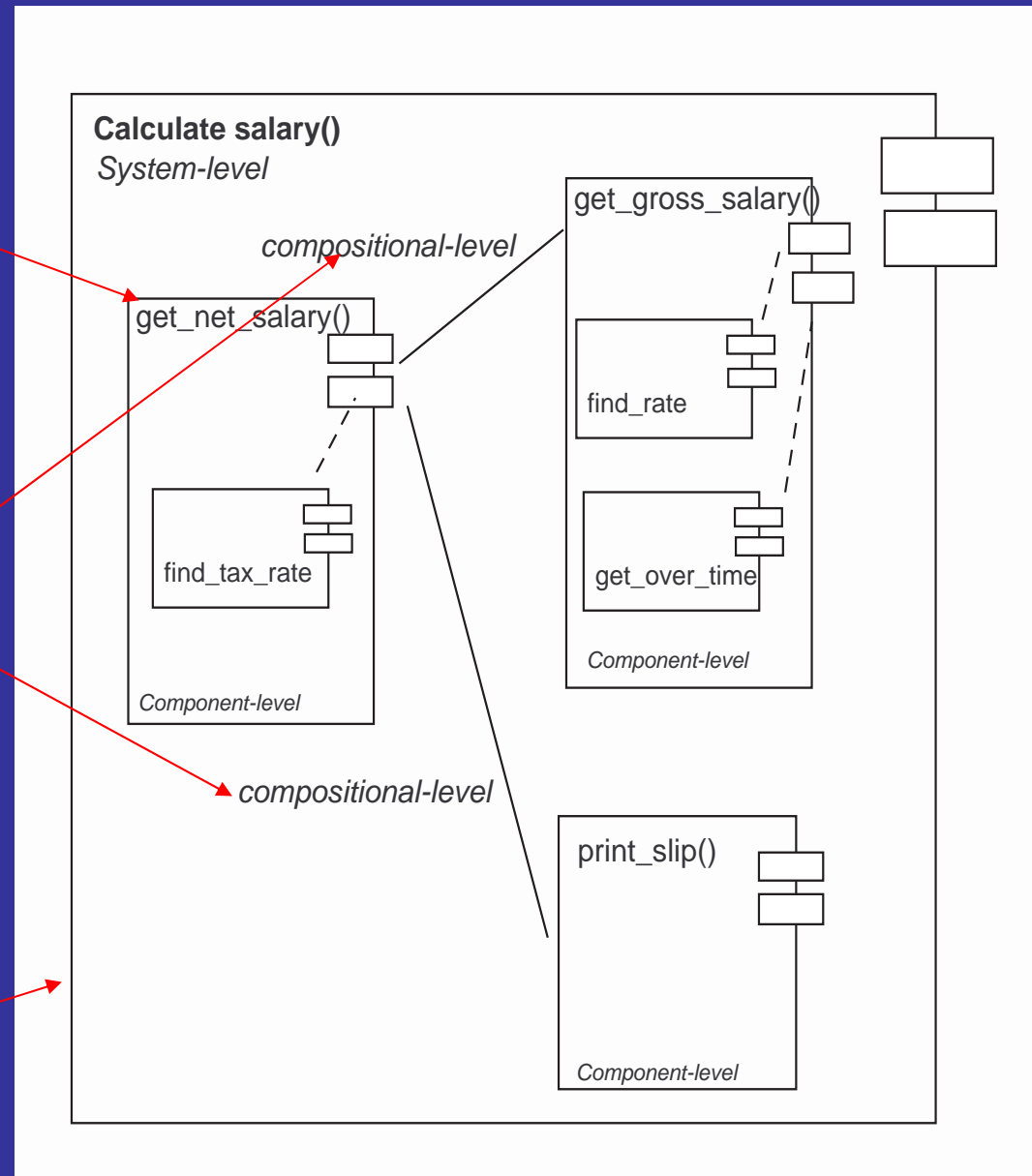
- An increasing interest in deploying software applications as services over the open communication channels such as Internet
- A software offering a service exists independently - developed, managed by third party service provider
- These services are aimed for direct integration with each other or with any application system dynamically at run-time to provide cross-application transactions
- A service may be **secure** in one application system, but the same service may not be secure in a different application due to different security requirements
- The term '**secure**' is over-used and somehow misleading because it does not state the specific type of security achieved
- Security of component based system can be seen from **three different perspectives** during three different stages of the component based systems hierarchy

Hierarchy of Composite Systems

- Component-level security

- Compositional-level security

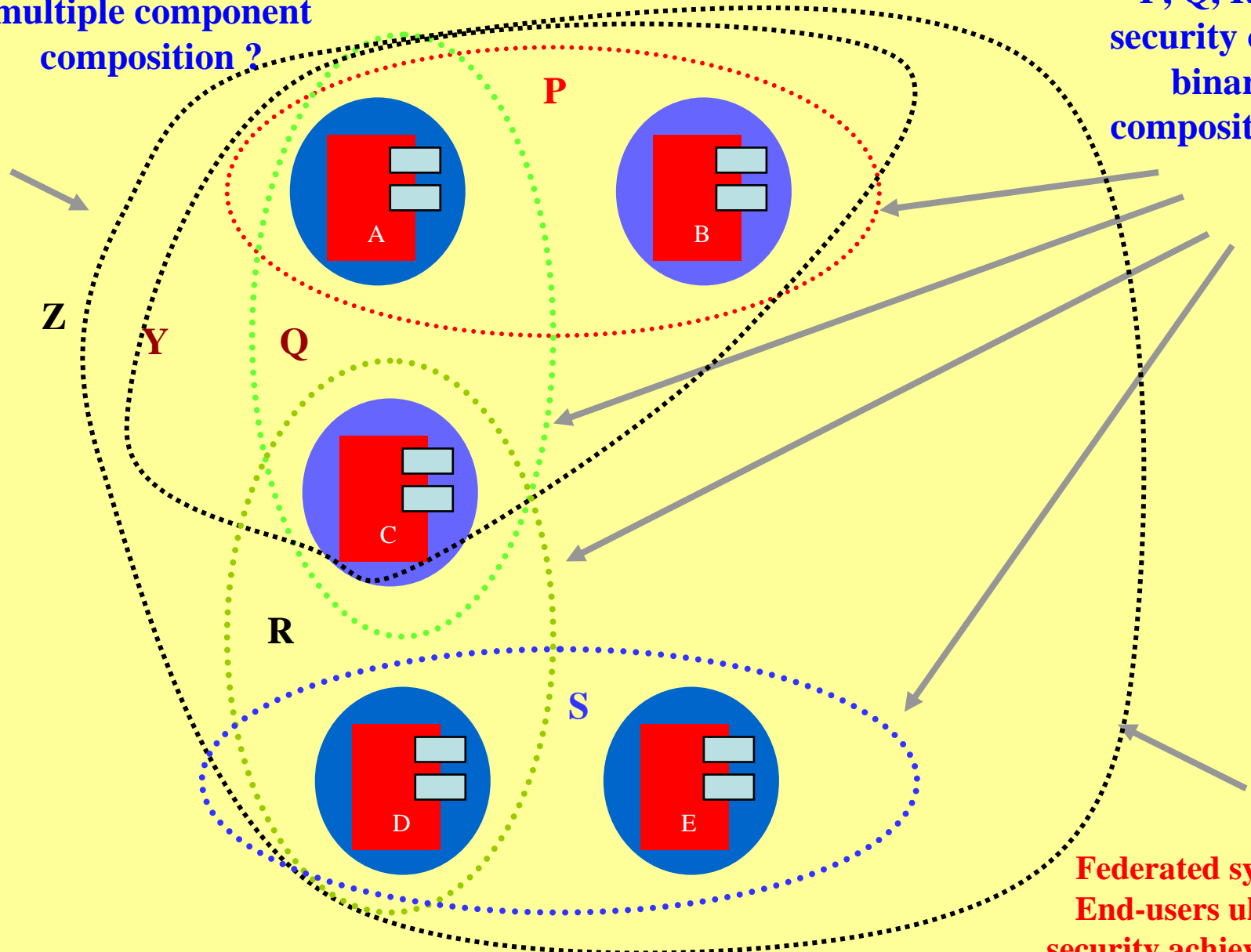
- End-users level security



Security Perspectives of Composite Systems

Y and Z: security of the multiple component composition ?

P, Q, R, S: security of the binary compositions?



Federated systems.
End-users ultimate
security achievements?

Three Perspectives of Systems Security

1. Computer security expert's perspective

- Focuses technical details of the component security such as encryption
- Identifies the threats of the component, define the security policies and functions (component design time)

2. Software engineer's perspective

- Interested in the compositional impact and conformity of the security properties (Composition time)

3. End-user's perspective

- Specific security objectives actually achieved at the system-level (Operational time)
- The latter perspective depends on the first two perspectives.
- End-users level security achievements can only be derived from the actual security enforced at the composition-level.
- The major focus of this paper is on the third perspective.

An Example of the Problem: A component based composite system

- A composite system processes online credit card payment
- It claims that the system is `secure`
- Does this claim spell out for the users whether
 - The card number will be kept confidential ?
 - The amount will be confidential ?
 - The amount will not be tampered ?
 - The payment beneficiary cannot deny later that the payment was received ?
 - The card holder also cannot deny the payment at a later stage? or
 - Some other security issues ?

Research Challenges Addressed

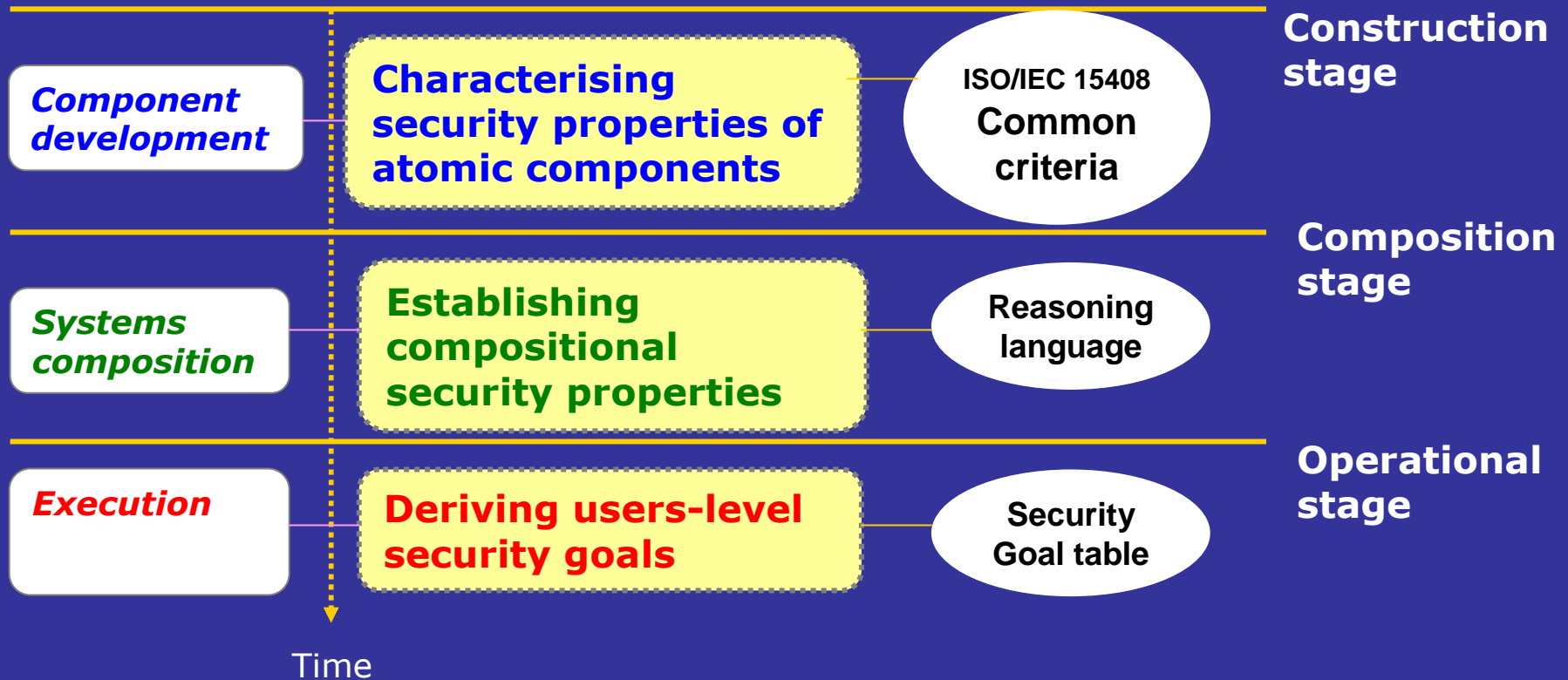
How to **characterize the security goals** of component based composite systems for the end-users level understanding

How the end-users **know the specific security objectives achieved** as (opposed to 'secure') of a composite system at runtime ?

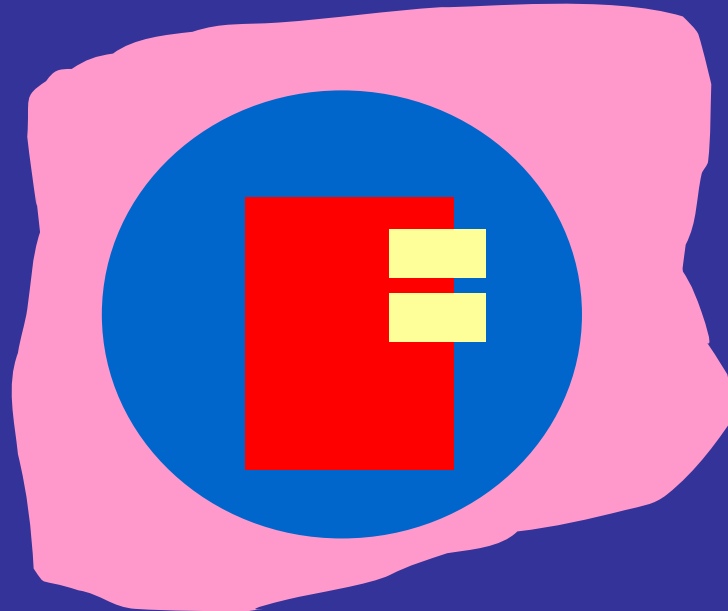
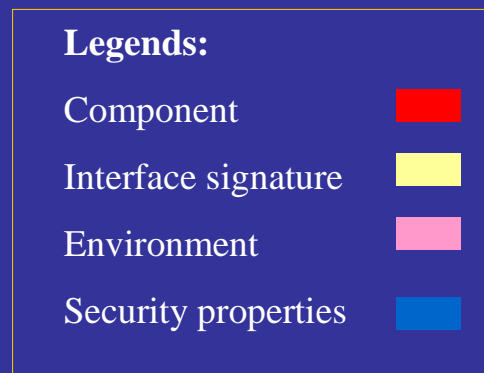
Approach

- Characterize the security properties of at the atomic component level (**security experts perspective**)
- Determine the composed security properties at the composition level (**software engineers' perspective**); and
- Derive high level security goals based on the above two properties (**end-users' perspective**)

Security Characterization Process



Security at the Component Level



- **Need to characterize the low level security properties**
- **Publishable security properties are mapped with the component functionality that they support.**

Characterization of Security Properties

$$C^i = \{f_1^i, \dots, f_n^i\}$$

$$f_j^i = (E_j^i, R_j^i)$$

$$E_j^i = \{e_{j,1}^i, \dots, e_{j,l}^i\}$$

$$R_j^i = \{r_{j,1}^i, \dots, r_{j,k}^i\}$$

C = component; f = functionality; E = ensured property; R = required property.

***i* = identity of the component C;**

***j* = identity of the functionality f;**

***l* = identity of an ensured security property of j;**

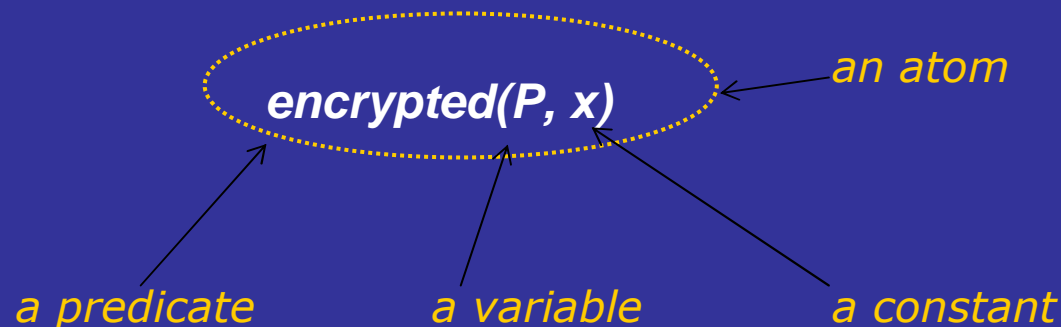
***k* = identity of the required security property of j;**

Security Characterisation Language (SCL)

- Security characterisation language (SCL) based on logic programming and BAN logic
- *Inference rules* are used to test the conformity of the security properties represented in terms of atoms
- A logic programming clause is composed of a *head* and a *body*.
- The symbol \leftarrow is read as ‘*derives*’.
- The left hand side of \leftarrow (the head of the rule) is typically an ensured security property which is a conclusion of an expression.
- The right hand side of \leftarrow (the body of the rule) typically represents the required security properties for a composition.
- In a rule where the body is empty and the head is a single atom such as “ $A \leftarrow$ ”, represents a ensured property which does not depend on any corresponding required properties.
- a clause without a head such as as “ $\leftarrow A$ ” is a mere required security property without any corresponding ensured security property
- Example:
`sees_encrypted(p,x)←encrypted(x,k),owned(k,p).`

Representation of Security Properties

- Security properties are expressed in symbolic notations called *atoms*
- An atom consists of a *predicate* name with a tuple of *variables* or *constants*
- The predicate name of an atom represents a security property such as *encrypted*, *key_generated*, *signed* and so on
- An example of an atom is:



- It states that an entity identified as *p* encrypted the object *x*

Example of Security Properties

security_property(O_{ID} , K_I , D_K)

where

O is an arbitrary finite set of security related operations performed by the entity identified with ID

K is an arbitrary finite set of security attributes such as key, password and so on ;and

D is a set of data that are affected by the operation O, and K is additional security information such as digital signature attached

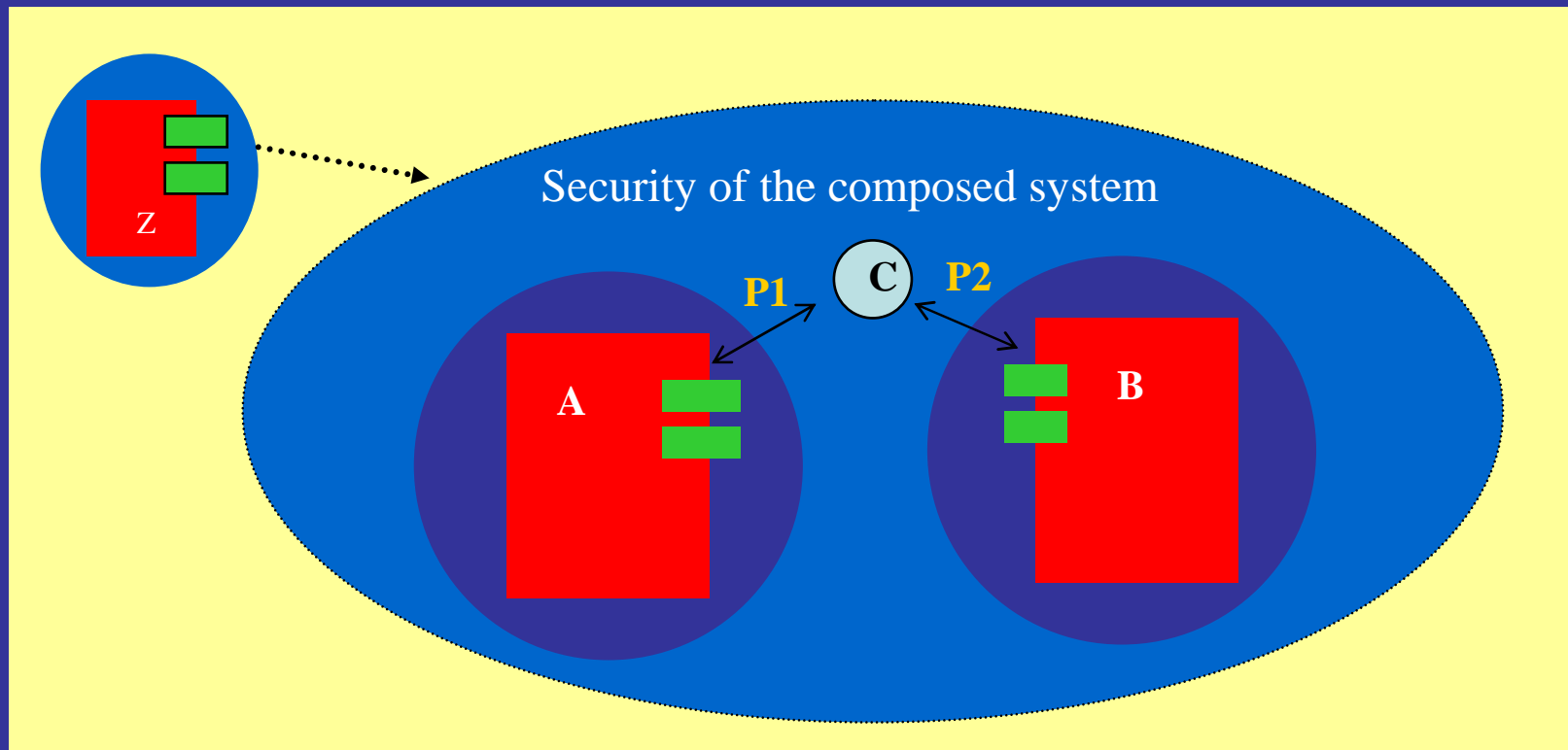
Example:

protect_in_data(encrypt_Q, key_{p+}, 'amount')





ISO/IEC 15408 Common Criteria

- **ISO/IEC 15408 Common Criteria for the Information Technology Security Evaluation, version 2.0**
- **Functional and assurances requirements**
- **Functional requirements comprise eleven security classes**
- **Each class has security families**
- **Security characterisation scheme is based on these classes**

Compositional Security Contract



Legends:

Component	
Interface signature	
Environment	
Security properties	

Representation of Compositional Security Contracts

$$CsC_{i,j} = c^i \xrightarrow{f_n^{c^j}} c^j = (E_{f_n}^{c^i} \Leftarrow R_{f_n}^{c^j}) \wedge (E_{f_n}^{c^j} \Leftarrow R_{f_n}^{c^i})$$

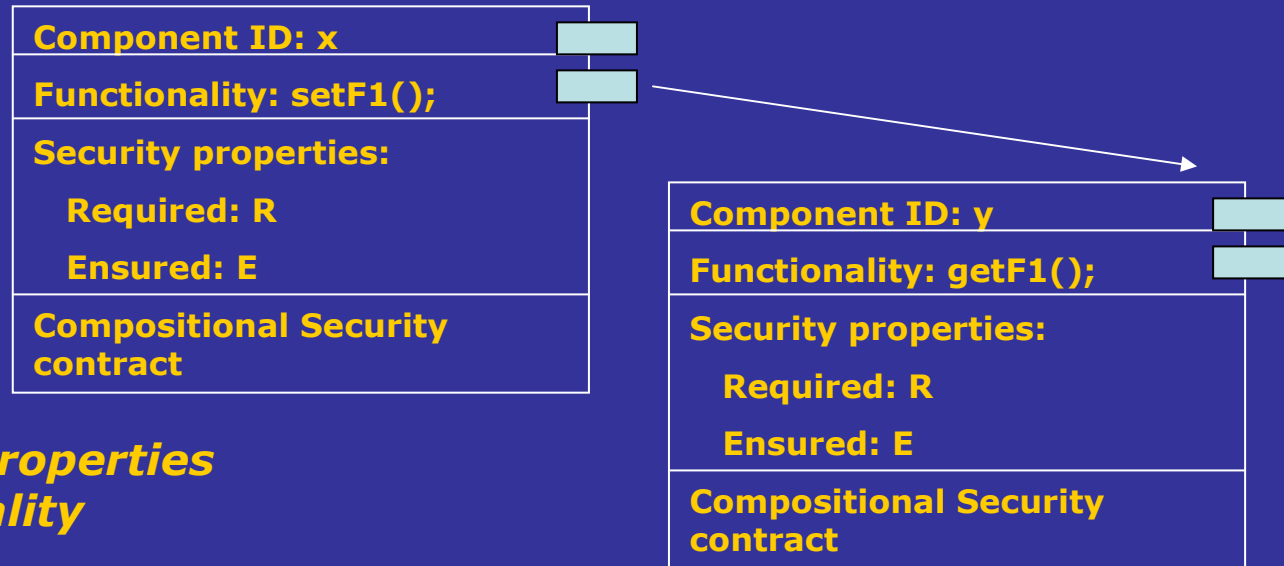
- A CsC is based on the degree of conformity between the security properties of two components
- $C_{a,b} = (\text{ensured}_b \leftarrow \text{required}_a) \wedge (\text{ensured}_a \leftarrow \text{required}_b)$
- Examples

$\text{ensured}_b = \text{protect_out}(\text{encrypt}_b, \text{key}_{A+}, \text{'diagnosis'})$

$\text{required}_a = \text{protect_in}(\text{encrypt}_B, \text{key}_{a+}, \text{'diagnosis'})$

Example of a Composition

We have codified the security functional properties defined in the ISO /IEC 15408 standard, Common Criteria.



Step 1: Read security properties related to the functionality

Step 2: Check compatibility of security properties

Step 3: Make a contract if consistent.

Compositional rule

$CsC(x, y) \leftarrow \text{encrypted}(\text{amount}, k^x), \text{signed}(\text{amount}, k^{-y})$

A compositional contract is established between two entities **x** and a **y** if the amount is encrypted with the public key **k** and it is digitally signed with the private key of **y**

Security Goals for the End-users

- A security goal is the ultimate security objective of a security property
- It is achieved as a result of the implemented security properties in the components
 - For example, a security goal could be defined as the
 - integrity of an object or a piece of data;
 - Confidentiality of a message;
 - Authentication of an entity or user
 - Authorization for an operation on certain object or data; or
 - Non-repudiation of a message and so on.
- The security goals of a composite system expose the specific security objectives achieved in the system.

Security Goals

- Formulated all security classes and their properties into Security Goals:
 - Cryptographic support
 - User Data protection
 - Identification and authentication
 - Security Audit
 - Privacy etc

Ensured properties	Descriptions
$owned(K, P)$	Confidentiality
$encrypted(X, K)$	Confidentiality
$signed(X, K)$	Authenticity, Integrity, nonrepudiation
$believes_produced(P, Q, X)$	Authenticity, nonrepudiation

Example

$CsC(x, y) \leftarrow \text{encrypted}(\text{amount}, k^x), \text{signed}(\text{amount}, k^{-y})$

The above does not state what type of security goal is actually achieved in the composition

- Need to spell out underlying security goals embedded in the compositional rules:
 - **Confidentiality** of the object amount is achieved with the atom: $\text{encrypted}(\text{amount}, k^x)$.
 - **Authenticity** of the entity y is established with the property: $\text{signed}(\text{amount}, k^{-y})$.
 - **Non-repudiation** is also achieved with the property: $\text{signed}(\text{amount}, k^{-y})$.
- Propose a format: $\text{goal}(X) \leftarrow \text{security_property}(X, Y, Z)$
- Security properties are mapped onto security goals:
 - $\text{Confidentiality}(x,y) \leftarrow \text{encrypted}(\text{amount}, k^x)$
 - $\text{Authenticity}(Y) \leftarrow \text{signed}(\text{amount}, k^{-y})$
 - $\text{Nonrepudiation}(Y) \leftarrow \text{signed}(\text{amount}, k^{-y})$

Conclusion

The main contributions of the paper include:

- Argued for a need for the separation of end-users' perspective from the software engineers' perspective
- A formulation technique of security goals based on security properties defined in Common Criteria
- An approach for deriving security goals from the implemented security properties in a composition.
- **More research needed in**
 - Standardization of security properties which could be universally used across the application domain
 - Defining an acceptable format for defining rules and reasoning engine of the derivation approach
 - The scalability of the proposed techniques needs to be examined with more realistic application systems.