ASSERT4SOA

Advanced Security Service cERTificate for SOA

Security Certification of Services

Claudio Ardagna, Ernesto Damiani *Università degli Studi di Milano*
Michele Bezzi, *SAP*

Motivation (1)

- Service-Oriented Architecture (SOA)
  - Business processes developed and deployed by means of multiple services communicating over the Net
  - Runtime composition of services made available by single suppliers
  - Remote users access services on a global ICT infrastructure
- Applications exposed to **new security risks and threats**
- Users increasingly concerned about the security of services

Motivation (2)

- SOA requires **re-thinking** of development, testing, and verification methodologies
- **Software assurance** for services to increase users' confidence and enact service composition
- **Certification** can play a role to establish a trust model suitable for (open) service ecosystems
  - Software or people can rely on the asserted properties, provided that the process of certification produces sufficient evidence

Motivation (3)

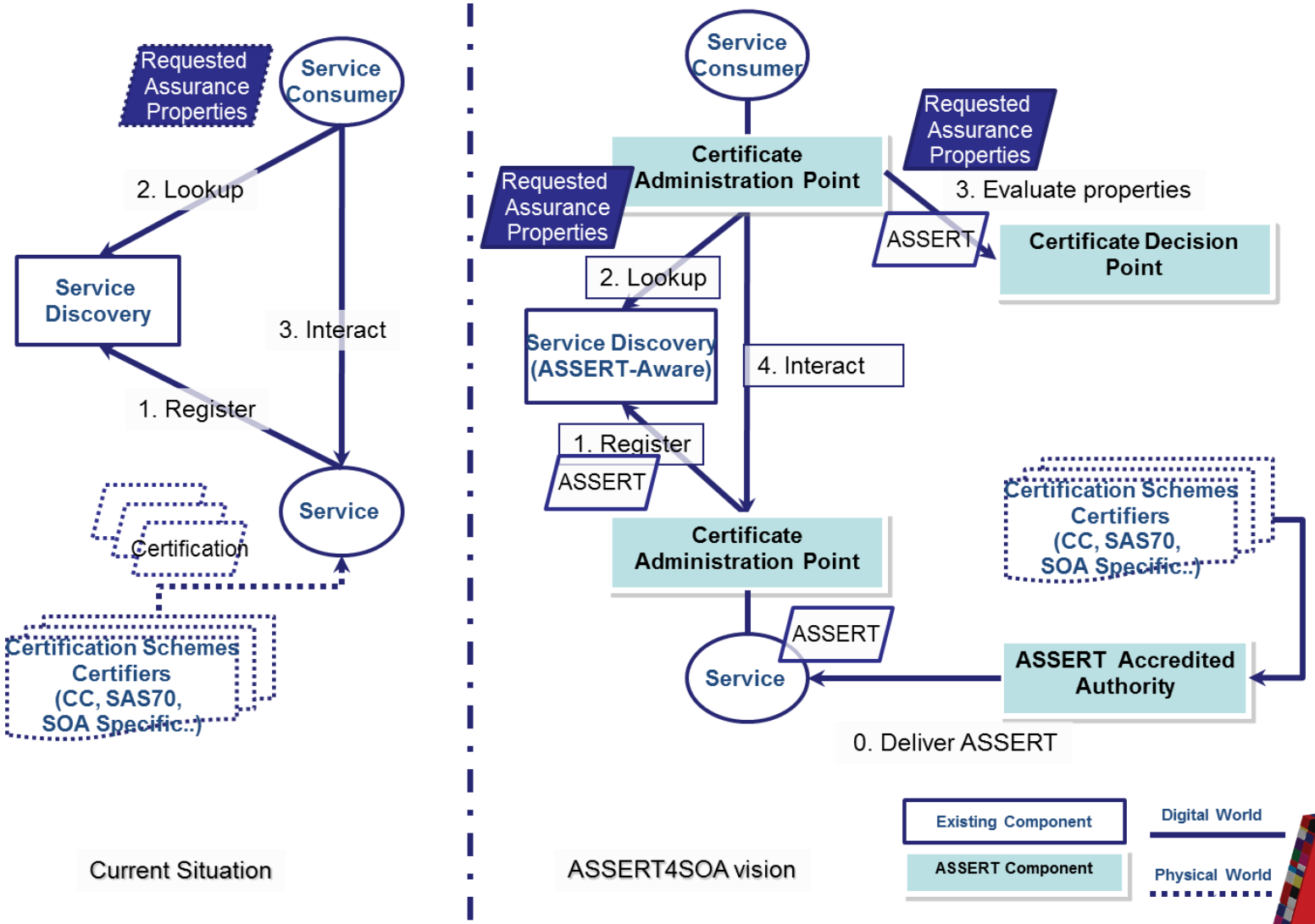
- Existing certification techniques and protocols are **not suitable** for services
  - Defined for traditional **monolithic** software components
  - Provide engineers in charge of software procurement with **human-readable evidences** signed by a trusted third party
  - Change in the system structure requires re–certification

- Service-oriented certification techniques and protocols
  - Require **dynamic and machine-readable certificates**
  - Require support for dynamic changes of components (i.e., at run–time)
  - Should be integrated in **run-time** service discovery and selection, and composition processes

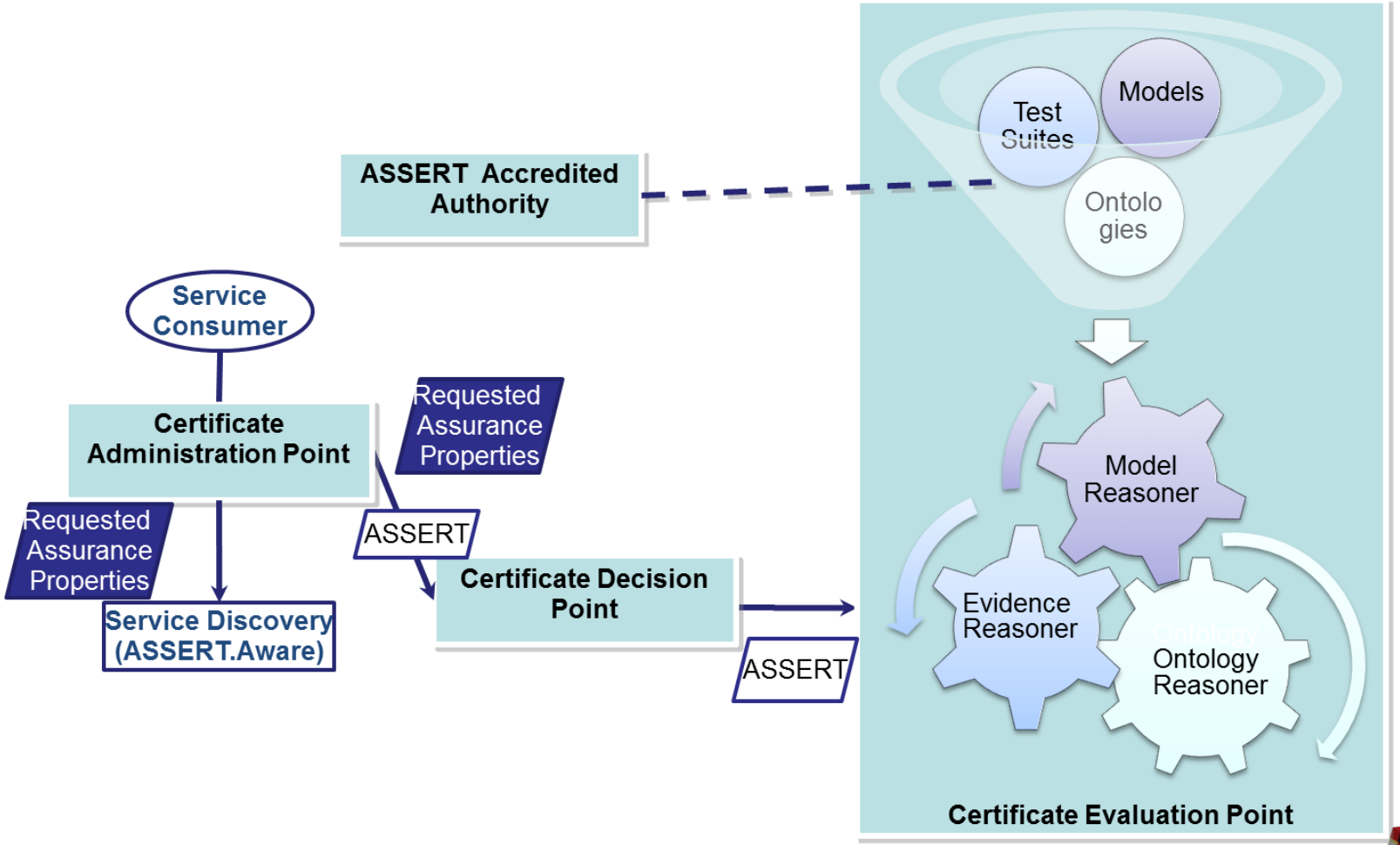
Advanced Security Service cERTificate for SOA (ASSERT4SOA)

- ASSERT4SOA project aims to
  - Produce **novel techniques and tools** – fully integrated within the SOA lifecycle – for expressing, assessing, and certifying security properties for complex service-oriented applications
    - Enable a **multi-party trust model** suitable for open service ecosystems
  - Integrate security certification in the **SOA service lifecycle**
    - Enable automatic processing of **security certifications** for complex service-oriented applications
  - Extend SOA infrastructure for certificate-based selection and comparison of services
    - Increase users' confidence on services and enable assurance-driven service composition

ASSERT4SOA Vision



ASSERT4SOA Certification



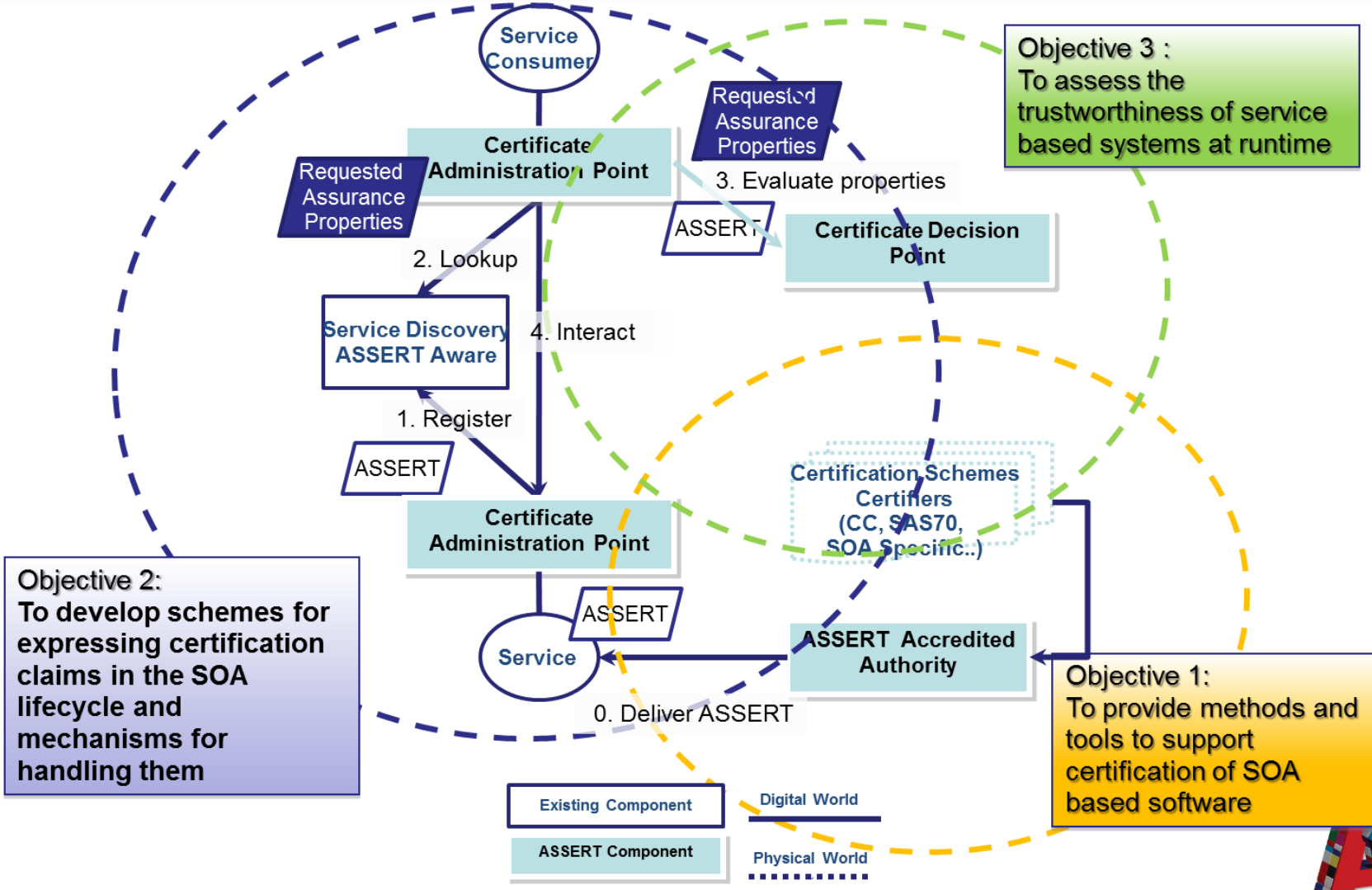
ASSERT4SOA Certificate Classes

- **Test-based certification** provides evidence-based proofs that a test carried out on the software has given a certain result, which in turn shows (perhaps with a certain level of uncertainty) that a given property holds for that software

- **Model-based certification** provides formal proofs that an abstract model (e.g., a set of logic formulas, or a formal computational model such as a finite state automaton) representing a software system holds a given property

- **Ontology-based certification** provides a solution to issue an ASSERT4SOA certificate starting from the certificates of a given software product (e.g., Common Criteria)

ASSERT4SOA Objectives



Test-based Certification of Services

Use Case: Remote Secure Storage

- Remote clients (e.g., software agents acting on behalf of human users, complex services) need a remote secure storage service
- The remote secure storage service allows users to remotely store, delete, update, and retrieve files, and browse folder directories
- The clients use the ASSERT4SOA framework to locate a service that matches their functional needs, as well as their requirements in terms of security assurance

Evidence-Based Certification

- Certification scheme for services
  - Evidence–based certification of services
  - Evidence–based certificates
  - A solution to manage, compare, and match service security certifications based on **testing**
- **Service composition process** driven by the analysis of certified properties of individual services at selection time
- A (certifiably correct) **inference process** that starting from certified properties of individual services computes the properties of the composed service

Evidence-Based Certificates

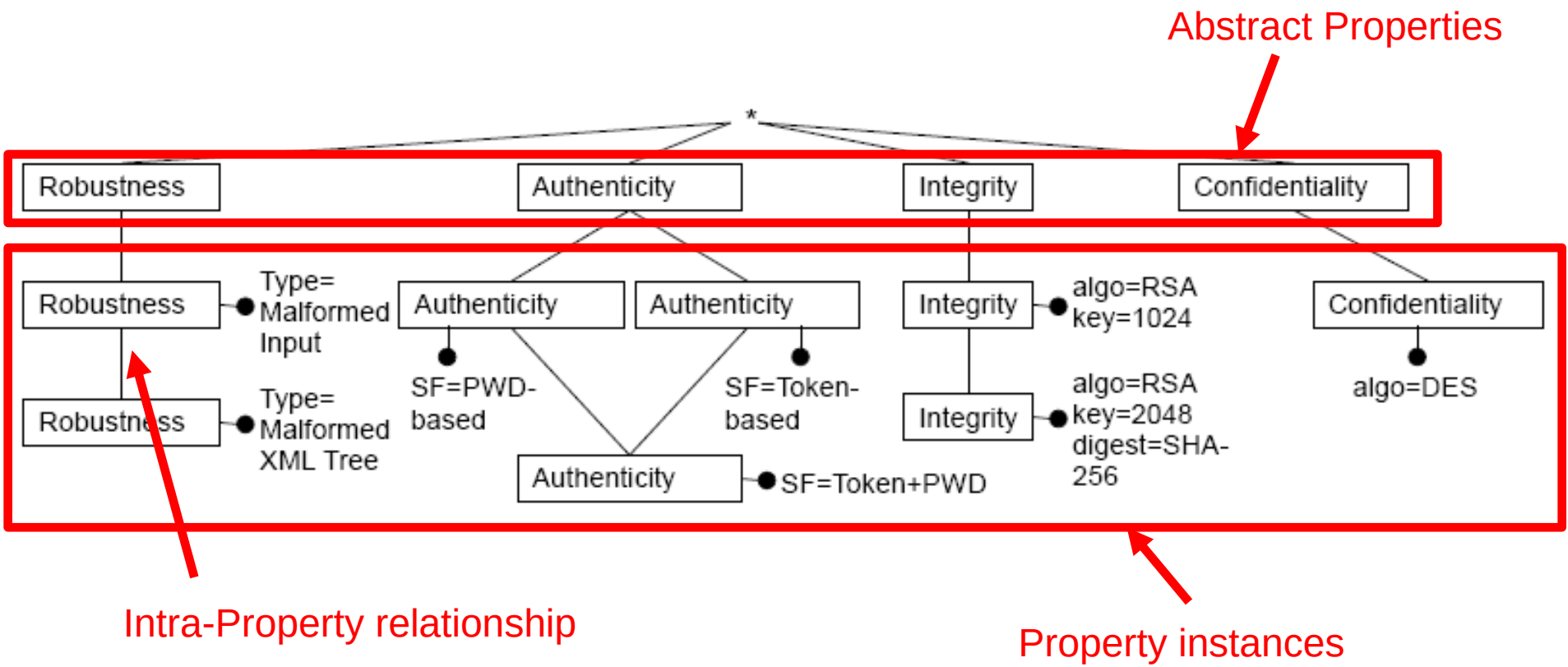
- **Evidence-based proofs** that a test carried out on the service has given a result
  - Support for some property to hold
- Require **machine-readable** (XML-based) certificates specifying
  - Security properties
  - Test-based evidence
- Support **dynamic selection** of single services

Hierarchy of Security Properties (1)

- **Abstract security properties**, generic security requirements for the service under evaluation (e.g., *Confidentiality*, *Integrity*)

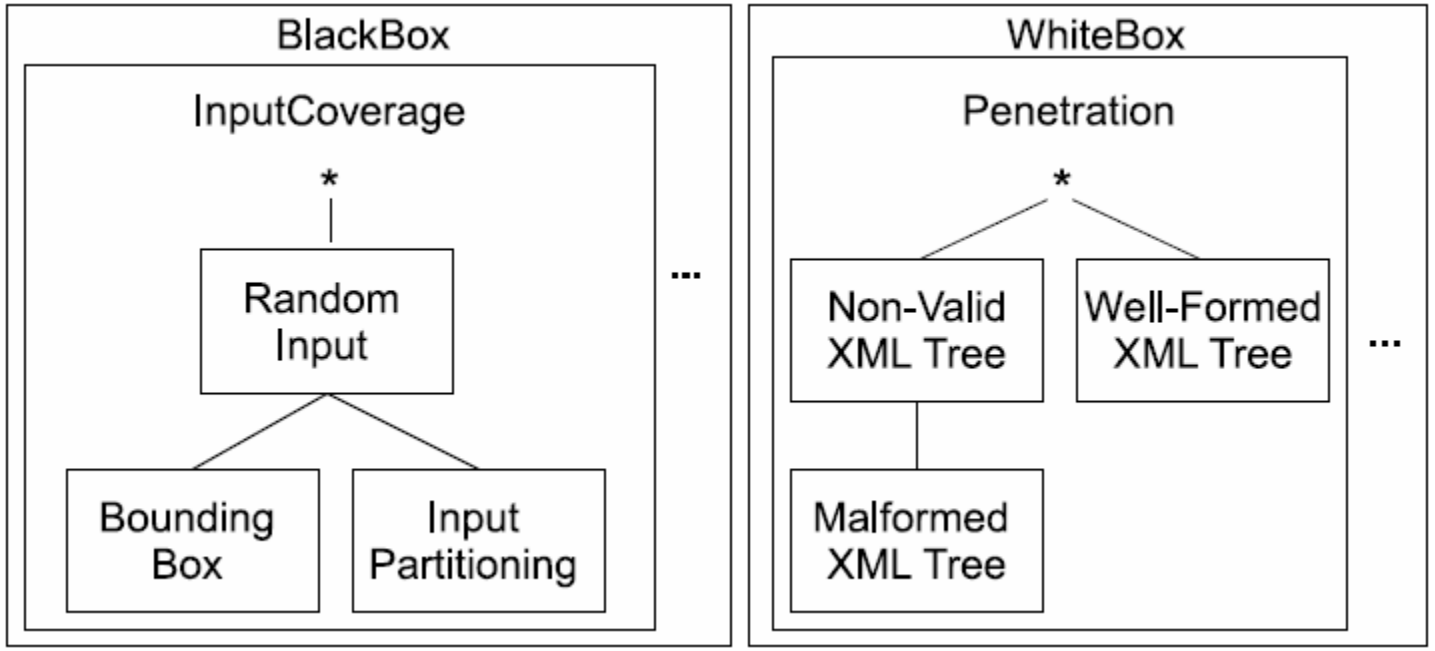
- **Property instances**, abstract properties enriched with a set of "class attributes"
  - Properties to be certified
  - Domain of each attribute has a partial/total order relationship
  - Example: *confidentiality* property with a DES algorithm and a key length of 128bits

Hierarchy of Security Properties (3)



Classes of Tests

- Each security property associated with one or more **test classes**
- Each test class contains a set of **test types**
- Test types organized in **hierarchies**



Test-Based Certification: Integrity of the Remote Secure Storage (1)

- The integrity of files and information should be guaranteed both on the **communication channel** and on the **physical storage**
  - Message signature
  - File signature
- To certify integrity of the service the (Lab accredited by the) certification authority must
  - Evaluate the **integrity of the message (data in transit)** by executing test cases proving that only files with a valid signature are processed and accepted
  - Evaluate the **integrity of the file (data at rest)** by executing test cases proving that only files with valid signatures are stored in the backend

Test-Based Certification: Integrity of the Remote Secure Storage (2)

- **Property**: Integrity
  Class Attributes: algorithm=RSA, digest=SHA-256, |key|=1024bit
- Test cases on message signature
  - TC1 (Valid Signature)
    - INPUT: $Message_i$ + Valid Signed Info
    - EXPECTED OUTPUT: $decrypt_{Pi}$[Signed Info] = digest[$Message_i$]
  - TC2 (Invalid Signature – Attack Modification of Signed Info)
    - INPUT: $Message_i$ + Invalid Signed Info
    - EXPECTED OUTPUT: $decrypt_{Pi}$[Signed Info] ≠ digest[$Message_i$] (FAIL)
  - TC3 (Malformed Header with Wrapper - Modified Body: Attack XML Signature Wrapping)
    - INPUT: $Message_i$ + Wrapper
    - EXPECTED OUTPUT: $decrypt_{Pi}$[Signed Info] ≠ digest[$Message_i$] (FAIL)

Test-Based Certification: Integrity of the Remote Secure Storage (3)

- **Property**: Integrity
  Class Attributes: algorithm=RSA, digest=SHA-256, |key|=1024bit
- Test cases on file signature
  - TC1 (Valid Signature)
    - INPUT: $File_i$ + Signed Digest
    - EXPECTED OUTPUT: $decrypt_{Pi}$[Signed Digest] = digest[$File_i$]
  - TC2 (Invalid Signature)
    - INPUT: $File_i$ + Signed Digest
    - EXPECTED OUTPUT: $decrypt_{Pi}$[Signed Digest] ≠ digest[$File_i$] (FAIL)

Certification-Aware SOA

- Service certification scheme to be **integrated** within the existing SOA infrastructure
  - Clients define **preferences** in terms of **certified properties, evidence,** and **tests**
  - **Security certificates** are awarded to the services

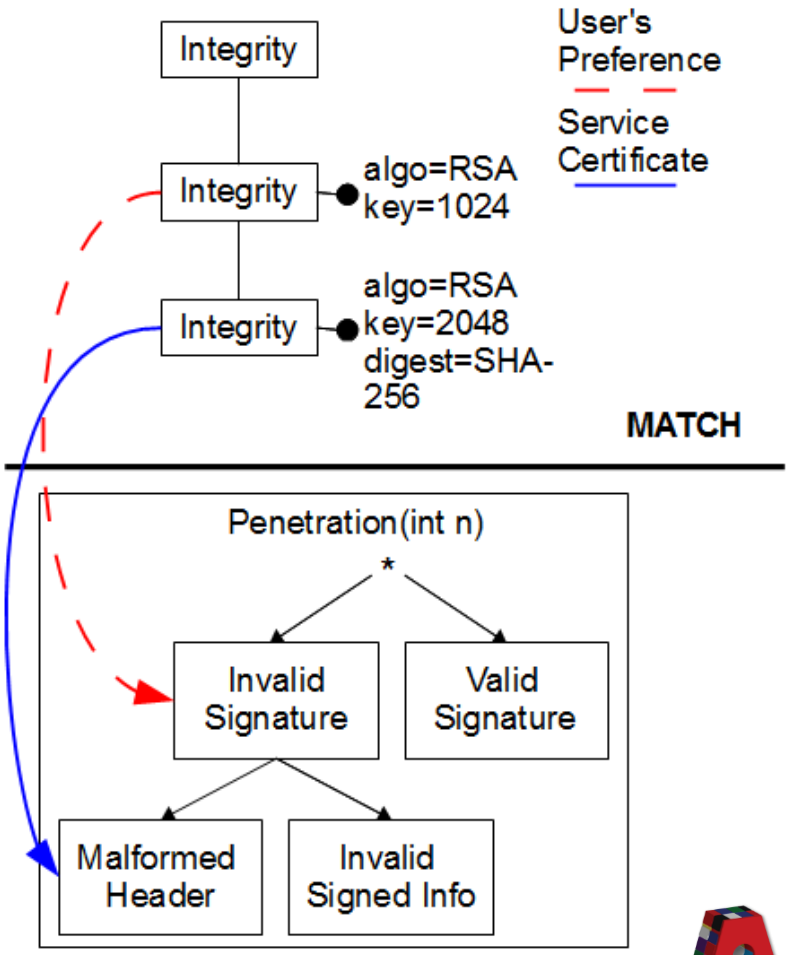
- Support runtime selection of services based on security certificates and clients preferences (matchmaking)
  - **Matching process**: a client searches services that expose a level of assurance (certificate) compatible with its preferences
  - **Comparison process**: a client compares functionally equivalent services with different certificates (partial order of services)

Matching Process

- **Matching process**
  - The client defines its preferences in terms of **requirements on security properties and evidences**
  - It automatically matches them against the certificates awarded to the services
  - It retrieves a **compatibility list** including all services that satisfy the client's preferences
- **Double matching**
  - **(property match)** there is a *relation* in the hierarchy between properties in the certificate and preferences
  - **(evidence match)** tests in the certificate satisfy the ones in the preferences

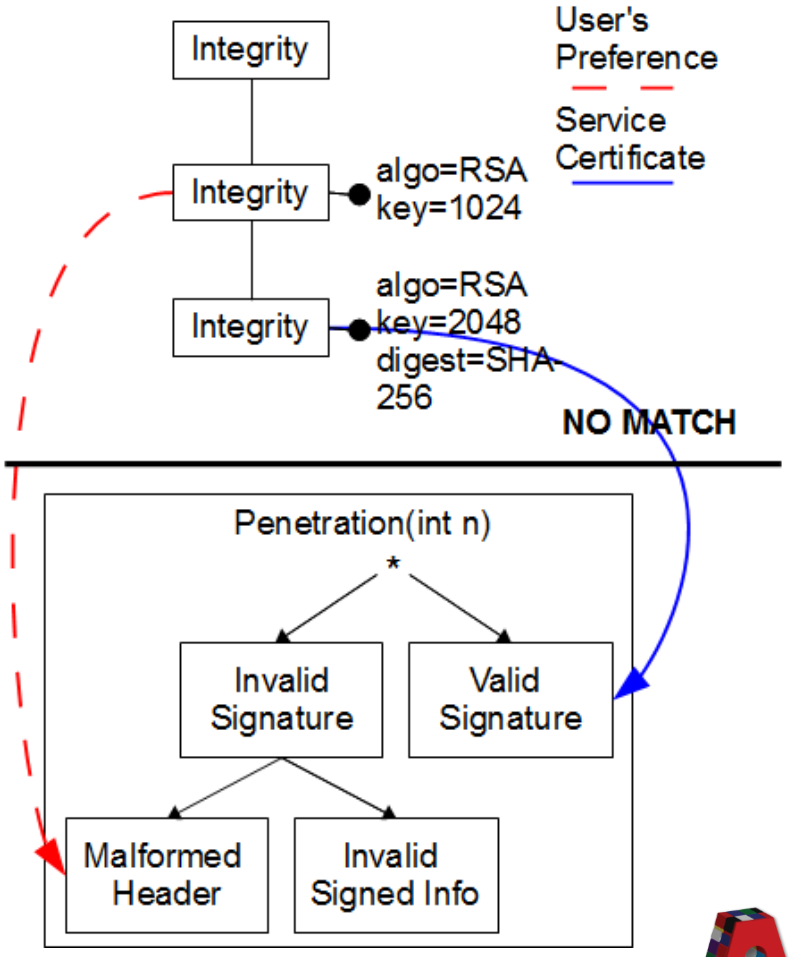
Matching Example (1)

- Preferences
  - Property: Integrity with RSA algorithm and a key of 1024 bit
  - Evidence: m penetration tests using invalid signature
- Certificate
  - Property: Integrity with RSA algorithm and a key of more than 1024 bit
  - Evidence: k>m penetration tests using Malformed Header with Wrapper (e.g., XML signature wrapping)



Matching Example (2)

- Preferences
  - Property: Integrity with RSA algorithm and a key of 1024 bit
  - Evidence: m penetration tests using Malformed Header with Wrapper
- Certificate
  - Property: Integrity with RSA algorithm, a key of 1024 bit, and SHA-256
  - Evidence: k>m penetration tests using valid signature



Next Steps

- Definition of the ASSERT4SOA framework and architecture to support certificate lifecycle (issuing, binding to service instances, update, revocation, negotiation and protection)
- Definition of the ASSERT4SOA language to specify all types of certificates
- Definition of the algorithms for certificate matching and comparison
- Specification of a certificate-aware service discovery supporting dynamic selection and discovery of services, and runtime composition

Conclusions

- Certification of services can be used to establish **trust in SOA**
- ASSERT4SOA is aimed at providing **techniques and tools** – fully integrated within the SOA lifecycle – for supporting a SOA-enhanced certification process
- Certification will increase users' confidence on service and enable **assurance-driven service composition**
  - **Preference-based** selection and integration

Thank you!

- Advanced Security Service cERTificate for SOA (ASSERT4SOA)
  - You live in a certified house
    you drive a certified car
    why would you use an uncertified service?

- For more information or to subscribe to the project newsletter http://www.assert4soa.eu/

Thank you for the attention