World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:1, No:8, 2007

Design of Integration Security System using XML Security

Juhan Kim, Soohyung Kim, and Kiyoung Moon

***Abstract*—**In this paper, we design an integration security system that provides authentication service, authorization service, and management service of security data and a unified interface for the management service. The interface is originated from XKMS protocol and is used to manage security data such as XACML policies, SAML assertions and other authentication security data including public keys. The system includes security services such as authentication, authorization and delegation of authentication by employing SAML and XACML based on security data such as authentication data, attributes information, assertions and polices managed with the interface in the system. It also has SAML producer that issues assertions related on the result of the authentication and the authorization services.

***Keywords*—**XML, XML Security, XACML.

I. Introduction

IN recent years, Web Services concepts have been introducing, specifications and products about it publishing and developing. Web Services security technologies for Web Services have been also standardizing, and XML security, core technologies of Web Services security, is almost specified by standardization organizations. Some XML security technologies such as XML Signature and XML Encryption have already became international standards and been applying at security applications.

The XML Security is XML-based security technologies adding XML syntax and processing rules to legacy cryptographic and security technologies for providing flexible, extensible and practical characteristic. In XML security, there are XKMS (XML Key Management Specification), XACML (eXtensible Access Control Markup Language), SAML (Security Assertion Markup Language), XML Encryption, XML Signature and the like, which are employing as XML protocol for PKI, access control to managed XML Security data, storing and transmitting the security information, authenticating one's identity, and other reasons, respectively.

XML signature aims to guarantee integrity and authentication to any digital content including XML documents. XML Encryption is a method whereby XML content can be transformed such that it is discernible only to the intended recipients and opaque to all others. XACML is to provide a consistent policy language for legacy access control products and systems to have interoperability and guarantee integration about each other, increase trust on carrying out security policies, and save cost on editing them. SAML is a framework for specifying and sharing "trust assertions" in XML.A trust assertion can be any data used to determine authorization, such as credentials, credit ratings, approved roles, and so on. PKI is complex security system environment providing encryption and digital signature through public key algorithm. XKMS defines trusted Web services for managing cryptographic keys, including public keys. XKMS services are currently defined as X-KISS (XML Key Information Service Specification) that supports services used by a party relying on a cryptographic key (location, and validation) and X-KRSS (XML Key Registration Service Specification) that supports services used by the holder of a cryptographic key (registration, revocation, reissue, and key recovery).

In this paper, we design an integration security system that provides authentication service, authorization service, and management service of security data and a unified interface for the management service by using the XML security. The interface is originated from XKMS protocol and is used to manage security data such as XACML policies, SAML assertions and other authentication security data including public keys.

The system includes authorization and delegation of authentication services by employing XACML policy decision point (PDP) and SAML PDP that decide authorization and authentication based on the security data such as authentication data, attributes information, assertions and polices managed with the interface in the system. It also has SAML producer that issues assertions related on the result of the authentication and the authorization services.

The system can be standalone system irrelative to its application platform. It can be also very useful to some systems that need authentication, integrated authorization, fine-grained access control, and secure protocol and management of security data among them, respectively. In particular, the delegation services can be useful to mobile environment that is hard to support legacy security services of wire environment.

In the next chapter, we introduce the overview of XML security. Then we describe the system using XML security and explain how it consists of, how it works and how it can be applied. Finally, we conclude by explaining about future works.

Juhan Kim Soohyung Kim, and Kiyoung Moon are with the Electronics and Telecommunications Research Institute (ETRI), 161 Gajeong-dong, Yuseong-gu, Daejeon, 305-350, Korea (e-mail: juhankim, lifewsky, kymoon@etri.re.kr).

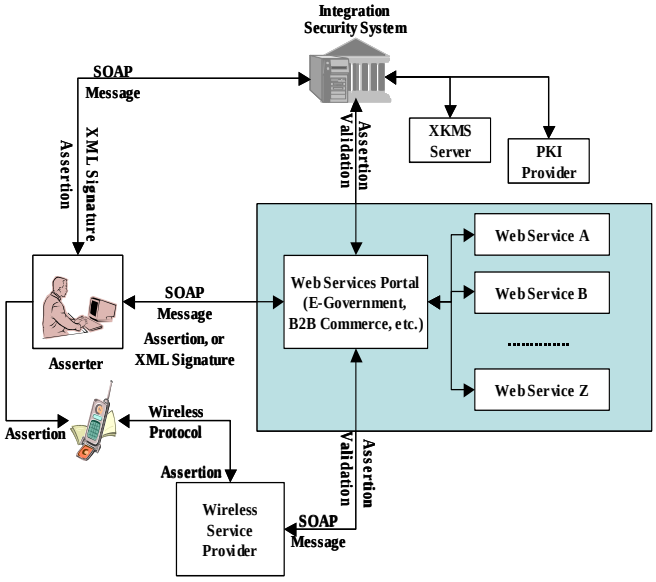

Fig. 1 The overview of the integration security system

II. Overview of XML Security

XML Security technologies include XML Signature for integrity and signing solutions, XML Encryption for confidentiality, XKMS for public key registration, location and validation, SAML for conveying authentication, authorization and attribute assertions, and XACML for defining access control rules.

A key objective of XKMS protocol design is to minimize the complexity of client application implementations by shielding them from the complexity and syntax of the underlying PKI used to establish trust relationships [1, 2].

XACML describes both a policy language and an access control decision request/response language. The policy language is used to describe general access control requirements, and has standard extension points for defining new functions, data types, combining logic, etc. The request/response language lets you form a query to ask whether a given action should be allowed, and interpret the result [3, 4].

SAML is a security credential standard. SAML provides standardized ways to use XML to represent security credentials and a protocol for requesting and receiving credential data from a SAML authority service. When combined with the WS-Security specification, SAML can be used to transport credential data in a SOAP message [5,6,7].

SAML Authorities comes in three types: authentication authorities, attribute authorities, and policy decision points (PDP). These three types authorities return three distinct types of assertions:

1) SAML Authentication Assertion—when a SAML Authentication Authority performs an action and, as a consequence, makes a determination about a particular subject's credentials, the result is returned as a SAML Authentication Assertion.
2) SAML Attribute Assertions—once an authentication assertion has been returned, a SAML Attribute Authority may be asked for the attributes associated with the subject. These are returned as a SAML Attribute Assertion.
3) SAML Authorization Assertions—the permissions associated with an authenticated subject with respect to a specific resource are returned by the PDP as a SAML Authorization Assertion.

III. Design of Integration Security System

Main functions of the integration security system we design are security data management and security services such as authentication service, delegation of authentication service and authorization service based on the management. The system can be applied like Fig.1.

A. Design Features

Some features of the system are the following.

1) A unified interface for managing many kinds of security data.
   - It is extending XKMS protocols such as X-KISS and X-KRSS to cover various security data.
2) Various security services based on the managed security data.
   - Authentication service using public key pair, and issuing and managing authentication assertion about the result of the authentication.
   - Authorization service using managed policies, and issuing and managing authorization assertion.
   - Supporting various authentication methods like biometrics data and assertions, and issuing assertion about the result.
3) Delegation of authentication service
   - The service that authenticates one's identity instead of a receiver with the authentication method that the receiver wants, and then that issues an assertion about the result for the receiver.
   - Because of the service, it is possible to support single sign on service on distributed computing environment, even if each terminal in the environment requires heterogeneous authentication method respectively.
   - In particular, since the service makes an assertion after authentication; one can put it on his mobile terminals that have limited resources and use it to mobile web sites for his identity authentication [12].
4) Standalone security system
   - It is natural that the system behaves like the XKMS server, since the system is actually originated from the idea of XKMS server. We modify the protocols such as X-KISS and X-KRSS to support other types of security and add new security services that can utilize such security data well.
5) Extensible system
   - Since the system we design has very flexible interface like the unified interface and independent security services to each other, it is easy to add new security services such as policy decision of digital right management (DRM) and platform for privacy preference (P3P) with least efforts [10,11].

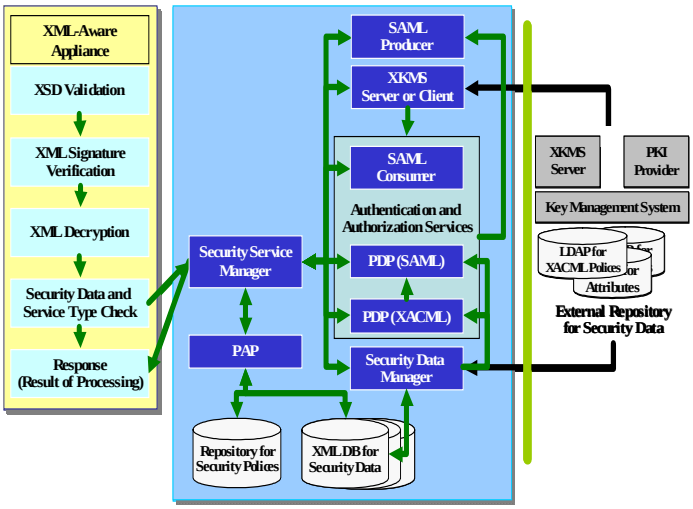

Fig. 2 The structure of the system

B. System Structure

There are four main parts that are XML-aware appliance, security service manager, security data manager and part for authentication and authorization consists of XKMS server, SAML PDP, XACML PDP, and SAML producer and consumer like Fig. 2.

XML-aware appliance takes SOAP message from the outside and then validates schema, verifies signature, decrypts the message, analyzes parameters like data, data type and service type, give them to the security service manager and then get the result from the manager, makes response message, and returns it.

The security service manager gets the parameters like security data and its type, and service type from the XML-aware appliance. It then distributes the parameters to adequate security service modules such as SAML producer, SAML consumer, SAML PDP, XKMS server, XACML PDP, and security data manager as followed.

1) Security data manager: It provides some functions such as validation, generation, storing, and retrieval about each type of security data in XML DB
2) XKMS server: It works like legacy XKMS server. If it do not have public key relevant to a request, the server finds it external XKMS servers that have trust relationship with this server.
3) PDP for SAML and XACML: They cover authentication and authorization service based on authentication results by biometrics, signature and assertion, polices of XML DB, and attributes from external LDAP.
4) SAML producer and SAML consumer: SAML producer makes assertion in according to the result from PDP, while SAML consumer verifies ones authentication that is issued by external SAML producers which have trust relationship with this system.
5) PAP (Policy Administration Point): It manages security data like polices and authentication information by who have rights to manage them. It also manages security polices for supervising this system by an administrator.
6) SOAP message: The system use SOAP to communicates with clients or external XKMS. Extended XKMS schema, the unified interface, is in SOAP body.

C. Protocols for the System

Protocols used in this system are from X-KISS and K-KRSS of XKMS as Fig. 3.

The unified interface consists of extended X-KRSS that can deal with various types of security data and extended X-KISS that is used to get security services. The services of the X-KISS can be achieved with the security data managed by the X-KRSS.

To support the extended X-KISS and X-KRSS, it is required to redefine XKMS schema. Especially, the <KeyBinding AbstractType> element, which includes XML schemas for security data, must be modified like the Fig.7 in Appendix.

The <KeyBindingAbstractType> element in the modified schema includes new elements such as <xacml:Policy>, <xacml:PolicySet>, <xacml-context:Request>, and <xacml-context:Response> for policy decision, <saml:Assertion> for authentication and authorization, <xenc:EncryptedKey> for secret key used at encryption/decryption, <xcbf:Biometric SyntexSets> for supporting biometrics authentication from XML common Biometric Format, and <wsse:Security> for Web Services security token from OASIS.

The figure also shows other modified elements like <xkms: CompoundRequest>, <xkms:CompoundResult> that are used to request a service to the system and get response from the system. The other elements such as <AuthenticationRequest>, <Policy DecisionRequest> and the like are to request authentication service and authorization service to the system, respectively.

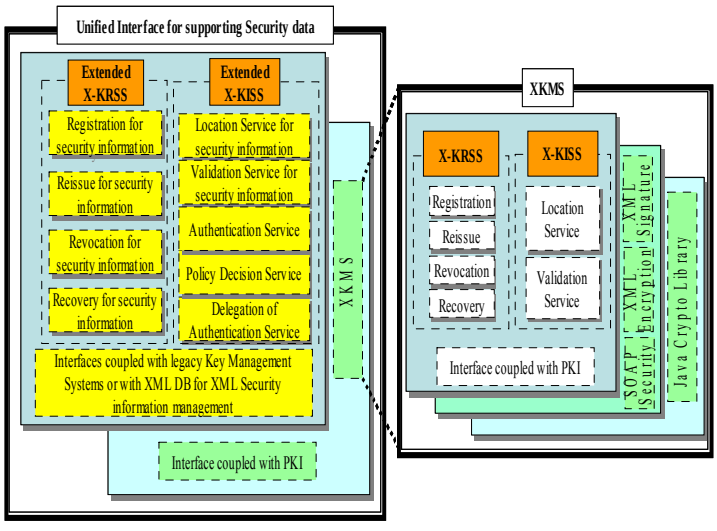

Fig. 3 The unified interface for managing security data is originated from XKMS and new security services such as authentication, policy decision (authorization) and delegation of authentication services are added on it

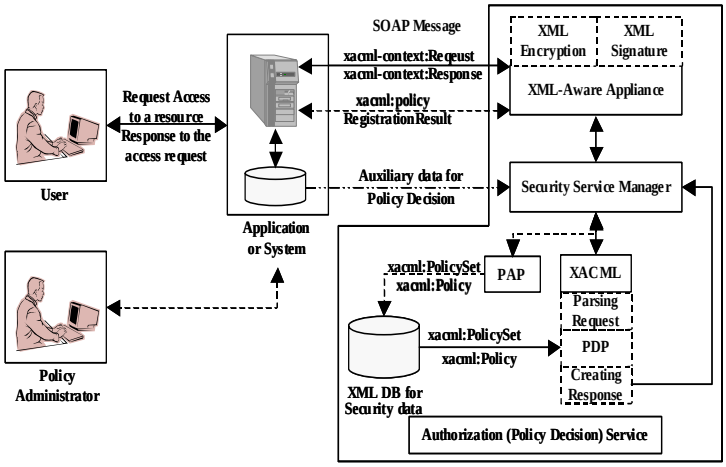

Fig. 4 The processing sequence of policy decision service

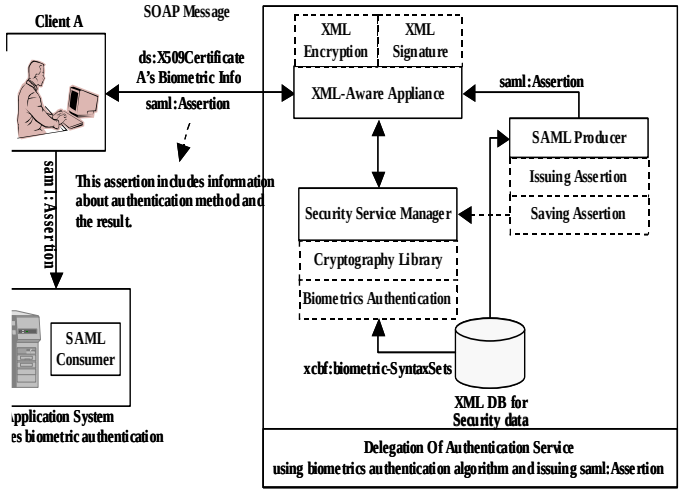

Fig. 5 The processing of delegation of authentication service and a reason why the service is necessary

D. Security Services in the System

There are security services such as authentication, policy decision (authorization) and delegation of authentication services including management service for security data in the system.

To provide policy decision service, it is required to store polices in advance to the system and an application system should supply auxiliary information needed to policy decision as shown in Fig. 4. That is, the application system should have role of context handler in XACML.

A user can get SAML assertion as a result of an access requesting to the system, when the user requests the access to the system directly. Then, in the system, the SAML producer gets the result of PDP instead of creating response context and makes assertions for the user. The user can send it to an application system for taking the access and then the application system confirms the assertion and grants the access.

A user can use the delegation of authentication service like Fig.5, when the user needs other types of authentication such as biometrics authentication at the request of an application system.

E. The System Architecture

The integration security system has the architecture as shown in Fig. 6. There is SOAP that has XML security library including XML Signature and XML Encryption [8,9]. There is an XKMS library part over the XML security library. At the top of this stack, there is schema extension part for the unified interface, service extension part for security data management and security services such as authentication, authorization and delegation of authentication services. SAML and XACML technologies are included in the place of the security services in this layer.

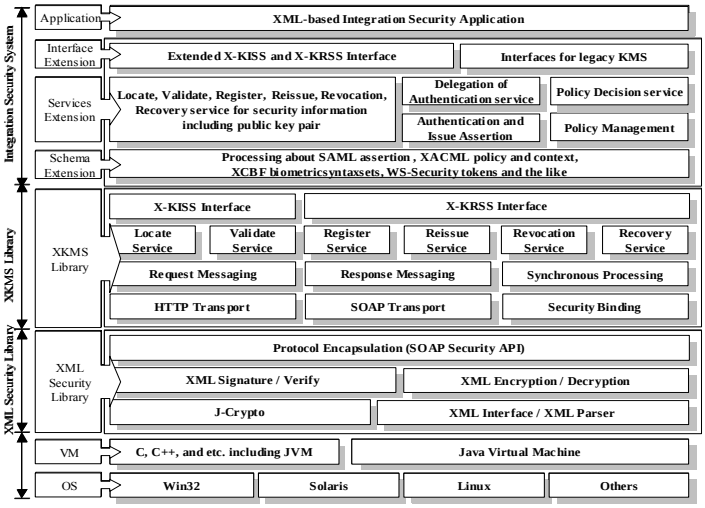

Fig. 6 The architecture for the system

IV. Conclusion

In this paper, we have designed an integration security system that can manage various kinds of security data including public keys with a unified interface, which is extending XKMS protocols. The system also has security services such as authentication, authorization and delegation of authentication services based on such the security data, and it is possible to support single sign on function with SAML assertions from the security services.

In the system we have designed, however, it is required to complement a few security consideration points such as protocol extinction problem between policy administrators and general user, and trust relationship problem how the system can build the relation among XKMS servers and other security data authorities.

This paper has described a system that can provide various security services based on management of many types of security data. Legacy security systems are dependents on application system and could not support such various security

data management and security services like delegation of authentication and policy decision services with single interface. This design can be adapted easily to distribute computing systems that should have different types of security data and requirements, and to XML Web Services environment that needs management about various kinds of XML data in XML security and Web Services security.

REFERENCES

[1] W3C, XML Key Management (XKMS 2.0) Requirements, May-2003.
[2] W3C, XML Key Management Specification Version 2.0, Apr-2003.
[3] OASIS, eXtensible Access Control Markup Language (XACML) Version 1.0, Feb-2003.
[4] OASIS, A Brief Introduction to XACML, Feb-2003.
[5] OASIS, Security Assertion Markup Language, Jan-2003.
[6] OASIS, Web Services Security (WS-Security) Version 1.0 Apr-2002.
[7] Frederick Hirsch, Getting Started with XML Security, http://www.sitepoint.com/, Nov-2003.
[8] W3C, XML Signature Syntax and Processing, Feb-2002.
[9] W3C, XML Encryption Syntax and Processing, Dec-2002.
[10] OASIS, XML Common Biometric Format, Aug-2003.
[11] W3C, The Platform for Privacy Preferences Specification, Apr-2002.
[12] Jongil Jeong, Dongkyoo Shin, Dongil Shin and Kiyoung Moon, Java-Based Single Sign-On Library Supporting SAML for Distributed Web Services, APWeb, April 2004.
[13] Kiyoung Moon et. al., Certificate validation Scheme of Open Grid Service Usage XKMS, GCC 2003.
[14] Namje Park. Et. al.,Development of XKMS-based Service Component for Using PKI in XML Web Service Environment, ICCSA, 2004.


```
<complexType name="KeyBindingAbstractType" abstract="true">
  <sequence>
    <element ref="ds:KeyInfo" minOccurs="0"/>
    <element ref="KeyUsage" minOccurs="0" maxOccurs="3"/>
    <element ref="UseKeyWith" minOccurs="0"
                                    maxOccurs="unbounded"/>
    <element ref="xenc:EncryptedKey" minOccurs="0"/>
    <element ref="xcbf:BiometricSyntaxSets" minOccurs="0"/>
    <element ref="saml:Assertion" minOccurs="0"/>
    <element ref="xacml:Policy" minOccurs="0"/>
    <element ref="xacml:PolicySet" minOccurs="0"/>
    <element ref="xacml-context:Request" minOccurs="0"/>
    <element ref="xacml-context:Response" minOccurs="0"/>
    <element ref="wsse:Security" minOccurs="0"/>
    <element ref="wsse:BinarySecurityToken" minOccurs="0"/>
    <element ref="wsse:UserNameToken" minOccurs="0"/>
    <element ref="wsse:SecurityTokenReference" minOccurs="0"/>
  </sequence>
<attribute name="Id" type="ID" use="optional"/> </complexType>
```

```
<element name="CompoundRequest"
                type="xkms:CompoundRequestType"/>
<complexType name="CompoundRequestType">
  <complexContent>
    <extension base="xkms:RequestAbstractType">
      <choice maxOccurs="unbounded">
        <element ref="xkms:LocateRequest" />
        <element ref="xkms:ValidateRequest" />
        <element ref="xkms:RegisterRequest" />
        <element ref="xkms:ReissueRequest" />
        <element ref="xkms:RecoverRequest" />
        <element ref="xkms:RevokeRequest" />
        <element ref="u-kms:AuthenticationRequest" />
        <element ref="u-kms:PolicyDecisionRequest" />
        <element ref="u-kms:DelegationRequest" />
      </choice>
    </extension>
  </complexContent> </complexType> </element>
```

```
<element name="u-kms:PolicyDecisionRequest"
            type="u-kms: PolicyDecisionRequestType"/>
<complexType name="PolicyDecisionRequestType">
  <complexContent>
    <extension base="xkms:RequestAbstractType">
      <sequence>
        <element ref="xacml-context:Request" />
      </sequence> </extension>
  </complexContent> </complexType> </element>

<element name="u-kms:PolicyDecisionResult"
                type="u-kms:PolicyDecisionResultType"/>
<complexType name="PolicyDecisionResultType">
  <complexContent>
    <extension base="xkms:ResultType">
      <sequence>
        <element ref="xacml-context:Response" />
      </sequence> </extension>
  </complexContent> </complexType> </element>
```

```
<element name="CompoundResult"
                type="xkms:CompoundResultType"/>
<complexType name="CompoundResultType">
  <complexContent>
    <extension base="xkms:ResultType">
      <choice minOccurs="0" maxOccurs="unbounded">
        <element ref="xkms:LocateResult" />
        <element ref="xkms:ValidateResult" />
        <element ref="xkms:RegisterResult" />
        <element ref="xkms:ReissueResult" />
        <element ref="xkms:RecoverResult" />
        <element ref="xkms:RevokeResult" />
        <element ref="u-kms:AuthenticationResult" />
        <element ref="u-kms:PolicyDecisionResult" />
        <element ref="u-kms:DelegationResult" />
      </choice>
    </extension>
  </complexContent>
</complexType>
</element>
```

```
<element name="u-kms:AuthenticationRequest" type=
                "u-kms:AuthenticationRequestType"/>
<complexType name="AuthenticationRequestType">
  <complexContent>
    <extension base="xkms:RequestAbstractType">
      <sequence>
        <element ref="xkms:RecoverKeyBinding" />
        <element ref="xkms:Authentication" />
      </sequence> </extension>
    <attribute name="u-kms:ReturnType" type="string"
                                    use="required"/>
  </complexContent> </complexType> </element>

<element name="u-kms:AuthenticationResult"
                type="u-kms:AuthenticationResultType"/>
<complexType name="AuthenticationResultType">
  <complexContent>
    <extension base="xkms:ResultType">
      <sequence>
        <element ref="xkms:UnverifiedKeyBinding" />
      </sequence> </extension>
  </complexContent> </complexType> </element>
```

```
<element name="u-kms:DelegationRequest" type=
                "u-kms:DelegationRequestType"/>
<complexType name="DelegationRequestType">
  <complexContent>
    <extension base="xkms:RequestAbstractType">
      <sequence>
        <element ref="xkms:RecoverKeyBinding" />
        <element ref="xkms:Authentication" />
      </sequence> </extension>
  </complexContent> </complexType> </element>

<element name="u-kms:DelegationResult"
                type="u-kms:DelegationResultType"/>
<complexType name="DelegationResultType">
  <complexContent>
    <extension base="xkms:ResultType">
      <sequence>
        <element ref="xkms:UnverifiedKeyBinding" />
      </sequence> </extension>
    <attribute name="u-kms:AuthenticationType"
                type="string" use="required"/>
  </complexContent> </complexType> </element>
```

Fig. 7 The modified schema for the unified interface