International Telecommunication Union

ITU-T

TELECOMMUNICATION
STANDARDIZATION SECTOR
OF ITU

X.1143
(11/2007)

SERIES X: DATA NETWORKS, OPEN SYSTEM COMMUNICATIONS AND SECURITY

Telecommunication security

Security architecture for message security in mobile web services

ITU-T Recommendation X.1143

ITU-T

ITU-T X-SERIES RECOMMENDATIONS

DATA NETWORKS, OPEN SYSTEM COMMUNICATIONS AND SECURITY

PUBLIC DATA NETWORKS	
Services and facilities	X.1–X.19
Interfaces	X.20–X.49
Transmission, signalling and switching	X.50–X.89
Network aspects	X.90–X.149
Maintenance	X.150–X.179
Administrative arrangements	X.180–X.199
OPEN SYSTEMS INTERCONNECTION	
Model and notation	X.200–X.209
Service definitions	X.210–X.219
Connection-mode protocol specifications	X.220–X.229
Connectionless-mode protocol specifications	X.230–X.239
PICS proformas	X.240–X.259
Protocol Identification	X.260–X.269
Security Protocols	X.270–X.279
Layer Managed Objects	X.280–X.289
Conformance testing	X.290–X.299
INTERWORKING BETWEEN NETWORKS	
General	X.300–X.349
Satellite data transmission systems	X.350–X.369
IP-based networks	X.370–X.379
MESSAGE HANDLING SYSTEMS	X.400–X.499
DIRECTORY	X.500–X.599
OSI NETWORKING AND SYSTEM ASPECTS	
Networking	X.600–X.629
Efficiency	X.630–X.639
Quality of service	X.640–X.649
Naming, Addressing and Registration	X.650–X.679
Abstract Syntax Notation One (ASN.1)	X.680–X.699
OSI MANAGEMENT	
Systems Management framework and architecture	X.700–X.709
Management Communication Service and Protocol	X.710–X.719
Structure of Management Information	X.720–X.729
Management functions and ODMA functions	X.730–X.799
SECURITY	X.800–X.849
OSI APPLICATIONS	
Commitment, Concurrency and Recovery	X.850–X.859
Transaction processing	X.860–X.879
Remote operations	X.880–X.889
Generic applications of ASN.1	X.890–X.899
OPEN DISTRIBUTED PROCESSING	X.900–X.999
TELECOMMUNICATION SECURITY	**X.1000–**

For further details, please refer to the list of ITU-T Recommendations.

ITU-T Recommendation X.1143

Security architecture for message security in mobile web services

Summary

ITU-T Recommendation X.1143 describes the security architecture and scenarios for message security in mobile web services.

Security services for messages are the most fundamental security requirements for mobile web services. Although the components for message security such as WS-Security have been standardized, standard architecture and service scenarios for providing message security for mobile web services have yet to be defined. Since simple object access protocol (SOAP) messages use hypertext transfer protocol (HTTP) ports, they cannot be filtered by firewalls; hence the need to provide a message filtering mechanism based on the message contents in the architecture for secure mobile web services as well as to integrate the security policy mechanism suitable for mobile web services message security and the message filtering mechanism into the architecture. Since many mobile terminals do not have sufficient processing power to support the web services protocol stack fully, and many back-end application servers are not based on web services, interworking mechanisms and scenarios between mobile web services and legacy non-web services applications should be provided.

This Recommendation seeks to establish a guideline for security architecture and security service scenarios for message security in mobile web services satisfying the above-mentioned requirements.

Source

ITU-T Recommendation X.1143 was approved on 13 November 2007 by ITU-T Study Group 17 (2005-2008) under the ITU-T Recommendation A.8 procedure.

Keywords

Message security, mobile web services, security gateway, security policy, web services security.

CONTENTS

Page

1 Scope ... 1

2 References ... 1

3 Definitions ... 2

3.1 Terms defined elsewhere ... 2

3.2 Terms defined in this Recommendation ... 3

4 Abbreviations and acronyms ... 4

5 Conventions ... 5

6 Introduction of mobile web services security-related standards ... 5

7 Requirements of security architecture and scenarios for message security in mobile web services ... 7

8 Security model for mobile web services ... 8

8.1 Security threats to mobile web services ... 8

8.2 Security requirements ... 9

8.3 Security functions for mobile web services ... 10

8.4 Security architecture for mobile web services ... 12

8.5 Message security service scenario for mobile web services ... 14

8.6 Message filtering ... 17

9 Use cases for message security in mobile web services ... 18

Appendix I – Reference security architecture for message security in mobile web services ... 20

I.1 Overall security architecture for message security in mobile web services ... 20

I.2 Components of the mobile web services security gateway ... 23

Appendix II – Use cases for message security in mobile web services ... 25

II.1 Use case 1 ... 25

II.2 Use case 2 ... 28

II.3 Use case 3 ... 30

II.4 Use case 4 ... 33

Appendix III – Examples of security policy description ... 36

Appendix IV – Comparison with other standards ... 38

IV.1 Parlay X ... 38

IV.2 AAA ... 39

IV.3 OMA ... 39

Bibliography ... 41

ITU-T Recommendation X.1143

Security architecture for message security in mobile web services

1 Scope

The scope of this Recommendation deals with the security architecture and security service scenarios for secure mobile web services as described below:

– Integrated security architecture for message security in mobile web services consisting of various mobile terminals and networks.

– Interworking mechanisms and service scenarios between applications that support the full web services security protocol stacks and legacy applications that do not support the full web services security protocol stack.

– Authentication, integrity and confidentiality mechanisms of the message in the mobile web services environment.

– Integrated security architecture that utilizes the security policy for message security in the mobile web services environment.

– Message filtering mechanism based on the message contents for the message security architecture.

– Reference message security architecture and security service scenarios for mobile web services.

The following objectives are not within the scope of this Recommendation:

– To define a new security policy language or an access control language.

– To define a new transport level security protocol or a message level security protocol.

2 References

The following ITU-T Recommendations and other references contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All Recommendations and other references are subject to revision; users of this Recommendation are therefore encouraged to investigate the possibility of applying the most recent edition of the Recommendations and other references listed below. A list of the currently valid ITU-T Recommendations is regularly published. The reference to a document within this Recommendation does not give it, as a stand-alone document, the status of a Recommendation.

[ITU-T X.800]	ITU-T Recommendation X.800 (1991), *Security architecture for Open Systems Interconnection for CCITT applications.*
[ITU-T X.805]	ITU-T Recommendation X.805 (2003), *Security architecture for systems providing end-to-end communications.*
[ITU-T X.1141]	ITU-T Recommendation X.1141 (2006), *Security Assertion Markup Language (SAML 2.0).*
[ITU-T X.1142]	ITU-T Recommendation X.1142 (2006), *eXtensible Access Control Markup Language (XACML 2.0).*
[IETF RFC 2246]	IETF RFC 2246 (1999), *The TLS Protocol Version 1.0*. <http://www.ietf.org/rfc/rfc2246.txt>
[IETF RFC 2828]	IETF RFC 2828 (2000), *Internet Security Glossary*. <http://www.ietf.org/rfc/rfc2828.txt>

[IETF RFC 3075]	IETF RFC 3075 (2001), *XML-Signature Syntax and Processing*. <http://www.ietf.org/rfc/rfc3075.txt>
[IETF RFC 3198]	IETF RFC 3198 (2001), *Terminology for Policy-Based Management*. <http://www.ietf.org/rfc/rfc3198.txt>
[OASIS WSS]	OASIS Standard (2006), *Web Services Security: SOAP Message Security 1.1 (WS-Security 2004)*. <http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf>
[OASIS WSS-SAML]	OASIS Standard (2006), *Web Services Security: SAML Token Profile 1.1*. <http://www.oasis-open.org/committees/download.php/16768/wss-v1.1-spec-os-SAMLTokenProfile.pdf>
[OASIS WSS-UsernameToken]	OASIS Standard (2006), *Web Services Security: UsernameToken Profile 1.1*. <http://www.oasis-open.org/committees/download.php/16782/wss-v1.1-spec-os-UsernameTokenProfile.pdf>
[OASIS WSS-X.509]	OASIS Standard (2006), *Web Services Security: X.509 Certificate Token Profile 1.1*. (Including errata) <http://www.oasis-open.org/committees/download.php/16785/wss-v1.1-spec-os-x509TokenProfile.pdf>
[OASIS WS-SecPol]	OASIS Standard, *WS-SecurityPolicy 1.2* (2007). <http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200512/ws-securitypolicy-1.2-spec-cd-01.pdf>
[W3C P3P]	W3C Recommendation (2002), *The Platform for Privacy Preferences 1.0 (P3P1.0) Specification*. <http://www.w3.org/TR/P3P/>
[W3C WSDL]	W3C Recommendation (2007), *Web Services Description Language (WSDL) Version 2.0 Part 0: Primer*. <http://www.w3.org/TR/wsdl20-primer/>
[W3C XML-Enc]	W3C Recommendation (2002), *XML Encryption Syntax and Processing*. <http://www.w3.org/TR/xmlenc-core/>

3 Definitions

3.1 Terms defined elsewhere

This Recommendation uses the following terms defined elsewhere:

3.1.1 access control: [ITU-T X.800]

3.1.2 authentication: [ITU-T X.800]

3.1.3 authorization: [ITU-T X.800]

3.1.4 availability: [ITU-T X.800]

3.1.5 confidentiality: [ITU-T X.800]

3.1.6 data integrity: [ITU-T X.800]

3.1.7 data origin authentication: [ITU-T X.800]

3.1.8 key: [ITU-T X.800]

3.1.9 key management: [ITU-T X.800]

3.1.10 non-repudiation: [ITU-T X.800]

3.1.11 policy decision point: [IETF RFC 3198]

3.1.12 policy enforcement point: [IETF RFC 3198]

3.1.13 privacy: [ITU-T X.800]

3.1.14 security architecture: [IETF RFC 2828]

3.1.15 security policy: [IETF RFC 2828]

3.1.16 security service: [IETF RFC 2828]

3.1.17 trust: [IETF RFC 2828]

3.2 Terms defined in this Recommendation

This Recommendation defines the following terms:

3.2.1 application service: This is an entity that provides various value-added services to the service requesters.

3.2.2 artifact: A piece of digital information. An artifact may be any size, and may be composed of other artifacts. Examples of artifacts: a message; a URI; an XML document; a PNG image; a bit stream. [b-W3C Glossary]

3.2.3 discovery: The act of locating a machine-processable description of a web service-related resource that may have been previously unknown and that meets certain functional criteria. It involves matching a set of functional and other criteria with a set of resource descriptions. The goal is to find an appropriate web service-related resource. [b-W3C Glossary]

3.2.4 discovery service: This is an entity that stores interface information for application services and related security policies for access to the application services by clients. Clients can get such information from the discovery service by sending queries that describe the services they want to access.

3.2.5 end point: An association between a binding and a network address, specified by a URI, that may be used to communicate with an instance of a service. An end point indicates a specific location for accessing a service using a specific protocol and data format. [b-W3C Glossary]

3.2.6 gateway: This is a network node that terminates a message on an inbound interface with the intent of applying some security processing or message conversions to the message, and presenting it to the target service through an outbound interface.

3.2.7 identifier: An identifier is an unambiguous name for a resource. [b-W3C Glossary]

3.2.8 policy server: This is an entity that manages security policies related to the security processing of messages and access control policies for the messages.

3.2.9 principal: This is an entity whose identity can be authenticated. Examples of principals include an end user or an organization.

3.2.10 registry: Authoritative, centrally controlled store of information. [b-W3C Glossary]

3.2.11 service description: A service description is a set of documents that describe the interface to and semantics of a service. [b-W3C Glossary]

3.2.12 service interface: A service interface is the abstract boundary that a service exposes. It defines the types of messages and the message exchange patterns that are involved in interacting with the service, together with any conditions implied by those messages. [b-W3C Glossary]

3.2.13 service provider: The person or organization that is providing a web service. [b-W3C Glossary]

3.2.14 service requester: A software agent that wishes to interact with a provider agent in order to request that a task be performed on behalf of its owner, the requester entity. [b-W3C Glossary]

3.2.15 SOAP intermediary: A SOAP intermediary is both a SOAP receiver and a SOAP sender and is targetable from within a SOAP message. It processes the SOAP header blocks targeted at it and acts to forward a SOAP message towards an ultimate SOAP receiver. [b-W3C Glossary]

3.2.16 SOAP message: The basic unit of communication between SOAP nodes. [b-W3C Glossary]

3.2.17 web service: A web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the web service in a manner prescribed by its description using SOAP-messages, typically conveyed using HTTP with an XML serialization in conjunction with other web-related standards. [b-W3C Glossary]

3.2.18 XML schema: The possible arrangement of tags and text in a valid representation of information. A schema might also be viewed as an agreement on a common vocabulary for a particular application that involves exchanging information. [b-Walsh]

4 Abbreviations and acronyms

This Recommendation uses the following abbreviations and acronyms:

AAA	Authentication, Authorization and Accounting
API	Application Programming Interface
ASP	Application Service Provider
CA	Certification Authority
HTTP	Hypertext Transfer Protocol
P3P	Platform for Privacy Preferences
PDP	Policy Decision Point
PEP	Policy Enforcement Point
PKI	Public-Key Infrastructure
SAML	Security Assertion Markup Language
SOAP	Simple Object Access Protocol
TLS	Transport Layer Security
TTP	Trusted Third Party
UDDI	Universal Discovery, Description, Integration
WSDL	Web Services Description Language
WS	Web Services
WTLS	Wireless Transport Layer Security
XACML	eXtensible Access Control Markup Language
XML	eXtensible Markup Language
XKMS	XML Key Management Specification

5 Conventions

None.

6 Introduction of mobile web services security-related standards

Web services is a set of protocols based on eXtensible Markup Language (XML). Figure 1 illustrates the base web services protocols.

– Simple object access protocol (SOAP): Defines the message format in XML containing the service request and response; SOAP is independent of any particular transport and implementation technology.

– Web services description language (WSDL): Describes a web service; it provides a programmatic way of describing what a service does, thereby paving the way for automation.

– Universal discovery, description, integration (UDDI): A cross-industry initiative to create a standard for service discovery together with a registry facility that facilitates the publishing and discovery processes.

Core web services technologies such as SOAP, WSDL and UDDI do not directly provide the security mechanism. As such, a comprehensive security model for web services has been developed by the industry. The web services security model introduces a collection of individual inter-related specifications describing an approach to layering security facilities into a web service environment. The architecture is designed to allow the mixing and matching of the specifications, thereby enabling implementers to deploy only the specific parts they need.

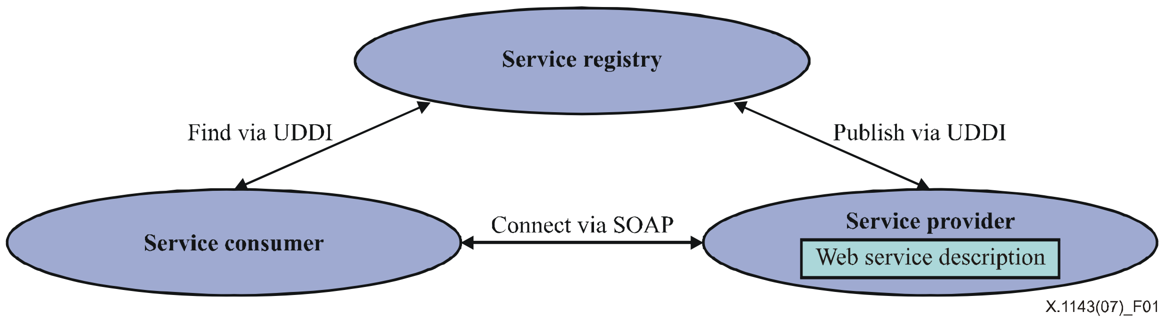


Figure 1 – Base web services protocols

The web services security roadmap that has been developed by the industry consists of an entire suite of specifications covering various facets of security (messaging, policies, trust, privacy, etc.). Figure 2 illustrates the roadmap.

The specifications build upon one another and are all built on top of a single specification, which is WS-Security (SOAP messages security). WS-Security defines a message security model.

The specifications are summarized as follows:

– WS-Security: Describes how to attach signatures and encryption headers to SOAP messages as well as how to attach security tokens, including binary security tokens such as X.509 certificates and Kerberos tickets, to messages.

– WS-Policy: Describes the capabilities and constraints of the security and other business policies on intermediaries and endpoints (e.g., required security tokens, supported encryption algorithms, privacy rules).

– WS-Trust: Describes a framework for trust models that enables web services to interoperate securely.

– WS-Privacy: Describes a model for how web services and service requesters state their privacy preferences and organizational privacy practice statements.
– WS-SecureConversation: Describes how to manage and authenticate message exchanges between parties, including security context exchange and establishment and derivation of session keys.
– WS-Federation: Describes how to manage and broker the trust relationships in a heterogeneous federated environment, including support for federated identities.
– WS-Authorization: Describes how to manage authorization data and authorization policies.

WS-Security (SOAP messages security) is a base protocol in the web services security architecture; it is now an OASIS standard. WS-Security describes extensions to the SOAP protocol to provide secure messaging particularly to ensure message integrity and message confidentiality.

To accomplish this, the specification describes how to attach signature and encryption headers as well as binary-encoded security tokens such as X.509 certificates and Kerberos tickets to SOAP messages.

Message integrity is established through a combination of XML signatures and security tokens. By pairing digital signatures with security tokens that either contain or imply key data, the recipient of a message can be assured that the message has been transmitted without modification by a trusted party. Digital signatures support multiple signatures from multiple actors on the same document as well as multiple signature formats.

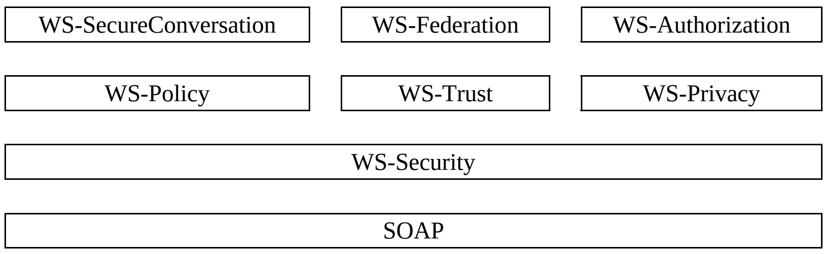


Figure 2 – Web services security roadmap

The security token mechanism defined by WS-Security is sufficiently extensible to support multiple security token formats (username-password pairs, X.509 certificates, etc.) and allow for the explicit inclusion of a token or an assertion regarding a security token that exists elsewhere.

Message confidentiality is established through a combination of XML encryption and security tokens. Similar to the case with digital signatures, the encryption mechanisms defined in WS-Security are designed to support a wide variety of encryption technologies, processes and operations by multiple actors. An encrypted element may also reference a security token.

The WS-Security specification also defines a mechanism for encoding binary security tokens such as X.509 certificates, Kerberos tickets and opaque encrypted keys, and for transmitting them with a SOAP message.

Figure 3 illustrates the structure of a message secured by WS-Security.

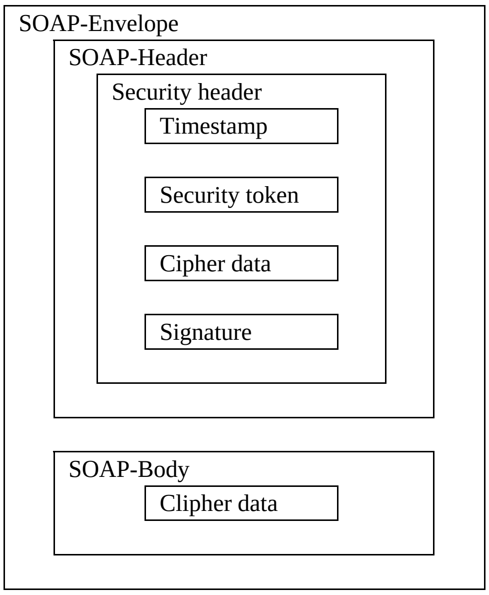


Figure 3 – Structure of a message secured by WS-Security

The XML signature for part of the SOAP message is generated and inserted into the security header element. Part of the SOAP message is encrypted using XML encryption, and its header information is inserted into the security header element; the encrypted part is then replaced by its cipher data. Security token(s) related to the digital signature or encryption is/are added to the security header element. A timestamp element used to determine the freshness of the message may be inserted into the security header element. The security header element is then inserted into the SOAP header part of the SOAP message.

WS-Security is a standard set of SOAP extensions that can be used when building secure web services for message-level integrity and confidentiality. It is a building block that can be used in conjunction with other web services extensions and higher-level, application-specific protocols. In the web services scenarios, SOAP messages may be exchanged via intermediary nodes and WS-Security can provide end-to-end security to the messages. The intermediary nodes may perform additional security processing on the SOAP messages if necessary.

The mobile industry is trying to apply web services technologies to the mobile domain since they can simplify the integration problems between operators, service providers and content providers. The convergence of mobile and web services technologies enables new services and business models and accelerates the development of mobile and fixed Internet technologies.

The mobile web services working group of Open Mobile Alliance (OMA) has developed a specification suite that will aid developers in applying web services to the mobile domain. A summary of the OMA specifications and the differences between the OMA specifications and this Recommendation are described in clause IV.3.

7 Requirements of security architecture and scenarios for message security in mobile web services

Applying existing security technologies to mobile web services directly is difficult because they have different security requirements. This clause analyses the security requirements for mobile web services.

In web services scenarios, SOAP messages may be exchanged via intermediary nodes. When SOAP messages traverse between an originator and the final SOAP endpoint, it may be a requirement that information be kept secret from SOAP intermediaries. Transport level security protocols such as TLS cannot satisfy this requirement. WS-Security is adequate for such cases since it can provide end-to-end security to the messages. Although WS-Security and other web services security technologies have been standardized, applying the technologies to the mobile environment is difficult since most of the mobile terminals do not have sufficient processing power to support the web services protocol stack fully. Moreover, many back-end application servers are not based on web services; hence the need for a security service architecture that provides security interworking mechanisms between mobile web services and legacy non-web services.

Since mobile web services use HTTP ports, and they cannot be filtered by firewalls, it is required to provide message filtering mechanism based on message contents in the security service architecture. It is also necessary to integrate security policy mechanisms suitable for mobile web services and the message filtering mechanism into the architecture for secure message delivery in the mobile environment.

8 Security model for mobile web services

This clause describes the security model for mobile web services. This clause identifies security threats and security requirements, and then describes the security functions for mobile web services security. Then, the security architecture and message security service scenarios for mobile web services are described. Appendices I and II describe the detailed reference security architecture and service scenarios for message security in mobile web services, respectively, based on this model.

8.1 Security threats to mobile web services

8.1.1 General security threats

– Masquerade: In a masquerade, an entity pretends to be a different entity. An authorized entity with few privileges may use a masquerade to obtain extra privileges by impersonating an entity having such privileges.

– Eavesdropping: Eavesdropping involves viewing information that should not be viewed, either by examining messages in transit or by examining the content stored in a server.

– Replay: A replay occurs when a message or part of it is repeated to produce an unauthorized effect.

– Modification of messages: The modification of a message occurs when the content of a data transmission is altered undetected, thereby resulting in an unauthorized effect.

– Man in the middle attack: A man in the middle attack is an attack wherein an attacker is able to read and modify at will the messages between two parties without either party knowing that the link between them has been compromised.

– Denial of service (DoS): Denial of service occurs when an entity fails to perform its proper function or acts in a manner that prevents other entities from performing their proper functions. Examples include overwhelming the server with requests requiring excessive processing or consuming excessive resources.

8.1.2 Security threats specific to mobile web services

– XML denial of service (XML-DoS): Since web services uses port 80, its XML traffic cannot be filtered by traditional firewalls. Traditional firewalls cannot detect content-level vulnerabilities. An attacker can make use of malicious XML messages, manipulate parts of XML data, or send an oversized XML payload that can trigger load-intensive operations at the target web services. Two of the most important DoS attacks in web services are "coercive parsing" and "oversize payload". A coercive parsing attack uses a deeply nested

XML document. On the other hand, an oversize payload attack uses an extremely large XML document to use up the memory of the service.

– XML message injection and manipulation: An attacker can modify parts of the XML messages or attachments to cause endless loops or failure of an XML parser. The attacker can also make use of recursive elements or XPath expression or unrelated message attachments to perform unintended processing that leads to service failure. This attack usually comes after a man in the middle attack.
– Session hijacking and theft: Some web services providers use session identifiers during communication to identify the service requesters. An attacker can steal and use the identifier information to hijack a session between the web services provider and the consumer.
– Parameter tampering: WSDL provides information on how to use parameters to execute a specific remote operation at the target web services. It is usually opened to the public. An attacker can manipulate different parameter options in order to execute an unauthorized operation.

8.2 Security requirements

– Access control: This is required to ensure that only authorized users or devices are allowed to access appropriate system resources or services.
– Authentication: This is required to confirm the identities of the communicating entities. Authentication ensures the validity of the claimed identities of the entities participating in communication and provides assurance that an entity is not attempting a masquerade or an unauthorized replay of a previous communication. Authentication techniques may be required as part of access control.
– Non-repudiation: Non-reputation provides the means for preventing an individual or an entity from denying having performed a particular action related to data by making available proof of various network-related actions.
– Data confidentiality: This is required to protect data that is being transported, processed or stored by a network service against unauthorized access or viewing.
– Communication security: This ensures that information flows between authorized endpoints only. Communication security ensures that the information is neither diverted nor intercepted as it flows between these endpoints.
– Data integrity: Data integrity ensures the correctness or accuracy of data. Data is protected against unauthorized modification, deletion and replication, providing an indication of these unauthorized activities.
– Availability: This ensures that there is no denial of authorized access to network element, stored information flows, services, and applications due to events affecting the network.
– Privacy: Privacy refers to the right of individuals to control or influence which information related to them may be collected and stored and to whom such information may be disclosed.

Table 1 summarizes the relationship between the general security threats and security requirements.

Table 2 summarizes the relationship between the mobile web services-specific security threats and security requirements.

In each table, cells marked with "X" mean that a security requirement in the row is related to a threat in the column.

Table 1 – Relationship between the general security threats and security requirements

Threats / Security requirements	Masquerade	Eavesdropping	Replay	Modification of messages	Man in the middle attack	Denial of service
Access control	X			X		X
Authentication	X		X	X	X	X
Non-repudiation				X	X	
Data confidentiality	X	X	X	X	X	
Communication security				X		
Data integrity	X			X	X	
Availability						X
Privacy	X	X	X	X	X	

Table 2 – Relationship between the mobile web services-specific security threats and security requirements

Threats / Security requirements	XML-DoS	XML message injection and manipulation	Session hijacking and theft	Parameter tampering
Access control	X	X		X
Authentication	X	X	X	X
Non-repudiation		X	X	
Data confidentiality		X	X	
Communication security				
Data integrity		X	X	
Availability	X			
Privacy		X	X	

8.3 Security functions for mobile web services

Figure 4 briefly shows the security functions for mobile web services grouped according to protocol layers.

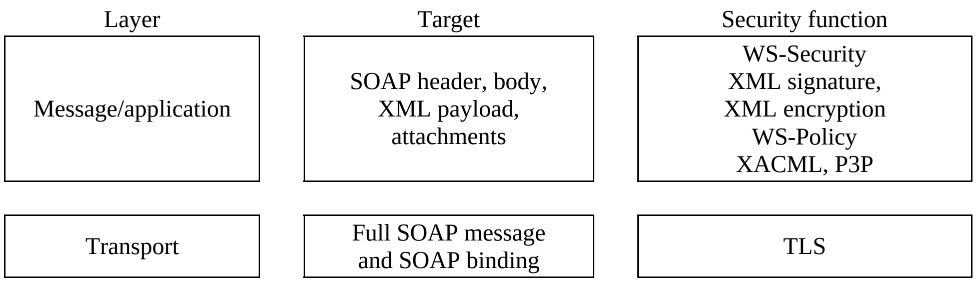


Figure 4 – Security functions for each protocol layer

For transport level security, TLS should be used. Through the transport security mechanism, the full SOAP message and SOAP binding are secured. TLS (see [IETF RFC 2246]) provides point-to-point security.

For message level security, WS-Security (see [OASIS WSS]) should be used. Through the message level security mechanism, parts of the SOAP message can be secured. It can also be used to secure the SOAP header, body and attachments, providing end-to-end security. Multiple intermediaries may exist between the original web services requester and the web services provider in mobile web services scenarios. As such, the message level security mechanism should be applied to such cases for end-to-end security.

The transport level security mechanism may be used to secure messages between two adjacent web services nodes. On the other hand, message level security mechanisms should be used in case multiple intermediaries exist between the service requester and the service provider.

XML signature and XML encryption should be used for message level security and application level security for protecting XML payloads and attachments. XML signature (see [IETF RFC 3075]) enables message authentication, integrity and non-repudiation. It supports signing of parts of XML messages, non-XML messages and multiple messages. XML encryption (see [W3C XML-Enc]) enables message confidentiality and supports the partial encryption of XML messages.

To specify the message security requirements or privacy requirements, WS-Policy may be used. WS-Policy defines the framework for allowing web services to express their constraints and requirements. Such constraints and requirements are expressed as policy assertions, with the WS-SecurityPolicy specification (see [OASIS WS-SecPol]) defining a set of security policy assertions for use with the WS-Policy framework with WS-Security.

XACML may be used for access control. The XACML standard (see [ITU-T X.1142]) describes both an XML-based policy language and an access control decision request/response language. Used to describe the general access control requirements, the policy language has standard extension points for defining new functions, data types and combining logic.

Platform for privacy preferences (P3P) (see [W3C P3P]) may be used for privacy. It makes statements on how personally identifiable information about an end-user is used by a server.

Although the fundamental technology for preventing DoS attacks has yet to be developed, packet filtering is helpful in terms of availability in the transport layer. Since web services uses port 80, its XML traffic cannot be filtered via traditional packet filtering. Thus, packet filtering cannot prevent XML-DoS attacks. Message filtering through content inspection and validation of incoming XML messages may be used for availability.

Table 3 summarizes the relationship between the mobile web services' security requirements and security functions. The transport level security functions and message/application level security functions related to specific security requirements are listed in the table below.

Table 3 – Relationship between security requirements and security functions

Security requirements	Transport level security function	Message/application level security function
Access control	TLS	XACML
Authentication	TLS	WS-Security, XML signature
Non-repudiation		WS-Security, XML signature
Data confidentiality	TLS	WS-Security, XML encryption
Communication security		
Data integrity	TLS	WS-Security, XML signature
Availability		
Privacy		WS-Policy, WS-Privacy, P3P

8.4 Security architecture for mobile web services

Figure 5 illustrates the security architecture for mobile web services.

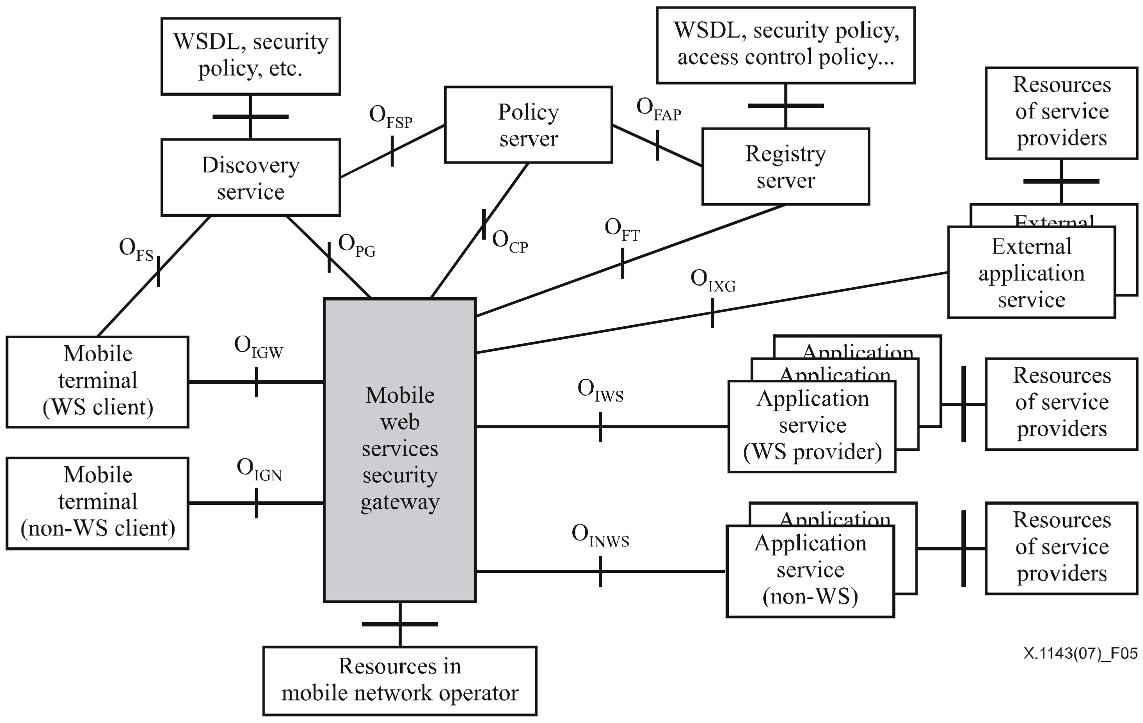

WSDL, security policy, etc.
WSDL, security policy, access control policy...
Resources of service providers
Discovery service
Policy server
Registry server
External
External application service
O_{FSP}
O_{FAP}
O_{FS}
O_{PG}
O_{CP}
O_{FT}
O_{IXG}
Mobile terminal (WS client)
Mobile terminal (non-WS client)
O_{IGW}
O_{IGN}
Mobile web services security gateway
O_{IWS}
O_{INWS}
Application
Application
Application service (WS provider)
Resources of service providers
Application
Application service (non-WS)
Resources of service providers
Resources in mobile network operator
X.1143(07)_F05

Figure 5 – Security architecture for mobile web services

It consists of the following components:

– Mobile terminal: A mobile terminal is a client of the mobile web services. It may or may not support the full web services protocol stack.

– Mobile web services security gateway (MWSSG): MWSSG is the core component in this model. All requests from mobile clients are sent to MWSSG, which provides a single access point to all application servers. It should also act as a policy enforcement point (PEP) to enforce security policy for access control.

– Policy server: The policy server manages security policies related to the security processing of the messages and access control policies for messages. The policy server should act as a policy decision point (PDP).

– Application service: The application service provides various value-added services to the clients. It may or may not be a web services provider. The application server may or may not be located in the internal domain of the mobile network operator.

– Discovery service: The discovery service stores the interface information for application services and related security policies for access to the application services by the clients. Clients can get such information from the discovery service by sending queries describing the services they want to access.

– Registry server: The registry server manages the interface information for application services, related security policies for access to the application services by the clients, and access control policies related to the request message to the target services. It should reside in the internal domain of the mobile operator. The registry server can be accessed by MWSSG, and it should deny access from clients in the external domain of the mobile operator.

  If the request from the client is in the form of a SOAP message, then MWSSG should validate the schema and check the known vulnerability of the SOAP message. Whether the SOAP message has been secured according to the given security policy should be verified as well. If the validation of the message is successful, then the message should be routed to the destination service. If the verification of the message fails, then the request should be rejected.

The following are the interfaces between components:

– O_{FS}: Interface between the discovery service and mobile terminals supporting the web services protocol stack; using this interface, mobile terminals can find and access service interface information described in WSDL (see [W3C WSDL]) and security policies to access the target web services.

– O_{FSP}: Interface between the discovery service and the policy server to find and access the security policies stored in the discovery service.

– O_{PG}: Interface between the discovery service and a mobile web services security gateway; MWSSG publishes service interface information which is necessary to access the target application services to the discovery service using this interface.

– O_{CP}: Interface between MWSSG and the policy server to check whether or not the requested messages have been secured properly according to the security policies and to make a policy decision on the request.

– O_{IGW}: Interface between a mobile terminal and MWSSG for sending or receiving a SOAP request or a response message; this is used in case the mobile terminal supports the web services protocol stack.

– O_{IGN}: Interface between a mobile terminal and MWSSG for sending or receiving a non-SOAP request or a response message; this is used in case the mobile terminal does not support the web services protocol stack.

– O_{IWS}: Interface between an application service and MWSSG for sending or receiving a SOAP request or a response message; this is used in case the application service supports the web services protocol stack.

– O_{INWS}: Interface between an application service and MWSSG for sending or receiving a non-SOAP request or a response message; this is used in case the application service does not support the web services protocol stack.

– O_{IXG}: Interface between MWSSG and an application service in the foreign domain for sending or receiving a request or a response message; the message may or may not be a SOAP message.

– O_{FT}: Interface between MWSSG and the registry server to find and access service interface information as well as the access control policies related to the request message sent by the client.

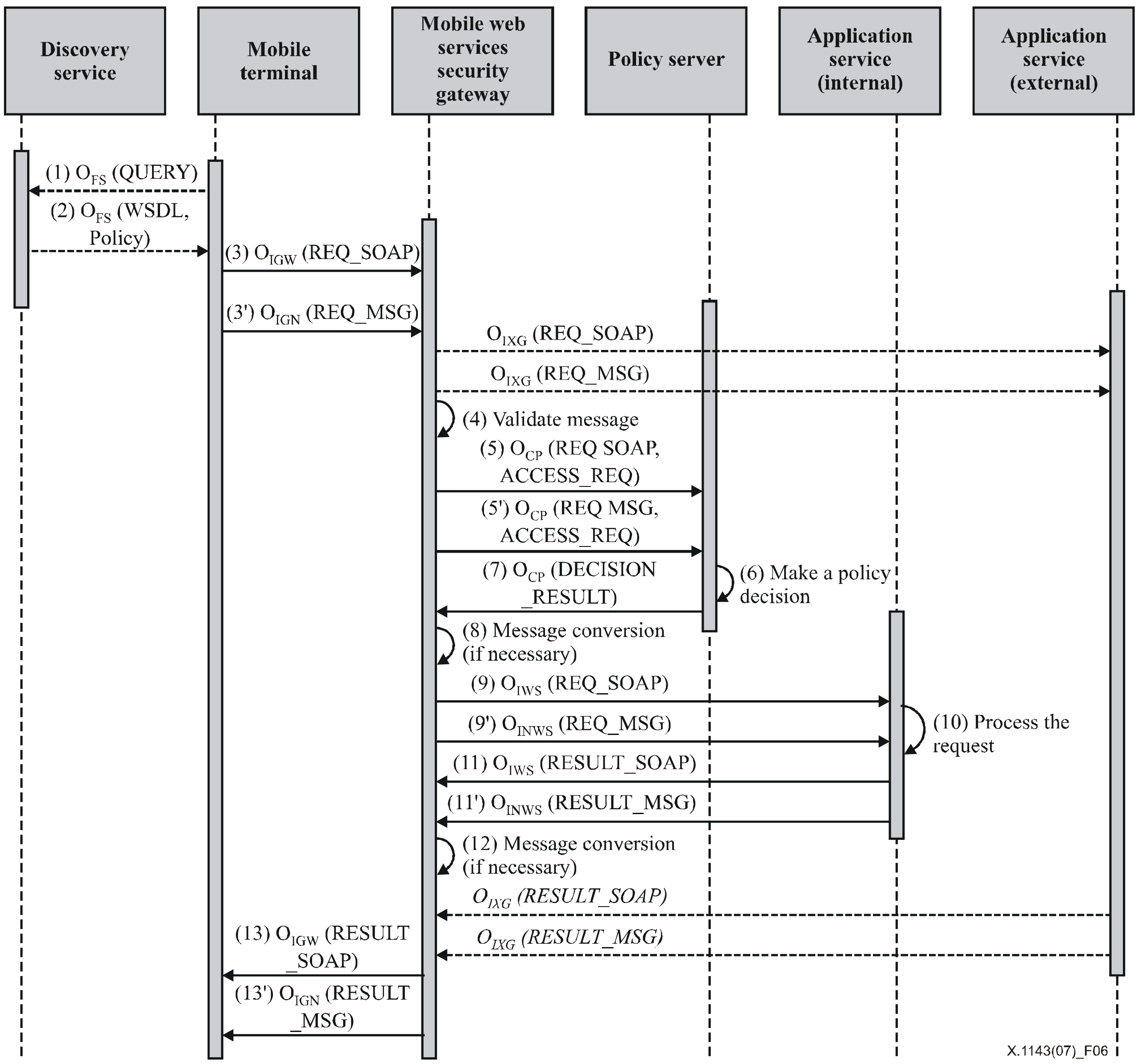
– O_{VM}: Internal interface between MWSSG and the message validator module which is a component of MWSSG; used to request the validation of the incoming message to the message validator module. The message validator module is used to validate SOAP messages. This internal interface is used to illustrate the message filtering procedure in Figure 7.

– O_{FAP}: Interface between the policy server and the registry server to find and access the access control policies related to the access request message sent by the client destined for the target application service.

The detailed reference security architecture for mobile web services is described in Appendix I.

8.5 Message security service scenario for mobile web services

Figure 6 illustrates the message security service scenario in the architecture above.



Discovery service
Mobile terminal
Mobile web services security gateway
Policy server
Application service (internal)
Application service (external)
(1) O_FS (QUERY)
(2) O_FS (WSDL, Policy)
(3) O_IGW (REQ_SOAP)
(3') O_IGN (REQ_MSG)
O_IXG (REQ_SOAP)
O_IXG (REQ_MSG)
(4) Validate message
(5) O_CP (REQ SOAP, ACCESS_REQ)
(5') O_CP (REQ MSG, ACCESS_REQ)
(7) O_CP (DECISION _RESULT)
(6) Make a policy decision
(8) Message conversion (if necessary)
(9) O_IWS (REQ_SOAP)
(9') O_INWS (REQ_MSG)
(10) Process the request
(11) O_IWS (RESULT_SOAP)
(11') O_INWS (RESULT_MSG)
(12) Message conversion (if necessary)
O_IXG (RESULT_SOAP)
O_IXG (RESULT_MSG)
(13) O_IGW (RESULT _SOAP)
(13') O_IGN (RESULT _MSG)
X.1143(07)_F06

Figure 6 – Message security service scenario for mobile web services

1) If it supports the web services protocol, the mobile client sends queries to find and access the service interface information of the target service that is usually described in WSDL using O_{FS}. The additional description of the QUERY parameter in Figure 6 is described in clause I.1.

2) The mobile client finds and accesses service interface information published by MWSSG such as interface information described in WSDL and security policies necessary to access the target service from the discovery service using O_{FS}. Such discovery operation is assumed to have been performed already prior to the sending of a request message, with the client knowing the necessary service interface information and security policies. This operation may be performed using the mobile Internet service provided by the mobile operator, or such information may be downloaded using other communication methods. If the mobile client does not support the web services protocol, the mobile client is assumed to have acquired the service interface information and security policies already that are needed to access the target service before sending the request.

3) If it supports the web services protocol and WS-Security, then the mobile client should send to MWSSG a request SOAP message that has been secured by applying WS-Security

according to the security policies to MWSSG using O_{IGW}. If it does not support the web services protocol and WS-Security, however, then the client should send to MWSSG a non-SOAP request message that has been secured by applying TLS according to the security policies to MWSSG using O_{IGN}. If the target service is located in the external domain, then the request should be routed to the destination domain by MWSSG using O_{IXG}.

4) MWSSG should validate the request message by validating the schema and checking the known vulnerabilities of the SOAP message. If message conversion is necessary to access the target service (i.e., from a SOAP message to a non-SOAP message or from a non-SOAP message to a SOAP message), then the request message should be decrypted at MWSSG prior to validation. In this case, MWSSG should have the necessary information to decrypt the message and validate the digital signature of it. If the message targets an application service in the external domain of the mobile operator, then decryption should not be performed at the internal MWSSG and should be sent to the destination domain since the internal MWSSG is not allowed to view the contents of the message.

5) MWSSG should request the checking of the request message to the policy server using $O_{CP.}$

6) The policy server should check whether the message conforms to the security policies. Afterwards, related message access control policies to the target service should be checked, and the policy decision on access to the target service should be made by the policy server. For the policy decision, attributes from security token(s) such as X.509 token, Username token, or SAML token in the request SOAP message may be checked and used. Such security tokens convey security information related to authentication and authorization. An X.509 token (see [OASIS WSS-X.509]) and a Username Token (see [OASIS WSS-UsernameToken]) support means to provide X.509 certificates and username/password between a web service provider and a requester, respectively. A SAML token (see [OASIS WSS-SAML]) enables attachment of SAML assertions (see [ITU-T X.1141]) into the SOAP message, thus security assertions can be exchanged with SOAP messages using the SAML token. Security attributes from such security tokens may be used for authentication and authorization of the request message. An X.509 token may also be used to verity the digital signature in other steps in the procedure.

7) MWSSG receives the policy decision from the policy server. If the request message does not conform to the given security policies, or in case access to the target service is denied by the access control policies, then an error message should be returned to the mobile terminal.

8) If the request message is a SOAP message, and the target service is not based on web services, then the SOAP message should be converted into a non-SOAP message that can be understood by the target service. If the message is not a SOAP message, and the target service expects a SOAP message, then MWSSG should convert the message into a SOAP message. MWSSG should secure it properly if the request will be sent to the external domain or if the communication channel between the target service and MWSSG is not secure enough, or in case they do not have trust relationship.

9) The processed request message should be sent to the target application service using O_{IWS} or O_{INWS}.

10) The target service decrypts and verifies the request message and processes it. If the target service and MWSSG have a trust relationship, and the communication channel between them is secure, then the target service can delegate the security processing to MWSSG. In this case, security processing in steps 8 and 10 may be skipped.

11) The response from the target service is secured and sent back to MWSSG. If the target service and MWSSG have a trust relationship, and the communication channel between them is secure, then security processing between them may be skipped. If the response

message is from the external domain, then it should have been in secured status. That is, the security processing such as digital signature and encryption has been performed to the message. In such case, MWSSG in the local domain should decrypt and validate the signature of the message.

12) Message conversion of the response message is performed in reverse order as described in step 8 if necessary. If such message conversion is required, then MWSSG should decrypt the response message prior to the conversion.

13) The response message is secured and sent back to the mobile client.

The detailed message filtering mechanism in steps 4-7 in Figure 6 is described in clause 8.6.

The detailed reference security service scenarios for message security in mobile web services are described in Appendix II.

8.6 Message filtering

Figure 7 illustrates the filtering process by MWSSG and the policy server. The filtering of messages is performed as follows:

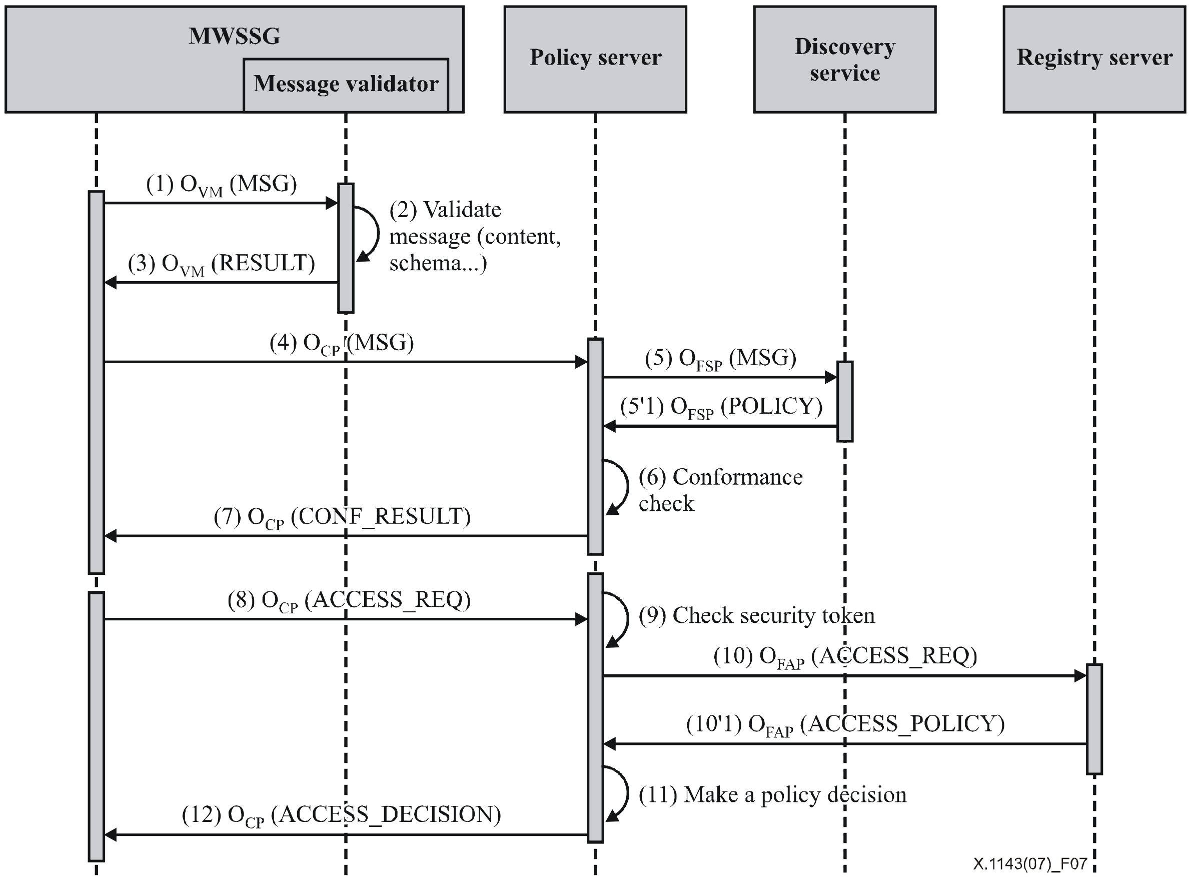


Figure 7 – Message filtering procedure

1) Upon receiving a request message, MWSSG should perform message validation using the message validator.

2) The message validator should inspect the contents of the message and check for any harmful data in it. If the incoming message is a SOAP message, then the XML schema should be validated and the known vulnerabilities of SOAP messages should be checked.

3) If validation fails, then the message should be rejected.

4) MWSSG requests conformance check to the policy server. This step is necessary to make sure that the request message has been secured according to the security policies published in the discovery service.

5) The policy server accesses the security policies related to the request message from the discovery service. Security policies published by MWSSG to the discovery service may also be stored in the registry server and may be used in this step instead of accessing the discovery service for better performance.

6) Whether the message has been secured according to the specified security policies should be checked by the policy server.

7) The conformance check result is returned to MWSSG. If the message is non-conforming to the policies, then the request should be rejected.

8) MWSSG should request a policy decision on access control to the request message to the policy server. This step is necessary to check whether the request message has the right to call the specific internal service or the method of the web service or resources.

9) For the policy decision, attributes from the security token(s) such as X.509 token, Username token or SAML token in the request SOAP message may be checked and used. If the request message is not in SOAP, then equivalent information such as X.509 certificate or id/password or SAML artefact related to the request message may be used.

10) The policy server accesses the access control policies related to the request message from the registry server.

11) The policy server should make an access control decision based on these polices.

12) If the decision is to deny access, then the request message should be rejected by the MWSSG.

Examples of security policies are shown in Appendix III.

9 Use cases for message security in mobile web services

This clause explains the use cases for message security in mobile web services. Use cases are classified into four categories. Table 4 summarizes the use cases.

Table 4 – Summary of use cases

1	Both the mobile client and the application server support the web services protocol stack (Use case 1)	The target web service is located in the internal domain of a mobile network operator (Use case 1-1)
		The target web service is located in the external domain of a mobile network operator (Use case 1-2)
2	The mobile client does not fully support the web services protocol stack, and the target service supports the web services protocol stack (Use case 2)	The target web service is located in the internal domain of a mobile network operator (Use case 2-1)
		The target web service is located in the external domain of a mobile network operator (Use case 2-2)
3	The mobile client supports the web services protocol stack, and the target service does not support the web services protocol stack (Use case 3)	The target web service is located in the internal domain of a mobile network operator (Use case 3-1)
		The target web service is located in the external domain of a mobile network operator (Use case 3-2)
4	Both the mobile client and the target service do not support the web services protocol stack (Use case 4)	The target web service is located in the internal domain of a mobile network operator (Use case 4-1)
		The target web service is located in the external domain of a mobile network operator (Use case 4-2)

These use cases are described in Appendix II.

Appendix I

Reference security architecture for message security in mobile web services

(This appendix does not form an integral part of this Recommendation)

This appendix describes the detailed reference security architecture for message security in mobile web services consisting of various mobile terminals and networks based on the security model defined in clause 8.

Use cases using this reference architecture including detailed processing flows are explained in Appendix II.

I.1 Overall security architecture for message security in mobile web services

Figure I.1 illustrates the reference security architecture for message security in mobile web services. It consists of the following components:

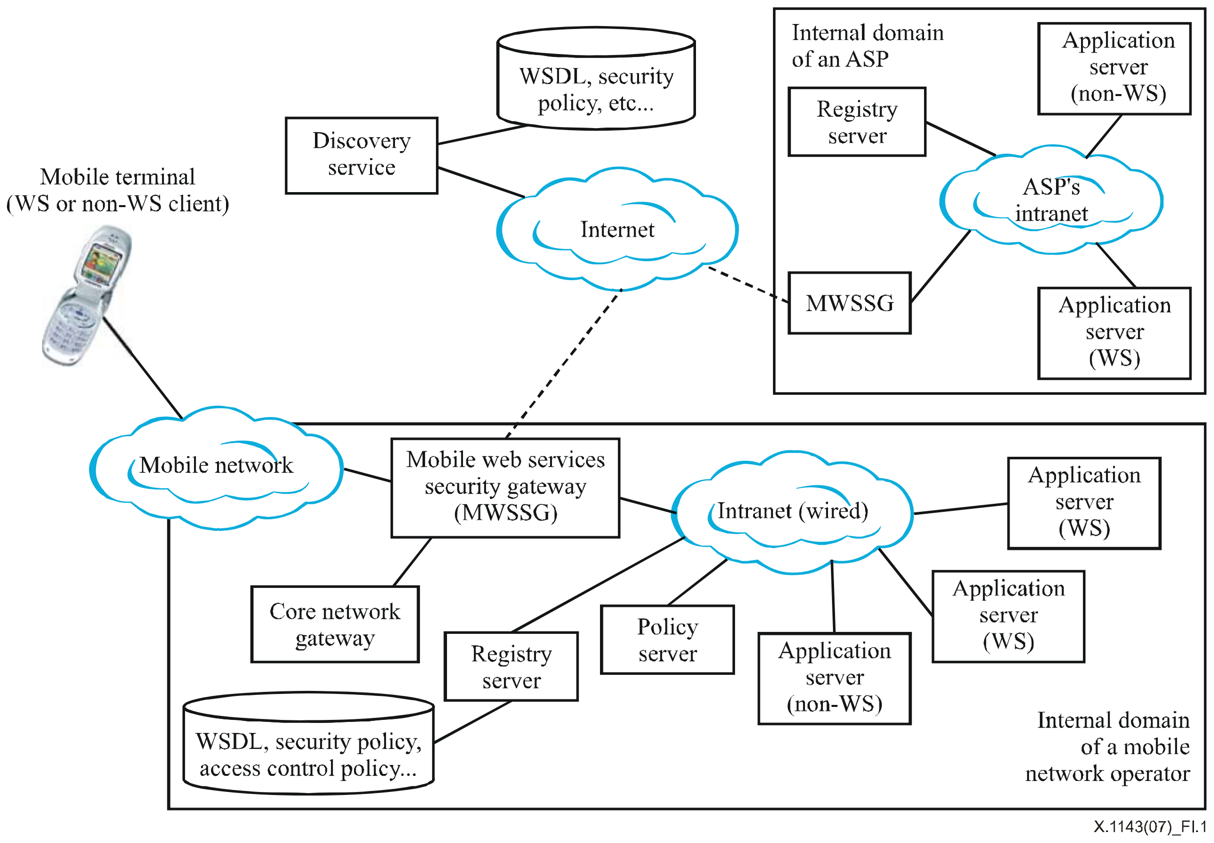

WSDL, security policy, etc...
Internal domain of an ASP
Application server (non-WS)
Registry server
Discovery service
Mobile terminal (WS or non-WS client)
Internet
ASP's intranet
MWSSG
Application server (WS)
Mobile network
Mobile web services security gateway (MWSSG)
Intranet (wired)
Application server (WS)
Core network gateway
Application server (WS)
Policy server
Registry server
Application server (non-WS)
WSDL, security policy, access control policy...
Internal domain of a mobile network operator
X.1143(07)_FI.1

Figure I.1 – Reference security architecture for message security in mobile web services

– Mobile terminal: A mobile terminal is a client of the mobile web services. It may or may not support the full web services protocol stack. For this proposed architecture, however, the mobile terminal is required to support TCP/IP and TLS (or WTLS).

– The policy server manages security policies related to the security processing of the messages and access control policies for them. These policies are used for message filtering. To perform message filtering using policies, the conformance check of the message to the security policies, and a policy decision on the access control of the request message, are performed by the policy server.

– Discovery service: The discovery service stores the interface information for application services and related security policies for access to the application services by the clients. Clients can get such information from the discovery service by sending queries describing the services they want to access.

  The discovery service may use the UDDI registry. The UDDI standard provides a standardized method of publishing and discovering information about web services. Clients use the UDDI programmers' API to publish services and query the registry to discover services that match various criteria. The XML message below is an example of a query to the discovery service for finding information on a service provider named XXY:

```
<uddi:find_business generic="2.0" maxRows="10">
    <uddi:name>
        XXY
    </uddi:name>
</uddi:find_business>
```

  The result of the query contains <businessInfo> structures that have information on service provider.

– Registry server: The registry server manages the interface information for application services, related security policies for access to the application services by the clients, and access control policies related to the request message to the target services. It should reside in the internal domain of the mobile operator. The registry server can be accessed by MWSSG, and it should deny access from the clients in the external domain of the mobile operator.

– Mobile web services security gateway (MWSSG): MWSSG is the core component in this reference architecture. All requests from mobile clients are sent to MWSSG, which provides a single access point to all application servers. It should also act as a policy enforcement point (PEP) to enforce security policy for access control.

  If the request is in the form of a SOAP message (i.e., the client is web services-enabled), then it should validate the message by validating the schema and checking the known vulnerabilities of the SOAP message. Whether the SOAP message has been secured according to the specified security policies should also be checked. If the validation of the message is successful, then the message is routed to the destination service by referencing the actual endpoint information from the registry server. If the validation of the message fails, then the request is rejected. This feature is helpful in reducing the load of the entire system since the message is rejected if it is not a legal message before it is delivered and processed by the target services.

  If the incoming request is not a SOAP message and is encrypted using TLS, the MWSSG decrypts it and validates the contents to check whether it is a valid message or not. MWSSG also plays the role of a message converter. In other words, if the request message is a SOAP message, and the target service is not based on web services, then the SOAP message is converted into a non-SOAP message that can be understood by the target service. In this case, application-specific knowledge is required for message conversion. The converted message may be secured again using TLS. If the incoming message is a non-SOAP message (i.e., the client is not web services-enabled), and the target service expects a SOAP message, then MWSSG converts the message into a SOAP message and secures it properly if necessary using the interface information described in WSDL and security policies from the registry server.

  If both the mobile client and the target web service provide the full web services protocol stack, then the request and response SOAP messages are secured using WS-Security and may be exchanged without decryption by intermediaries.

– An application server provides services to mobile clients; it may or may not support the full web services protocol stack. The application server publishes its service interface and endpoint information to the registry server.

– Only the interface exposed by MWSSG is published into the discovery service. The subset of the service interfaces required for the client to invoke the web service via MWSSG is published by the MWSSG to the discovery service. In this case, the endpoint of the target web service is replaced by the endpoint of the MWSSG. As a result, the client only knows the endpoint of the MWSSG.

– The core network gateway provides access to the network elements of the network operator. An example of the core network gateway is the Parlay/OSA gateway used to link applications using the Parlay/OSA APIs with the existing network elements. The Parlay/OSA gateway is under the control of the network operator or service provider, and is a single point through which all Parlay/OSA interactions pass. MWSSG may use the core network gateway to use the functions provided by the network elements required for security management. The Parlay/OSA gateway may be accessed using the Parlay X web services. Since the Parlay X web services are web services-based, these services can also be protected by MWSSG.

  Trusted third party (TTP) such as CA or an XML key management specification (XKMS) server may be used to provide certificate issuance services. The online certificate status protocol (OCSP) server or XKMS server may be used for certificate validation services.

  Using the service interface information described in WSDL and security policies from the discovery service, the clients send a request message to MWSSG. The request is then verified and routed to the actual target web service by referencing the service interface information from the registry server.

The proposed architecture covers the legacy network components and services that do not support the full web services protocol stack. It utilizes the security policy and key management for message security in mobile web services environments. Illegal request messages are filtered out before they are transmitted to the application server, thereby helping reduce the load on the entire system. This architecture provides a single access point to all application servers and hides the deployment structure of the servers from clients; thus it allows changes of deployment structure of the servers without changing the clients. It also improves security, since access control and other security processing can be done at the single access point.

I.2 Components of the mobile web services security gateway

Figure I.2 illustrates the components of MWSSG.

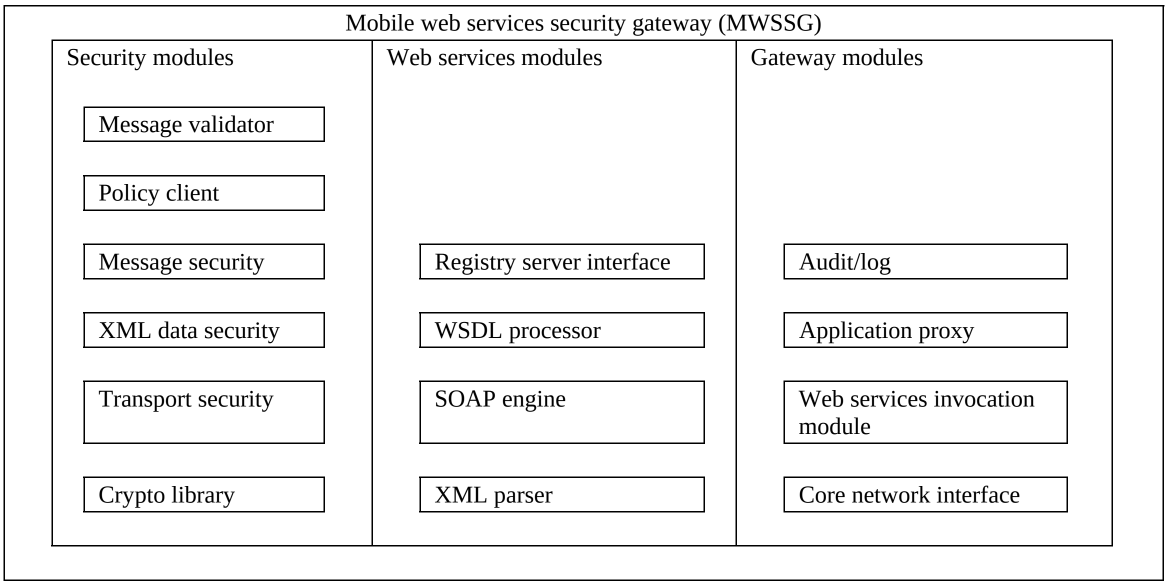


Figure I.2 – Components of MWSSG

I.2.1 Security modules

These modules provide security functionality to MWSSG.

– The crypto library provides encryption, digital signature and certificate processing functions necessary for message security.

– The transport security module provides TLS for transport level security.

– The XML data security module provides the XML digital signature function and XML encryption function to secure XML payloads.

– The message security module implements the WS-Security specification for SOAP message security.

– The policy client requests the conformance check of the message to the security policy or requests a policy decision on access control of the request message to the policy server.

– The message validator module is used to validate SOAP messages. Validation includes validating the schema and checking the known vulnerabilities of SOAP messages and security policy conformance.

I.2.2 Web services modules

These modules provide web services-related functionality to MWSSG.

– The XML parser supports the parsing and manipulation of XML documents.

– The SOAP engine processes the web services request and response and other web services-related processing.

– The WSDL processor retrieves the service interface information described in WSDL and generates the client code from it to access the target service.

– The registry server interface is used to access the registry server. The service interface information of the target services (usually described in WSDL), security policies and access control polices related to the services, and other interface information, are retrieved from the registry server.

I.2.3 Gateway modules

These modules provide functionality related to application proxy and message validation to MWSSG.

– The core network interface is used to access the network elements in the network operator through the core network gateway. This interface is usually used for monitoring the core networks. The core network interfaces may be implemented using Parlay X APIs. Core network elements can be accessed through the Parlay gateway, and Parlay X gives application developers access to the Parlay gateways using web services.

– The web services invocation module is used for invoking services that are not based on the web services from the mobile clients that send request messages in SOAP.

– The application proxy is used for invoking web services from legacy clients that do not support the web services. It converts the non-SOAP request message into a SOAP message and the response SOAP message into a non-SOAP message.

– The audit/log module is used to audit and log the system events.

Appendix II

Use cases for message security in mobile web services

(This appendix does not form an integral part of this Recommendation)

This appendix explains the use cases of the reference security architecture in Appendix I in detail. Use cases are classified into four categories.

The following conditions are assumed in the use cases:

– Mobile clients or application servers may or may not support the full web services protocol stack.

– Application servers are located in the internal domain of a mobile network operator or in the external domain connected to the Internet.

– Mobile clients, MWSSG and application servers have already obtained certificates issued by trusted third party (TTP) such as CA or XKMS server.

– The client has already been issued security token(s) such as X.509 token, Username token or SAML token; they are inserted into the request message. If the request message is not in SOAP, then equivalent information such as X.509 certificate, id/password or SAML artefact related to the request message are used instead.

II.1 Use case 1

II.1.1 Use case 1-1

Figure II.1 illustrates the use case of the message security between a web services-enabled mobile client and a web services-enabled application server. In this case, the target web service is located in the internal domain of the mobile network operator.

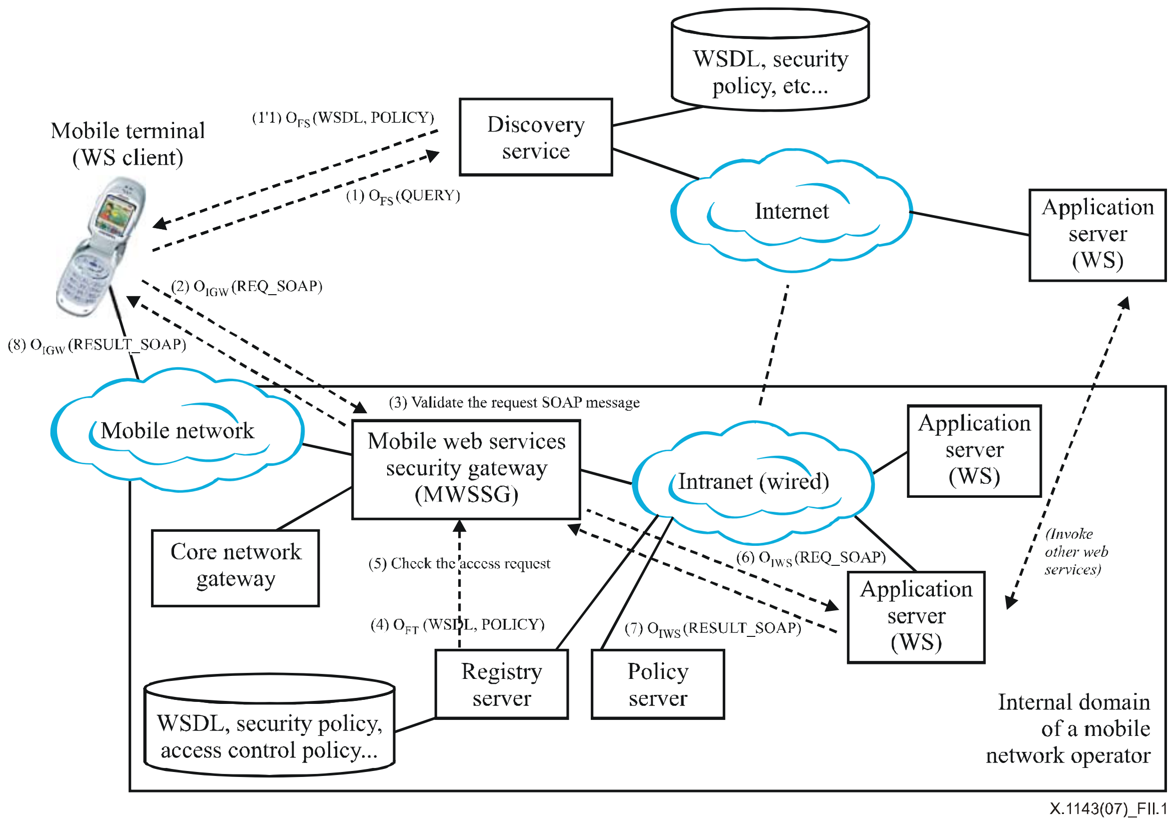


Figure II.1 – Use case 1-1

1) A mobile client finds and accesses the service interface information described in WSDL and security policies of the target service from the discovery service. They have been published by MWSSG. The service interface information described in WSDL contains the endpoint of the MWSSG (not the endpoint of the actual service). The actual endpoint information is stored in the registry server in the internal domain of the mobile network operator; the registry server cannot be accessed directly by the clients.

2) The client sends to MWSSG a request SOAP message that has been secured by applying WS-Security according to the security policies. The SOAP message is generated by referencing the service interface information described in WSDL.

3) MWSSG receives the request SOAP message, validates the schema and checks the known vulnerabilities of the SOAP message, and checks whether the message has been secured according to the security policies. The service interface information described in WSDL and security policies accessed from the discovery service are used. If validation fails, MWSSG should reject the request. Steps 1-7 of clause 8.6 are performed at this step.

4) MWSSG accesses the service interface information described in WSDL and security policies of the target web service from the registry server. The actual endpoint of the target service can be obtained from the service interface information described in WSDL.

5) At this step, steps 8-12 of clause 8.6 are performed to check the access control decision of the request message.

6) MWSSG invokes the target web service by sending the received SOAP message to the target web service using the actual endpoint information. If the SOAP message is not decrypted at the MWSSG and is sent to the target web service, then end-to-end security between the client and the web service is achieved. If the MWSSG and the target web service have a trust relationship, then the MWSSG may decrypt the message, validate its signature, and check the contents if the target web service has poor processing power or if it

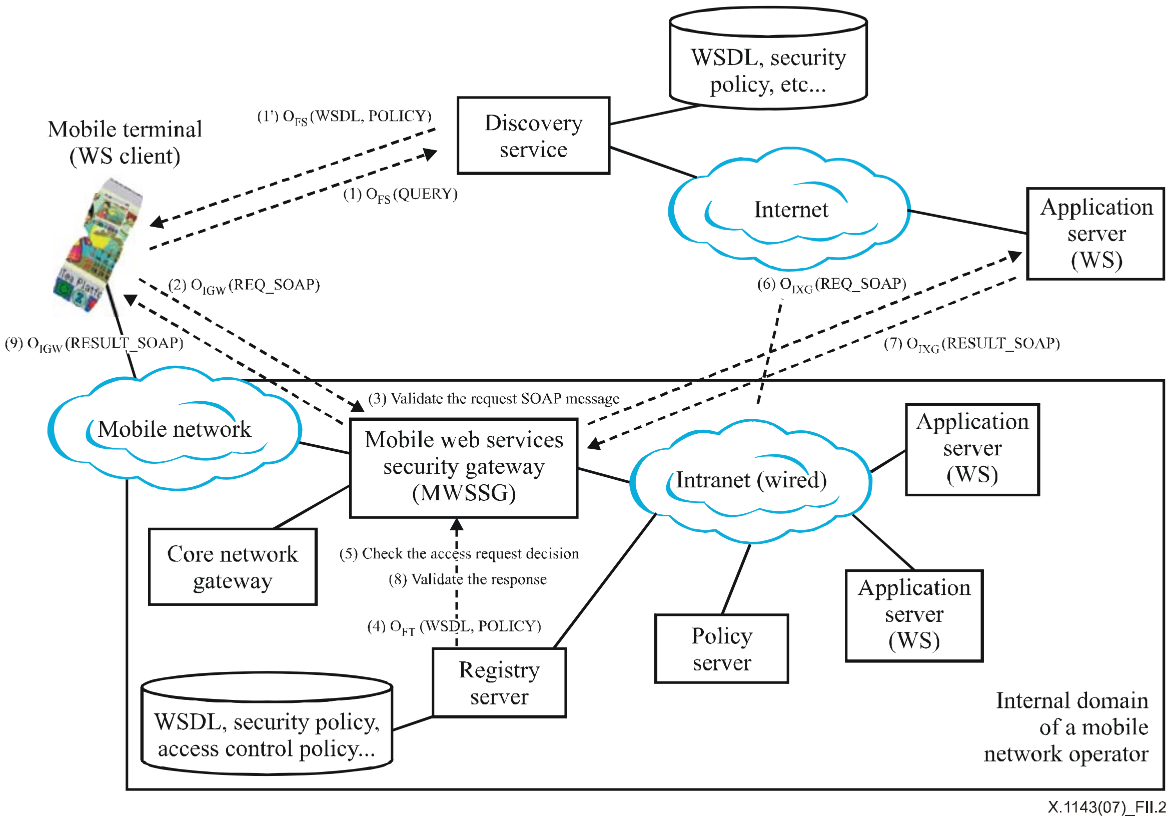
does not have the web services security functionality. The target web service may invoke other web services to perform the requested operations.

7) The response from the target web service is in a SOAP message; it is secured by applying WS-Security according to the security policies of the target service and sent to MWSSG. If the MWSSG and the target web service have a trust relationship, the target service may skip the security processing of the message. In this case, MWSSG secures the message before sending it back to the mobile client according to the security policies accessed from the Registry Server.

8) MWSSG sends the secured SOAP response message to the mobile client.

II.1.2 Use case 1-2

Figure II.2 illustrates the use case of the message security between a web services-enabled mobile client and a web services-enabled application server. In this case, the target web service is located in the external domain of the mobile network operator.



WSDL, security policy, etc...
Discovery service
Mobile terminal (WS client)
(1') O_{FS}(WSDL, POLICY)
(1) O_{FS}(QUERY)
Internet
Application server (WS)
(2) O_{IGW}(REQ_SOAP)
(9) O_{IGW}(RESULT_SOAP)
(6) O_{IXG}(REQ_SOAP)
(7) O_{IXG}(RESULT_SOAP)
(3) Validate the request SOAP message
Mobile network
Mobile web services security gateway (MWSSG)
Intranet (wired)
Application server (WS)
Core network gateway
(5) Check the access request decision
(8) Validate the response
(4) O_{FT} (WSDL, POLICY)
Registry server
Policy server
Application server (WS)
WSDL, security policy, access control policy...
Internal domain of a mobile network operator
X.1143(07)_FII.2

Figure II.2 – Use case 1-2

Steps 1-5 are the same as steps 1-5 of clause II.1.1.

6) This step is the same as step 6 of clause II.1.1, except that the SOAP request message should not be decrypted at MWSSG since the request has to be sent to the external domain. There may be another MWSSG at the target domain, and it may validate the request message in the same manner as the MWSSG of the source site.

7) This step is the same as step 7 of clause II.1.1, except that the target web service should secure the SOAP message using WS-Security (if there is another MWSSG at the target domain, then the SOAP message may be secured by the MWSSG instead of the target service).

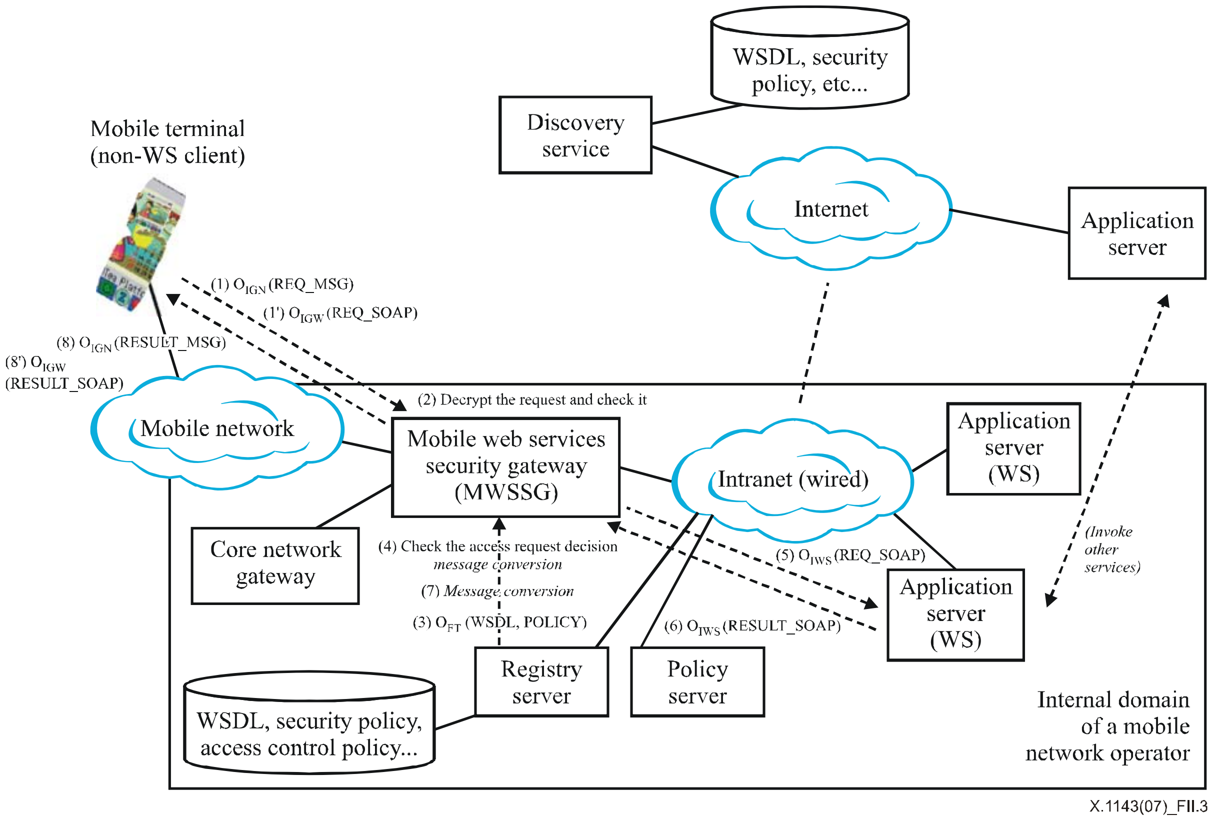
8) MWSSG validates the response SOAP message by validating the schema and checking the known vulnerabilities of the SOAP message and conformance to the security policies. At this step, steps 1-7 of clause 8.6 are performed.

9) If step 8 is successful, the response SOAP message is sent to the mobile client.

II.2 Use case 2

II.2.1 Use case 2-1

Figure II.3 illustrates the use case of the message security between a legacy mobile terminal and a web services-enabled application server. This case covers two cases: the mobile client does not support the web services functionality at all; and the mobile client supports web services but not the WS-Security functionality.



WSDL, security policy, etc...
Discovery service
Mobile terminal (non-WS client)
Internet
Application server
(1) O_{IGN} (REQ_MSG)
(1') O_{IGW} (REQ_SOAP)
(8) O_{IGN} (RESULT_MSG)
(8') O_{IGW} (RESULT_SOAP)
(2) Decrypt the request and check it
Mobile network
Mobile web services security gateway (MWSSG)
Intranet (wired)
Application server (WS)
Core network gateway
(4) Check the access request decision
message conversion
(7) Message conversion
(3) O_{FT} (WSDL, POLICY)
(5) O_{IWS} (REQ_SOAP)
(6) O_{IWS} (RESULT_SOAP)
(Invoke other services)
Application server (WS)
Registry server
Policy server
WSDL, security policy, access control policy...
Internal domain of a mobile network operator
X.1143(07)_FII.3

Figure II.3 – Use case 2-1

The mobile client is assumed to have the interface and security policies information to access the target service via the MWSSG before sending the request.

1) The client sends to MWSSG a request message that has been secured by applying TLS or WTLS according to the security policies. If the client does not support the Web Services functionality at all, then the message is not in SOAP. If the client supports basic web services functionality but not WS-Security, then the message may be in SOAP, and the full message is encrypted by TLS or WTLS. In this case, the interface information described in WSDL from the discovery service may be used to generate the request message.

2) MWSSG decrypts the request and checks it. At this step, steps 1-7 of clause 8.6 are performed.

3) MWSSG accesses the service interface information described in WSDL and security policies of the target web service from the registry server. The actual endpoint of the target service can be obtained from the interface information.

4) Steps 8-12 of clause 8.6 are performed to check the access control decision on the request message. If the request message is not in SOAP, it is converted into a SOAP message. If the request message is already in SOAP, conversion is not required. The request SOAP message should be secured by WS-Security if MWSSG and the target service do not have a trust relationship.

5) The request message is sent to the target web service by MWSSG. The target web service may invoke other application services to perform the requested operations.

6) The response from the target web service is in SOAP message; it may have been secured by applying WS-Security or not depending on the trust relationship between MWSSG and the target service. The response is sent to the MWSSG.

7) MWSSG decrypts the response message if it was secured. If the request message was in SOAP, then it secures the SOAP message using TLS. If the request message was not in SOAP, however, then it converts the SOAP message into a non-SOAP message and secures it by applying TLS.

8) MWSSG sends the resulting SOAP message back to the mobile client.

II.2.2 Use case 2-2

Figure II.4 illustrates the use case of the message security between a legacy mobile terminal and a web services-enabled application server.

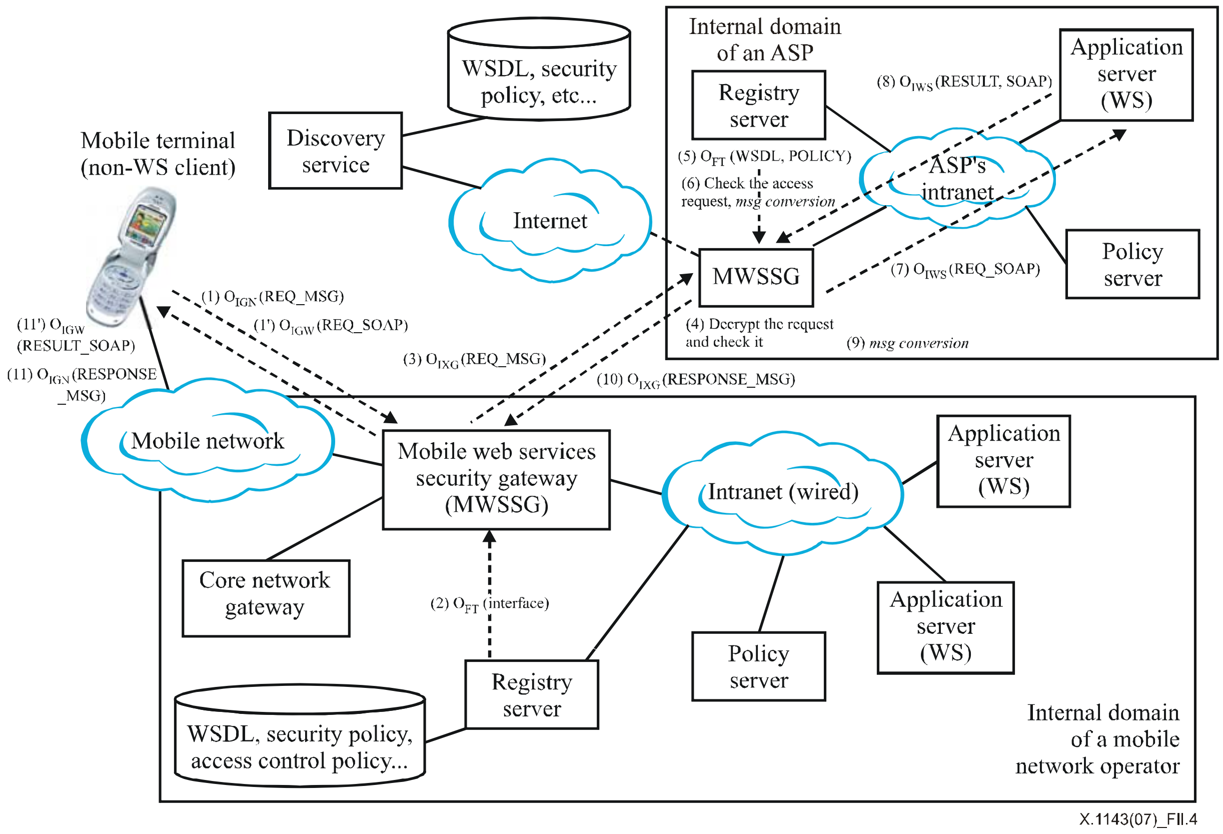

Internal domain of an ASP
Application server (WS)
Registry server
(8) O_{IWS} (RESULT, SOAP)
WSDL, security policy, etc...
Mobile terminal (non-WS client)
Discovery service
(5) O_{FT} (WSDL, POLICY)
(6) Check the access request, *msg conversion*
ASP's intranet
Internet
MWSSG
(7) O_{IWS} (REQ_SOAP)
Policy server
(1) O_{IGN} (REQ_MSG)
(1') O_{IGW} (REQ_SOAP)
(11') O_{IGW} (RESULT_SOAP)
(11) O_{IGN} (RESPONSE _MSG)
(4) Decrypt the request and check it
(9) *msg conversion*
(3) O_{IXG} (REQ_MSG)
(10) O_{IXG} (RESPONSE_MSG)
Mobile network
Mobile web services security gateway (MWSSG)
Intranet (wired)
Application server (WS)
Core network gateway
(2) O_{FT} (interface)
Application server (WS)
Policy server
Registry server
WSDL, security policy, access control policy...
Internal domain of a mobile network operator
X.1143(07)_FII.4

Figure II.4 – Use case 2-2

1) This step is the same as step 1 in clause II.2.1.

2) The local MWSSG accesses the service interface information of the target MWSSG from the local registry server. The service interface information of the target MWSSG was published to the local registry server and used by the local MWSSG to access the destination MWSSG.

3) The local MWSSG bypasses the request to the MWSSG located in the target domain using the service interface information. Since the local MWSSG and the target web service do not have the trust relationship, the local MWSSG is not allowed to decrypt and view the contents of the message.

4) The target MWSSG decrypts and checks the request. At this step, steps 1-7 of clause 8.6 are performed.

5) The target MWSSG accesses the service interface information described in WSDL and security policies of the target web service from the registry server in its domain. The actual endpoint of the target service can be obtained from the interface information.

6) Steps 8-12 of clause 8.6 are performed for access control decision on the request message. If the request message is not in SOAP, it is converted into a SOAP message. If the request message is already in SOAP, however, such conversion is not required. The request SOAP message should be secured by WS-Security if MWSSG and the target service do not have the trust relationship.

7) The request message is sent to the target web service by the target MWSSG. The target web service may invoke other application services to perform the requested operations.

8) The response from the target web service is in SOAP message; it may have been secured by applying WS-Security or not depending on the trust relationship between MWSSG and the target service. The response is then sent to the target MWSSG.

9) The target MWSSG decrypts the response message if it was secured. If the request message was in SOAP, it secures the SOAP message using TLS. If the request message was not in SOAP, then it converts the SOAP message into the non-SOAP message and secures it by applying TLS.

10) The resulting message is sent to the source MWSSG.

11) The source MWSSG sends the response back to the mobile client.

II.3 Use case 3

II.3.1 Use case 3-1

Figure II.5 illustrates the use case of the message security between a web services-enabled mobile terminal and a legacy application server.

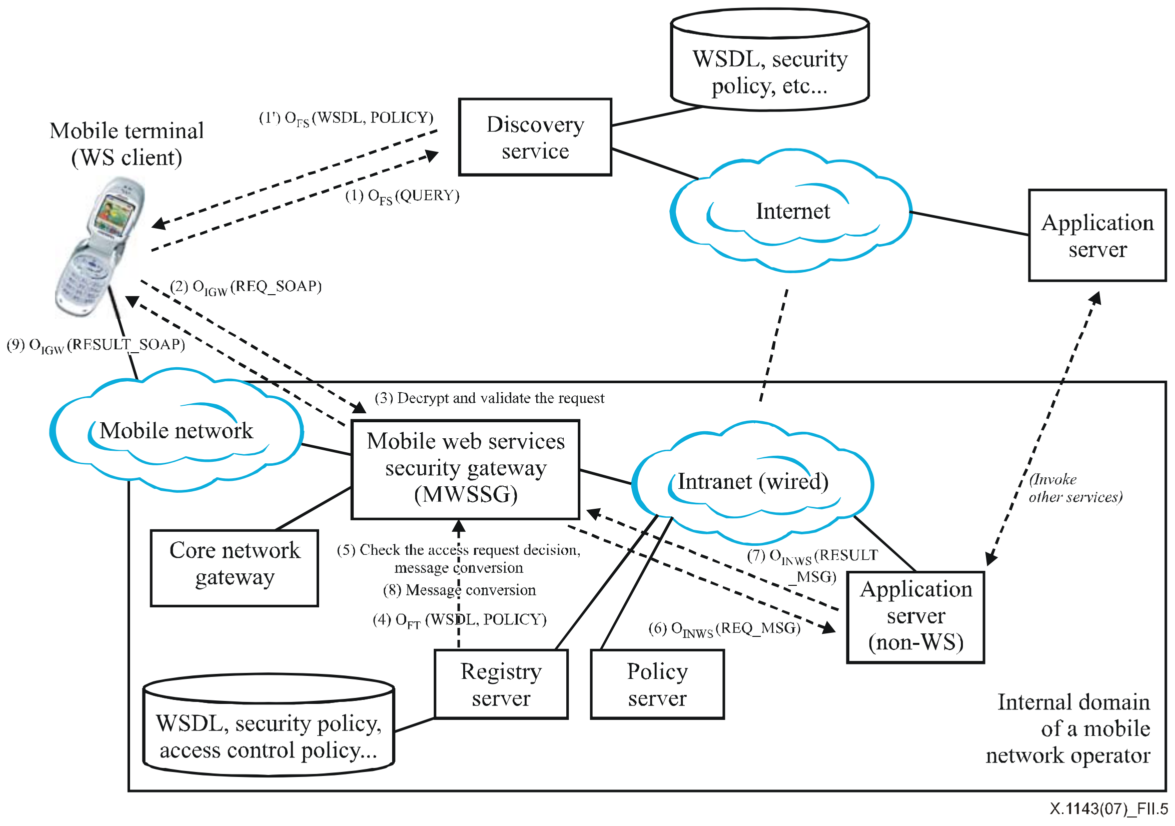

WSDL, security policy, etc...
Discovery service
Mobile terminal (WS client)
(1') O_{FS} (WSDL, POLICY)
(1) O_{FS} (QUERY)
Internet
Application server
(2) O_{IGW} (REQ_SOAP)
(9) O_{IGW} (RESULT_SOAP)
(3) Decrypt and validate the request
Mobile network
Mobile web services security gateway (MWSSG)
Intranet (wired)
(Invoke other services)
Core network gateway
(5) Check the access request decision, message conversion
(8) Message conversion
(4) O_{FT} (WSDL, POLICY)
(7) O_{INWS} (RESULT_MSG)
(6) O_{INWS} (REQ_MSG)
Application server (non-WS)
Registry server
Policy server
WSDL, security policy, access control policy...
Internal domain of a mobile network operator
X.1143(07)_FII.5

Figure II.5 – Use case 3-1

1) A mobile client finds and accesses the service interface information described in WSDL and security policies of the target service from the discovery service. They have been published by MWSSG.

2) The client sends to MWSSG a request SOAP message that has been secured by applying WS-Security according to the security policies. The SOAP message was generated by referencing the service interface information described in WSDL.

3) MWSSG receives the request SOAP message, decrypts it, and verifies the signature of the request message. It validates the schema, checks the known vulnerabilities of the SOAP message, and also checks that the message was secured according to the security policy. The interface information described in WSDL and security policies accessed from the discovery service are used. If validation fails, MWSSG should reject the request. At this step, steps 1-7 of clause 8.6 are performed.

4) MWSSG accesses the service interfaces and security policies of the target service from the registry server. The actual endpoint of the target service can be obtained from the registry server.

5) At this step, steps 8-12 of clause 8.6 are performed to check the access control decision of the request message. The SOAP request message is then converted into a non-SOAP request message that can be understood by the target service. The converted message may be secured again using TLS if necessary.

6) The request message is sent to the application server.

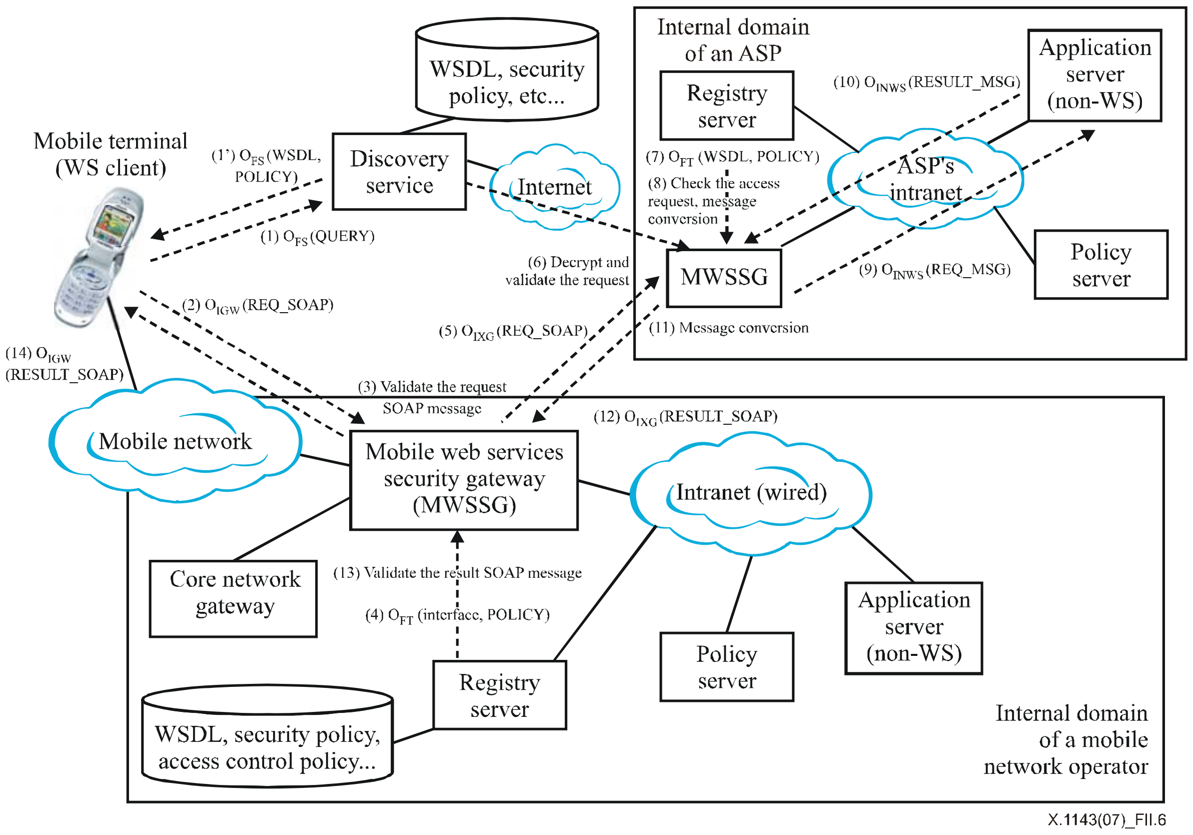
7) The target service receives the request, decrypts it if it was encrypted, and sends the response back to MWSSG. The response may be encrypted using TLS if necessary.

8) MWSSG decrypts the message if it was encrypted and converts the message back into a SOAP message.

9) The response SOAP message is sent back to the mobile client.

II.3.2 Use case 3-2

Figure II.6 illustrates the use case of the message security between a web services-enabled mobile terminal and a legacy application server.



WSDL, security policy, etc...
Internal domain of an ASP
Application server (non-WS)
Registry server
Mobile terminal (WS client)
(1') O_FS (WSDL, POLICY)
Discovery service
Internet
(10) O_INWS (RESULT_MSG)
(7) O_FT (WSDL, POLICY)
(8) Check the access request, message conversion
ASP's intranet
(1) O_FS (QUERY)
(6) Decrypt and validate the request
MWSSG
(9) O_INWS (REQ_MSG)
Policy server
(2) O_IGW (REQ_SOAP)
(5) O_IXG (REQ_SOAP)
(11) Message conversion
(14) O_IGW (RESULT_SOAP)
(3) Validate the request SOAP message
(12) O_IXG (RESULT_SOAP)
Mobile network
Mobile web services security gateway (MWSSG)
Intranet (wired)
Core network gateway
(13) Validate the result SOAP message
(4) O_FT (interface, POLICY)
Application server (non-WS)
Policy server
Registry server
WSDL, security policy, access control policy...
Internal domain of a mobile network operator
X.1143(07)_FII.6

Figure II.6 – Use case 3-2

Steps 1-2 are the same as steps 1-2 in clause II.3.1.

3) MWSSG receives the request SOAP message, validates the schema, and checks the known vulnerabilities of the SOAP message. It also checks whether the message has been secured according to the security policies. The service interface information described in WSDL and security policies accessed from the discovery service are used. If validation fails, MWSSG should reject the request. At this step, steps 1-7 of clause 8.6 are performed.

4) MWSSG accesses the service interface information and security policies of the target MWSSG from the local registry server. The information of the target MWSSG was published to the local registry server and used by the local MWSSG to access the destination MWSSG.

5) MWSSG bypasses the request to the MWSSG located in the target domain using the service interface information from the registry server. Since the local MWSSG and the target service do not have the trust relationship, the local MWSSG is not allowed to decrypt and view the contents of the message.

6) The destination MWSSG decrypts and verifies the signature of the request message and validates the schema, checks the known vulnerabilities of the SOAP message, and checks security policy conformance, since the request is from the external domain. At this step, steps 1-7 of clause 8.6 are performed.

7) The destination MWSSG accesses the service interfaces information and security policies of the target service from the registry server in its domain. The actual endpoint of the target service can be obtained from this registry server.

8) At this step, steps 8-12 of clause 8.6 are performed to check the access control decision on the request. The SOAP request message is then converted into a legacy request message that can be understood by the target service. The converted message may be secured again using TLS if necessary.

9) The destination MWSSG sends the request message to the destination service.

10) The target service receives the request, decrypts it if it was encrypted, and sends the response back to the destination MWSSG. The response may be encrypted using TLS if it necessary.

11) The destination MWSSG decrypts the message if it was encrypted and converts the message back into SOAP and secures it by applying WS-Security according to the security policies.

12) The destination MWSSG sends the response back to the local MWSSG.

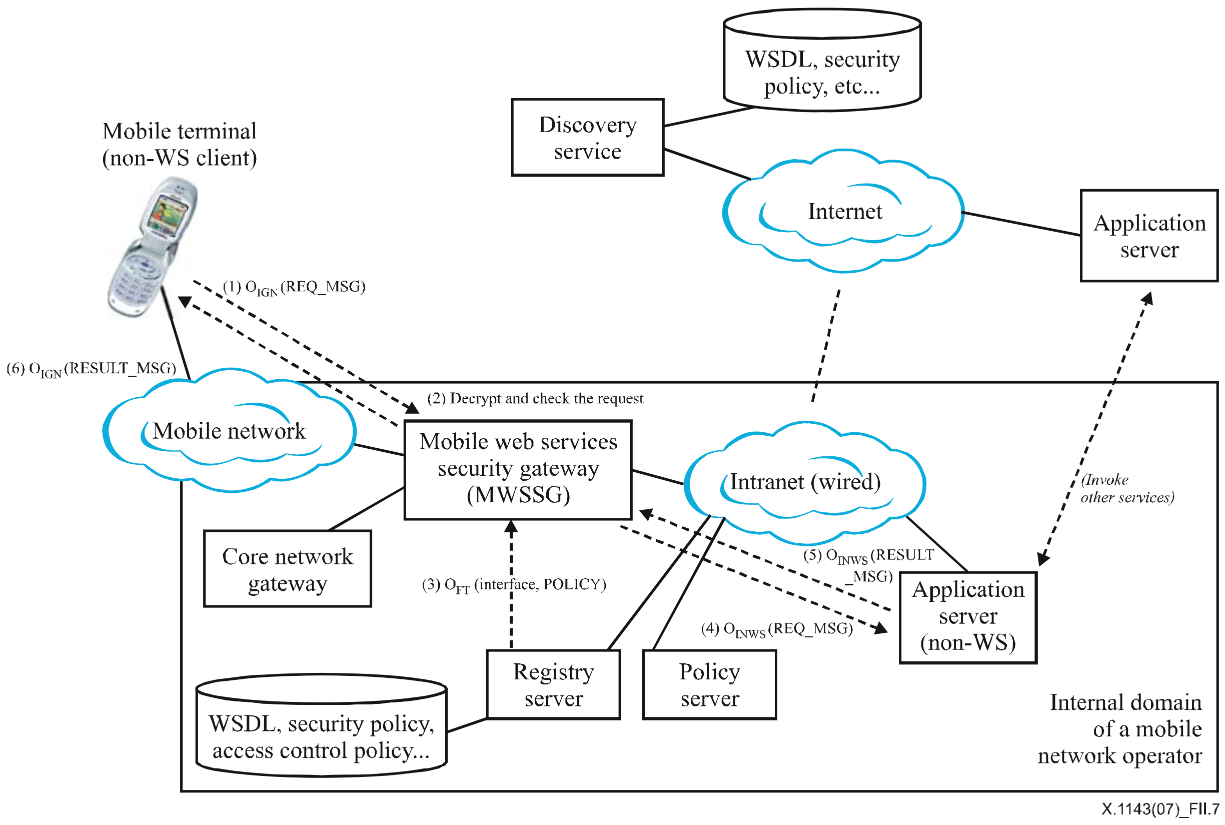
13) The local MWSSG validates the schema, checks the known vulnerabilities of the SOAP message, and security policy conformance, since the request is from the external domain. At this step, steps 1-7 of clause 8.6 are performed.

14) The response message is sent back to the mobile client.

II.4 Use case 4

II.4.1 Use case 4-1

Figure II.7 illustrates the use case of the message security between a legacy mobile terminal and a legacy application server.



WSDL, security policy, etc...
Discovery service
Mobile terminal (non-WS client)
Internet
Application server
(1) O_{IGN} (REQ_MSG)
(6) O_{IGN} (RESULT_MSG)
(2) Decrypt and check the request
Mobile network
Mobile web services security gateway (MWSSG)
Intranet (wired)
(Invoke other services)
Core network gateway
(3) O_{FT} (interface, POLICY)
(5) O_{INWS} (RESULT _MSG)
Application server (non-WS)
(4) O_{INWS} (REQ_MSG)
Registry server
Policy server
WSDL, security policy, access control policy...
Internal domain of a mobile network operator
X.1143(07)_FII.7

Figure II.7 – Use case 4-1

1) The client sends to MWSSG a request message that has been secured by applying TLS or WTLS according to the security policies for the target service.

2) MWSSG decrypts the request and checks whether it is a valid message. In this step, steps 8-12 of clause 8.6 are performed to check the access control decision on the request message.

3) MWSSG accesses the interface information and security policies of the target service from the registry server.

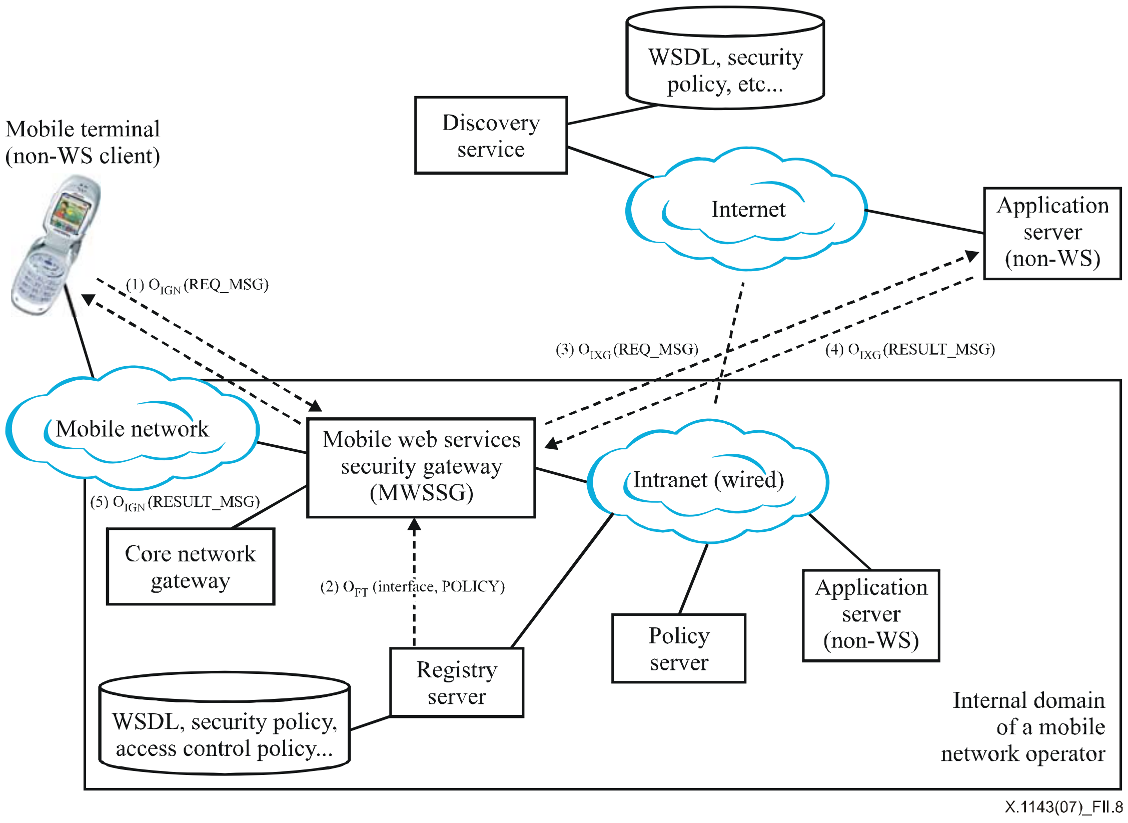
4) MWSSG sends the decrypted request to the target service. It may encrypt the request message again according to the security policies.

5) The target service sends the response message to MWSSG. It may encrypt the response message again according to the security policies.

6) MWSSG receives the response and sends it back to the mobile client. If the response is not already encrypted, MWSSG encrypts the message and sends it back to the client.

II.4.2 Use case 4-2

Figure II.8 illustrates the use case of the message security between a legacy mobile client and a legacy application server.



WSDL, security policy, etc...
Discovery service
Mobile terminal (non-WS client)
Internet
Application server (non-WS)
(1) O_{IGN}(REQ_MSG)
(3) O_{IXG}(REQ_MSG)
(4) O_{IXG}(RESULT_MSG)
Mobile network
Mobile web services security gateway (MWSSG)
Intranet (wired)
(5) O_{IGN}(RESULT_MSG)
Core network gateway
(2) O_{FT} (interface, POLICY)
Policy server
Application server (non-WS)
Registry server
WSDL, security policy, access control policy...
Internal domain of a mobile network operator
X.1143(07)_FII.8

Figure II.8 – Use case 4-2

1) The client sends a request message to the local MWSSG that has been secured by applying TLS or WTLS according to the security policies for the target service.

2) MWSSG accesses the interface information and security policies for the target service or target MWSSG from the registry server.

3) MWSSG simply bypasses the request to the destination service or the destination MWSSG using the information. Since the local MWSSG and the target service do not have the trust relationship, the local MWSSG is not allowed to decrypt and view the contents of the message.

4) The target service receives the request, decrypts it, generates a response, secures it by applying TLS, and sends it back to the local MWSSG.

5) The local MWSSG sends the response back to the mobile client.

Appendix III

Examples of security policy description

(This appendix does not form an integral part of this Recommendation)

This appendix describes examples of security policy.

Examples of security policy are described using WS-SecurityPolicy (see [OASIS WS-SecPol]) and XACML (see [ITU-T X.1142]) since they are generally used for web services. Note, however, that other suitable security policy language may be used. The message filtering mechanism in this Recommendation is not dependent on a specific security policy language.

The following is a simple example of the security policy using WS-SecurityPolicy for specifying the message security requirement:

```
<?xml version="1.0" encoding="UTF-8"?>
<wsp:Policy
       xmlns:sp="http://docs.oasis-open.org/ws-sx/ws-securitypolicy/200702"
       xmlns:wsp="http://www.w3.org/ns/ws-policy"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.w3.org/ns/ws-policy
       http://www.w3.org/2007/02/ws-policy.xsd">
   <wsp:ExactlyOne>
      <wsp:All>
             <wsp:All>
               <sp:SignedParts>
                    <sp:Body/>
               </sp:SignedParts>
            </wsp:All>
            <wsp:All>
                   <sp:EncryptedParts>
                      <sp:Body/>
                   </sp:EncryptedParts>
               </wsp:All>
               <sp:X509Token/>
      </wsp:All>
   </wsp:ExactlyOne>
</wsp:Policy>
```

This policy specifies that <Body> of the SOAP message should be signed and encrypted, and that X.509 token should be used.

The following is an example of the security policy for the access control of messages written in XACML:

```
<?xml version="1.0" encoding="UTF-8"?>
<Policy xmlns="urn:oasis:names:tc:xacml:2.0:policy:schema:os"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="urn:oasis:names:tc:xacml:2.0:policy:schema:os
   http://www.itu.int/ITU-T/formal-language/xml/database/itu-
t/x/x1142/2006/x1142-AnnD.2%20XACML%20policy%20schema.xsd"
   PolicyId="urn:oasis:names:tc:example:Ex1"
   RuleCombiningAlgId="identifier:rule-combining-algorithm:deny-overrides">
<Description>
         Example message access control policy
    </Description>
<Target/>
<Rule RuleId="urn:oasis:names:tc:xacml:2.0:example:Ex1" Effect="Permit">
        <Description>
              Only administrator can Reboot the system
       </Description>
       <Target>
           <Subjects>
              <Subject>
                   <SubjectMatch
MatchId="urn:oasis:names:tc:xacml:1.0:function:string-equal">
                     <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#string">
                           administrator
                     </AttributeValue>
                     <SubjectAttributeDesignator
AttributeId="urn:oasis:names:tc:xacml:1.0:subject:subject-id"
     DataType="http://www.w3.org/2001/XMLSchema#string"/>
                   </SubjectMatch>
              </Subject>
           </Subjects>
           <Resources>
              <Resource>
                  <ResourceMatch
MatchId="urn:oasis:names:tc:xacml:1.0:function:anyURI-equal">
                <AttributeValue
DataType="http://www.w3.org/2001/XMLSchema#anyURI">
                      http://localhost:8080/services/Reboot
                </AttributeValue>
                <ResourceAttributeDesignator
     DataType=http://www.w3.org/2001/XMLSchema#anyURI
     AttributeId="urn:oasis:names:tc:xacml:1.0:resource:resource-id"/>
                   </ResourceMatch>
             </Resource>
           </Resources>
       </Target>
     </Rule>
</Policy>
```

By enforcing this policy, a message that requests the reboot of the system is accepted if the message contains the subject ID "administrator". Otherwise, the request message is rejected.

Appendix IV

Comparison with other standards

(This appendix does not form an integral part of this Recommendation)

IV.1 Parlay X

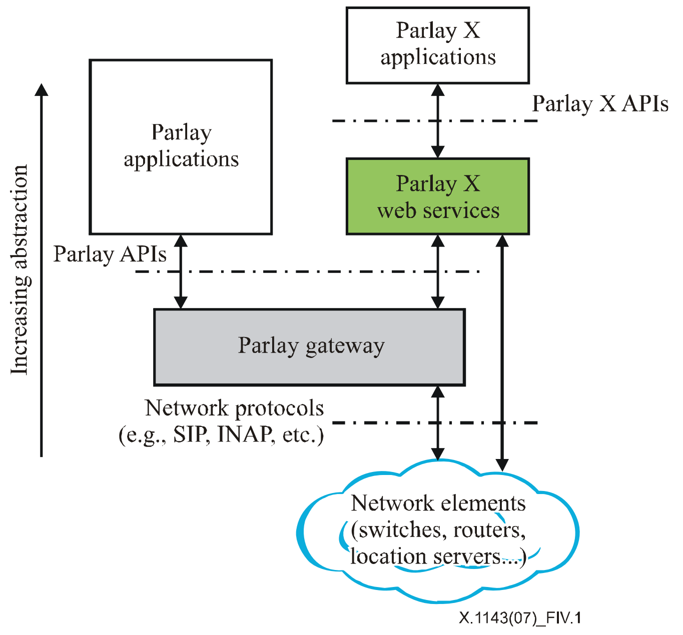


Figure IV.1 – Parlay X architecture

Parlay X is one of the specifications that are currently being worked on by the working groups of Parlay. Parlay describes a network API that allows developers to create applications that access the public network. It is specified in a number of forms, including interface description language (IDL), web services description language (WSDL), and Java. One of the main strengths of Parlay is its network and technology independence. Parlay achieved this through their gateway technology.

Figure IV.1 gives a functional representation of the Parlay X architecture and shows the relationship between Parlay X web services and Parlay APIs. A Parlay gateway typically implements the Parlay APIs. Parlay X web services represent an abstraction and simplification of the Parlay APIs.

MWSSG is a gateway used to secure web services at the message level, and is different from such gateways. MWSSG may use the Parlay gateway to use the functions provided by the network elements required for security management. Parlay/OSA gateway may be accessed using Parlay X web services. Since Parlay X web services are web services-based, these services can also be protected by MWSSG. Parlay gateway and MWSSG are not duplicate gateways; they can be combined for better mobile web services.

IV.2 AAA

Authentication, authorization and accounting (AAA) is the architectural framework for configuring a set of three independent security functions in a consistent manner. AAA provides a modular way of performing the following services:

– Authentication: Provides the method of identifying users, including login and password dialog, challenge and response, messaging support, and encryption, depending on the selected security protocol.

– Authorization: Provides the method for remote access control including one-time authorization or authorization for each service, per-user account list and profile, user group support, and support of IP, IPX, ARA, and Telnet.

– Accounting: Provides the method for collecting and sending security server information used for billing, auditing and reporting, e.g., user identities, start and stop times, executed commands (such as PPP), number of packets, and number of bytes.

MWSSG focuses on the security of messages such as authentication and access control of web services messages. MWSSG assumes that user authentication and authorization are performed using existing security services such as AAA in the mobile operator domain but does not provide duplicate security services with AAA.

IV.3 OMA

The OMA mobile web services (MWS) working group has developed specifications defining the application of Web Services within the OMA architecture. It does not intend to develop new competing specifications where recognized standards exist which have been shown to fulfil the MWS requirements, but to leverage existing standards where applicable to achieve the OMA goal of convergence.

The OMA web services enabler (OWSER) 1.1 specifications define the means by which OMA applications can be exposed, discovered and consumed using web services technologies. [b-OMA OWSER-Core] provides the basic web service infrastructure required to offer and consume web services in an OMA environment. [b-OMA OWSER-over] is informative, providing an overview of the OWSER and web service architecture and technologies to be used to publish, discover and use web services in a secure manner. [b-OMA Style] provides informative guidelines on the use of WSDL that may be used by the OMA-defined Web Services.

OWSER: Overview document describes the practical deployment patterns for web services, including routing pattern, gateway pattern, proxy pattern, interceptor pattern, adapter pattern, delegate pattern, filter pattern, etc.

Figure IV.2 illustrates the gateway pattern and the adapter pattern in OWSER.

The message security architecture in this Recommendation can be categorized into the gateway pattern and the adapter pattern in [b-OMA OWSER-over]. Note, however, that the architecture described in [b-OMA OWSER-over] is inclusive and conceptual, and the concrete security service architecture is not described. On the other hand, this Recommendation describes the detailed security service architecture and usage scenarios. There is no conflict between this Recommendation and underlying mobile web services architecture of OWSER. This Recommendation can be used as a security services architecture and use case that may be integrated into the OMA mobile web services architecture.

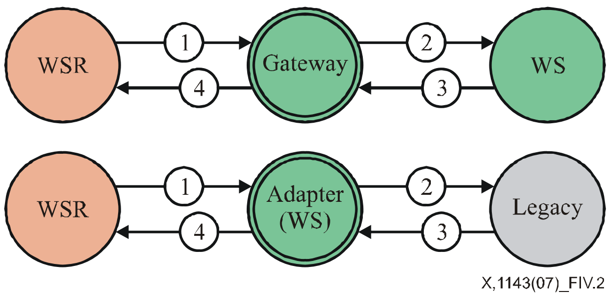


Figure IV.2 – Gateway pattern and adapter pattern in OWSER

This Recommendation complements the OMA mobile web services architecture by providing detailed security services architecture and additional security mechanisms such as message filtering, integrated security policy mechanism, and integration scenarios between various components including non-WS components.

In [b-OMA OWSER-core], security technologies related to web services security, such as WS-Security, XML signature, XML encryption, SAML, XACML and XKMS are recommended for use in the OMA architecture. In that specification, only core components adopted from existing technologies are specified, and it does not provide security services architectures and service scenarios. It does not consider aspects of nodes that have constraining factors for web services, whereas this Recommendation provides such aspects and scenarios. It does not provide message filtering mechanisms, whereas this Recommendation specifies the mechanism.

In conclusion, this Recommendation can be used as use cases for security services that can be used on OMA mobile web services architecture and it complements the OMA architecture by providing concrete security services architecture and additional security mechanisms and integration scenarios between various components including non-WS components. This Recommendation can be regarded as a guideline for message security in the OMA mobile web services environment.

Bibliography

[b-ITU-T X.1121]	ITU-T Recommendation X.1121 (2004), *Framework of security technologies for mobile end-to-end data communications*.
[b-ETSI ES 202 504-1]	ETSI ES 202 504-1 Draft V0.0.3 (2007), *Open Service Access (OSA) Parlay X Web Services Part 1: Common (Parlay X 3)*. <http://www.parlay.org/imwp/idms/popups/pop_download.asp?ContentID=11408>
[b-IETF RFC 2903]	IETF RFC 2903 (2000), *Generic AAA Architecture*. <http://www.ietf.org/rfc/rfc2903.txt>
[b-OASIS UDDI]	OASIS Committee Specification (2002), *UDDI Version 2.04 API Specification*. <http://www.uddi.org/pubs/ProgrammersAPI-V2.04-Published-20020719.htm>
[b-OASIS WS-SConv]	OASIS Standard (2007), *WS-SecureConversation 1.3*. <http://docs.oasis-open.org/ws-sx/ws-secureconversation/200512/ws-secureconversation-1.3-os.html>
[b-OASIS WS-Trust]	OASIS Standard (2007), *WS-Trust 1.3*. <http://docs.oasis-open.org/ws-sx/ws-trust/200512/ws-trust-1.3-os.html>
[b-OMA OWSER-Core]	Open Mobile Alliance OMA-TS-OWSER_Core_Specification-V1_1-20060328-A (2006), *OMA Web Services Enabler (OWSER): Core Specifications, Approved Version 1.1*. <http://member.openmobilealliance.org/ftp/public_documents/mws/Permanent_documents/OMA-TS-OWSER_Core_Specification-V1_1-20060328-A.zip>
[b-OMA OWSER-Over]	Open Mobile Alliance OMA-AD-OWSER_Overview-V1_1-20060328-A (2006), *OMA Web Services Enabler (OWSER): Overview, Approved Version 1.1*. <http://member.openmobilealliance.org/ftp/public_documents/mws/Permanent_documents/OMA-AD-OWSER_Overview-V1_1-20060328-A.zip>
[b-OMA Style]	Open Mobile Alliance OMA-OWSER-Best_Practice-WSDL_Style_Guide-V1_0-20040715-A (2005), *WSDL Style Guide References*. <http://member.openmobilealliance.org/ftp/Public_documents/MWS/2005/OMA-MWS-2005-0086-CR-WSDL-Style-Guide-References.zip>
[b-OMA WTLS]	Open Mobile Alliance WAP-261-WTLS-20010406-a (2001), *Wireless Transport Layer Security, Version 6*. <http://www.wmlclub.com/docs/especwap2.0/WAP-261-WTLS-20010406-a.pdf>
[b-Walsh]	Walsh, Norman., O'Reilly Media, Inc. (1999), *Understanding XML Schemas*. <http://www.xml.com/pub/a/1999/07/schemas/index.html?page=2>
[b-W3C ExCano]	W3C Recommendation (2002), *Exclusive XML Canonicalization Version 1.0*. <http://www.w3.org/TR/xml-exc-c14n/>
[b-W3C Glossary]	W3C Working Group Note (2004), *Web Services Glossary*. <http://www.w3.org/TR/ws-gloss/>

[b-W3C Policy1]	W3C Recommendation (2007), *Web Services Policy 1.5 – Framework*. <http://www.w3.org/TR/ws-policy/>
[b-W3C Policy2]	W3C Proposed Recommendation (2007), *Web Services Policy 1.5 – Attachment*. <http://www.w3.org/TR/ws-policy-attach/>
[b-W3C Schema1]	W3C Recommendation (2004), *XML Schema Part 1: Structures Second Edition*. <http://www.w3.org/TR/xmlschema-1/>
[b-W3C Schema2]	W3C Recommendation (2004), *XML Schema Part 2: Datstypes Second Edition*. <http://www.w3.org/TR/xmlschema-2/>
[b-W3C SOAP]	W3C Recommendation (2007), *SOAP Version 1.2 Part 0: Primer (second edition).* <http://www.w3.org/TR/soap12-part0/>
[b-W3C XKMS]	W3C Recommendation (2005), *XML Key Management Specification 2.0 (XKMS 2.0).* <http://www.w3.org/TR/xkms2/>
[b-W3C XML]	W3C Recommendation (2006), *Extensible Markup Language (XML) 1.1 (second edition).* <http://www.w3.org/TR/xml11/>

Printed in Switzerland
Geneva, 2008