3GPP TSG SA WG3 Security — S3#26 **S3-020637**

19 – 22 November 2002

Oxford, UK

Source:	**Siemens**
Title:	**Work in OMA and W3C on certificate handling**
Document for:	**Discussion**
Agenda Item:	**7.7 (Support for subscriber certificates)**

Abstract

This contribution is to inform the reader about completed and ongoing activities in OMA and W3C on issues of relevance to the work item "subscriber certificates". The contribution highlights that both, the WPKI defined by OMA and XKMS being defined by W3C provide means for a mobile client to request a certificate from a registration authority. In order to secure the certificate request, some bootstrapping information is required. This contribution is to be seen in conjunction with a companion contribution from Siemens, also submitted to SA3#26, where it is argued that there is no need for 3GPP to attempt defining their own certificate request procedures, but that 3GPP should provide means to solving the bootstrapping problem.

---

1. Introduction

There is significant demand to use certificate based trust services via wireless devices to access Internet services. An important prerequisite to use such services is the ability of the wireless device to obtain a suitable certificate. Several initiatives aim to provide this prerequisite in a lightweight manner that is appropriate for wireless environments, e.g. the Open Mobile Alliance (OMA) and the World Wide Web Consortium (W3C).

The OMA specifies the Wireless Application Protocol (WAP) standards, currently in version 2.0. The WAP Public Key Infrastructure (WPKI) specification among other things describes how to obtain wireless client certificates. The W3C specifies several security related Extended Markup Language (XML) based standards. The XML Key Management Specification (XKMS) among other things details how to request certificates in a client-friendly manner.

This article summarises the WPKI specification, as defined by the OMA and the XKMS draft, as defined by the W3C, focusing on the certificate request procedure and its prerequisites. It also sketches the current state of discussion in both fora regarding these matters.

2. OMA/WPKI

The OMA (former WAP Forum) has published four wireless security standards:

1. The Wireless Transport Layer Security (WTLS) is a PKI-enabled transport-level security protocol based on TLS. It has been adapted to low-bandwidth air interfaces in wireless networks. In WAP 2.0 also plain TLS profiled for WAP may be used.

2. The WML Script Crypto API (WMLSCrypt) provides digital signature functionality. It allows to employ WIM services. Further functions like encryption are currently being defined. ECMA Script Crypto Object is currently profiled for use in WAP. This will include functions for key generation, and certificate handling.
3. The WAP Identity Module (WIM) is a tamper resistant unit in the WAP device, commonly implemented using smart cards. It stores key material and provides cryptographic computation services. On board key generation mechanisms and provisioning of key assurance information that proves a key was generated on a WIM are currently being specified.
4. The Wireless Application Protocol PKI (WPKI) is a wireless adaptation of a traditional PKI for mobile environments. Figure 1 below sketches the basic WPKI components and functionality.

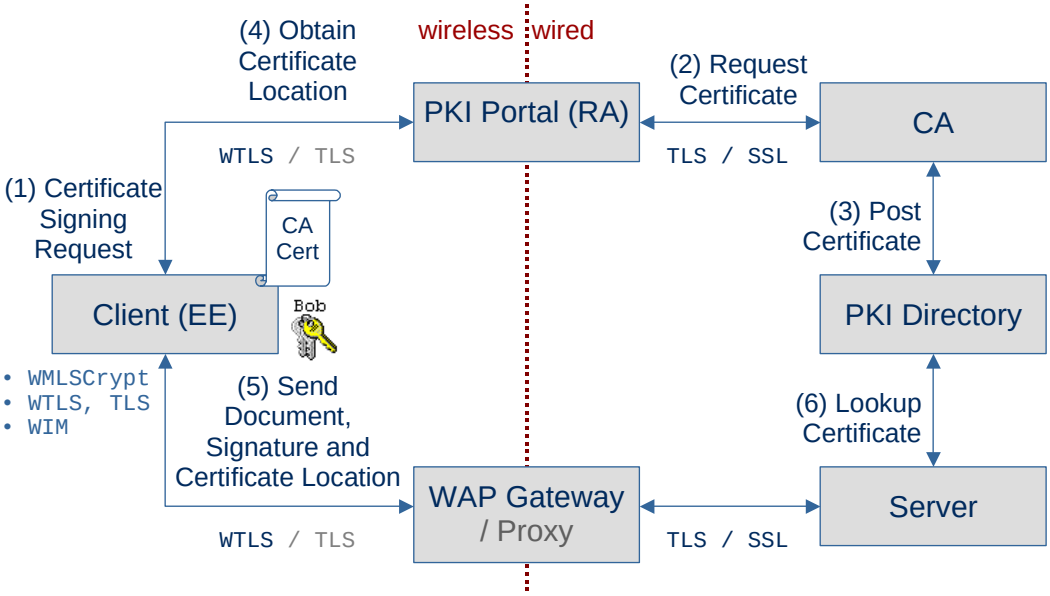

Figure 1: Wireless Application Protocol (WAP) 2.0 PKI

It requires the same **components** like a traditional PKI, i.e. an End-Entity Application (EE), a Registration Authority (RA), a Certification Authority (CA) and a PKI Directory, as depicted in Figure 1. However, in WPKI, the EE and RA are implemented differently, and a new notion, referred to as the PKI Portal, is introduced. The EE in WPKI runs on the WAP device. It uses the WMLSCrypt and the WTLS API, and the WIM for key services and cryptographic operations. It is responsible for the same functions as the EE in a traditional PKI, i.e. key handling, certificate management and validation. The PKI Portal can be a dual-networked system, like a WAP Gateway. It typically functions as RA and is responsible for translating requests of WAP clients to the RA and CA. The PKI Portal interoperates with WAP devices on the wireless side and CAs on the wired network. The PKI Portal adapts the conversation with the wireless client to the capabilities of that clientFor communication with the CA the PKI Portal may use standard certificate management protocols like Certificate Management Protocol (CMP) or Certificate Management Messages over CMS (CMC), both defined by the IETF.

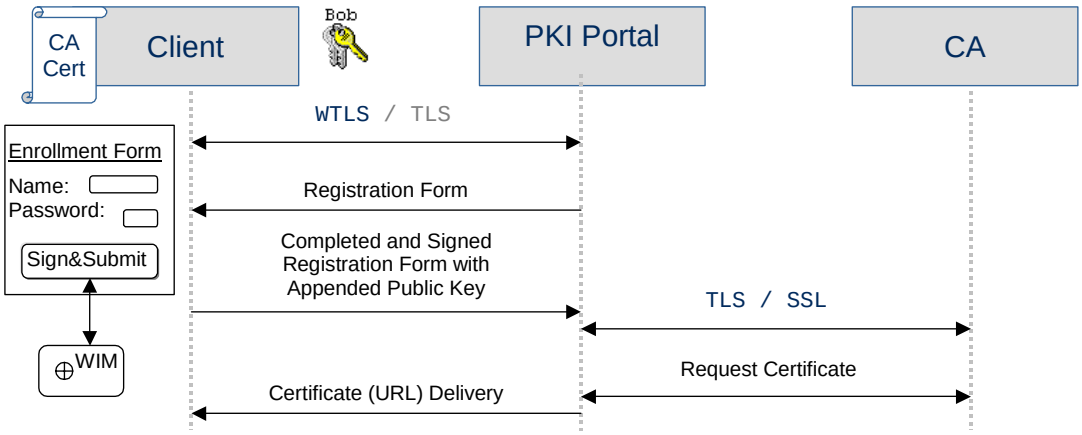

Figure 2: WPKI Client Signature Key Registration Process

Wireless clients may **register** Over-the-Air (OTA) by contacting the PKI Portal (= RA). Different mechanisms are used for authentication and for signature (=non-repudiation) key certification. For authentication keys proof-of-posession is performed via (W)TLS, whereas for signature keys proof-of-posession is given via WMLScrypt SignText. The initial authentication to bootstrap certification may be supported by an existing registration, by passwords transmitted over (W)TLS, or by an existing certificate which may be a certificate on a previously registered public key or a pre-installed device certificate. The notion of a device certificate is sketched in an appendix in the WPKI standard. Device Certificates reside on a device with PKI support like the WIM. A mobile user may be supplied with initial unpersonalized certificates, e.g. by the Mobile Network Operator or SIM/WIM-Card Issuer. The RA can validate the device certificate and knows the private key is on a secure device, normally the WIM. Figure 2 sketches a typical registration information flow, if a client requests certification of a signature key. The client initiates a (W)TLS connection to a PKI Portal. The PKI Portal authenticates itself during the (W)TLS handshake and presents the client with a form to provide the registration credentials, i.e. username and password. On submission, the form is signed by the WMLSCrypt function SignText. By this, the corresponding public key gets appended automatically. The signed and completed enrollment form is returned to the PKI Portal. For registration of an authentication key the process is similar. Only here the client uses the public key to be registered in a (W)TLS handshake together with a self-signed certificate, and client is done again using a password.

Client Certificates in WPKI may be **delivered** Over-the-Air in 3 basic manners. The CA may deliver the certificate to the Client. Here the Client must store and manage the certificate, and relaying parties get the certificate from the client. The CA alternatively may publish the certificate in a directory only, and simply indicate this fact to the client. The relaying party then must search for the certificate by the client's public key id. As a third alternative the CA may return a Client certificate location (URL) to reduce storage and transmission bandwidth. A relaying party here can use the URL to retrieve the certificate.

WPKI **adapts** the IETF PKIX standards for wireless environments, namely PKI protocols, certificate formats, and cryptographic algorithms and keys. It defines the PKI model and operations. WPKI supports WTLS, X.509-WAPCert and X.509-PKIX certificates. Current WPKI applications are entity and message authentication for WTLS and WMLSCrypt. WPKI uses WTLS and the WMLSCrypt signText function, which allow for efficient encoding and submitting of PKI service requests. The WTLS certificate specification provides a simple certificate format in an ad-hoc encoding. The WAPCert specification provides a compact certificate profile as a sub-profile of X.509-PKIX. Both formats reduce the size as compared to standard X.509 certificates. Client certificates use X.509-PKIX format, but, as described before, those will not normally be transmitted over the air or be stored on the client.

Status **validation** mechanisms, like CRLs and OCSP for the wireless client are not yet specified for WPKI. To provide a work-around for the lack of client-side status validation facilities, short-lived server certificates were introduced to obviate the need for a separate revocation check. The CA authenticates a server typically for one year. The CA issues a new short-lived certificate, with a lifetime of typically 48 hours, every day of that year. For revocation the CA simply ceases issuing further short-lived certificates. The client requires a sufficiently accurate time awareness. General use of OCSP within WPKI is discussed within the OMA security group and a recent draft reflecting this discussion dates from August 2002. This draft both profiles the IETF OCSP standard for deployment in WAP and specifies OCSP piggybacking. Certificate status validation support via piggybacking OCSP responses based on an idea by R. Rivest [1] is being discussed for future WPKI versions. Here a server obtains a current OCSP status report for its certificate first, which it then submits to the wireless client together with its other authentication credentials.

A WPKI roadmap with respect to XKMS is discussed within the OMA security group, who in June 2002 also has participated in the requirements engineering of XKMS. XKMS would facilitate PKI integration of mobile clients by offloading certificate handling to XKMS servers. However this requires XMLDSig support, which the OMA security group also considers. It may be of interest that several key driving persons behind both standards are the same, mostly coming from the major PKI companies, like VeriSign, Entrust and Baltimore.

3. W3C/XKMS

XML Key Management Specification (XKMS 2.0) is developed in the W3C. It specifies protocols for distributing and registering public keys, for use in conjunction with the XML Signature specification developed jointly by the W3C and the IETF, and an anticipated companion standard for XML Encryption. A key objective of the XKMS protocol design is to minimize the complexity of client application implementations by shielding them from the complexity and syntax of the underlying PKI used to establish trust relationships. The underlying PKI may be based upon a different specification such as X.509/PKIX, SPKI or PGP. XKMS comprises two parts, the XML Key Information Service Specification (X-KISS) and the XML Key Registration Service Specification (X-KRSS).

The **X-KISS** specification defines a protocol for a trust service that resolves public key information and validates certificates. The X-KISS protocol allows a client of such a service to delegate part or all of the required tasks. Its functions include the location of required public keys and describing the binding of such keys to identification information. For example, the key may be specified by a name only, the local trust policy of the client may require additional information in order to trust the key, or the key may be encoded in an X.509 certificate that the client cannot parse. In the case of an encryption operation the client may not know the public key of the recipient and ask the trust service to locate it.

The **X-KRSS** specification defines a protocol for a web service that accepts registration of public key information, supports certificate revocation, and in certain cases also key recovery. In most applications, the Registration Service will provide key information to other trust services like a separate underlying PKI such as PKIX. Once registered, the public key may be used in conjunction with other web services including X-KISS.

A client may request that the Registration Service bind information to a public key. The information bound may include a name, an identifier or extended attributes. The key pair to which the information is bound may be generated in advance by the client or, to support key recovery, may be generated on request by the service. In the latter case the registration protocol may also be used for subsequent recovery of a private key.

The X-KRSS protocol provides for authentication of the applicant and, in the case that the key pair is generated by the client, Proof of Possession (POP) of the private key. A means of communicating the private key to the client is provided in the case that the private key is generated by the Registration Service. Currently means of registering RSA and DSA keys

are specified, as well as a framework for extending the protocol to support other cryptographic algorithms such as Diffie-Hellman and Elliptic Curve variants.

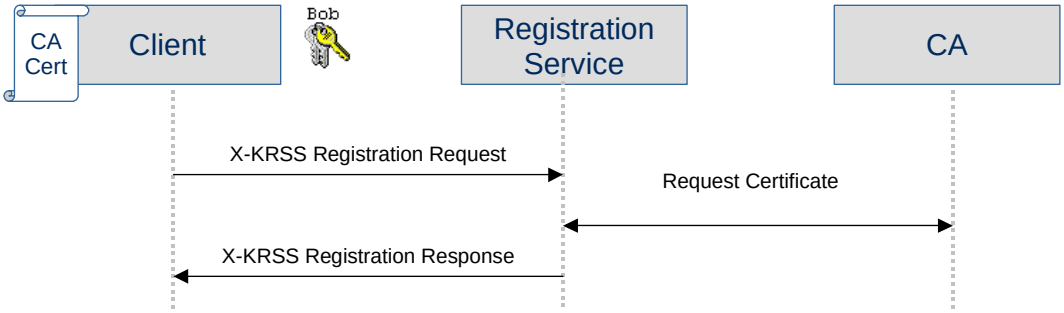

Figure 3: X-KRSS Client Registration Process

The basic X-KRSS client **registration** information flow is sketched in Figure 3. For the client it basically consists of a single request-response pair. The registration request message indicates the requested assertion, which the registration service should provide. The registration service may require the client to provide additional information to authenticate the request. It may require the client to provide Proof of Possession of the private key.

```
<Register>
   <Prototype Id="keybinding">
      <Status>Valid</Status>
      <KeyID>MyKeyID</KeyID>
      <ds:KeyInfo>
         <ds:KeyValue><ds:RSAKeyValue> .... </ds:RSAKeyValue></ds:KeyValue>
         <ds:KeyName>MyKeyName</ds:KeyName>
      </ds:KeyInfo>
      <PassPhrase>
         HMAC-SHA1 (HMAC-SHA1 ("MyRevocationPassphrase", 0x2), 0x3)
      </PassPhrase>
   </Prototype>
   <AuthInfo>
      <AuthUserInfo>
         <ProofOfPossession><ds:Signature .... /></ProofOfPossession>
         <KeyBindingAuth>
            <ds:Signature URI="#keybinding" [HMAC-SHA1 (KeyBinding,
                        HMAC-SHA1 ("MyBootstrappingPassword", 0x1))] />
            </KeyBindingAuth>
      </AuthUserInfo>
   </AuthInfo>
   <Respond>
      <string>RetrievalMethod</string>
   </Respond>
</Register>
```

Figure 4: X-KRSS Client Registration Request Example

As sketched in the example in Figure 4, the client may provide a password to bootstrap the certification.

The **response** to an X-KRSS certification request may among other things include a Key Name, Public key parameters, X509v3 certificates or certificate chains, that authenticate the specified key, an X509v2 Certificate Revocation List, a PKIX OCSP token that validates an X509v3 certificate that authenticates the key, PGP key signing data or a collection of them, and SPKI key signing data, as well as encrypted private key data.

3. Summary

PKI plays an important role in meeting e- and m-commerce security requirements. WPKI is a modification of a traditional PKI. WPKI has been adapted using more efficient cryptography and data transport techniques in order to work with today's personal wireless devices and the narrow-band wireless networks. During the WPKI client enrolment procedure, the client may submit a username and password to bootstrap the certification.

XKMS is developed by the W3C. It specifies protocols for distributing and registering public keys. A key objective of the XKMS protocol design is to minimise the complexity of client application implementations by shielding them from the complexity and syntax of the underlying PKI. The XKMS Key Registration Service Specification (X-KRSS) allows clients to request enrolment from a Registration Service. During the XKMS client enrolment procedure, the client may submit a key name and a password to bootstrap the certification.

4. References

[1] Ronald L. Rivest, "Can We Eliminate Certificate Revocations Lists?", in: Financial Cryptography, p. 178-183, 1998, http://citeseer.nj.nec.com/rivest98can.html

WPKI-Related Standards (http://www.wapforum.org)

- WAP-211-X.509 profiles the contents, formats & encoding for WAP certificates & CRLs.
- WAP-198-WIM defines WIM requirements.
- WAP-217-WPKI profiles the existing PKIX standards for the wireless environment.

XKMS-Related Standards (http://www.w3.org/2001/XKMS)

- XML Key Management Specification (XKMS 2.0, 18 March 2002)
- D. Eastlake, J. R., D. Solo, M. Bartel, J. Boyer , B. Fox , E. Simon. XML-Signature Syntax and Processing, World Wide Web Consortium
- D. Box, D Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H. Frystyk Nielsen, S Thatte, D. Winer. Simple Object Access Protocol (SOAP) 1.1, W3C Note 08 May 2000