

Outlining Wireless Public Key Infrastructure

Abstract: This paper describes on-going work in the WAP Forum on Wireless Public Key Infrastructure (WPKI), describing the current specifications, their motivation and status and likely future developments¹.

Stephen Farrell,
Chief Security Architect, Baltimore Technologies, Monday, 10 July 2000

Introduction

Recent developments in wireless networks, and in particular in wireless access to the Internet, have resulted in the development of a number of specifications in various industry fora. This paper describes the WAP Forum's [WAP] work on PKI, which is part of the WAP 1.3 specifications that will be finalised late in calendar 2000. The emphasis in this paper is on describing the motivation behind the resulting specifications, which are freely available for download from the sites given in the references.

In order to understand the context of this work, it is useful to examine briefly some of the differences between wired and wireless networks, or at least those aspects that affect how security services are provided. A more broadly based description can be found in [IAB00].

Traditionally, the operators of wireless networks have had much more control over the use of the networks than in the wired case. This has led to the "walled garden" approach to running the network, where the operator has significant control while they are within the walled garden and there are some practical restrictions on connections to the wider Internet. The WAP Forum has to some extent also followed this model.

When dealing with wireless networks one cannot ignore the fact that one is dealing with various constraints that do not apply in the typical Internet case. The best known constraints are limited bandwidth, high message latency, the very limited user interface, and the requirement for a small code "footprint", all of which limit our ability to use existing security (and other) services which have typically been designed without considering such constraints.

In fact, there are two additional constraints that are often more influential. Firstly, the relative lack of technical awareness of the average user compared to the average PC user limits the amount of useful security-related user interaction. Secondly, the fact that users are generally not interested in security functionality (also true for wired networks, but generally forgotten!), further decreasing the footprint that is willingly devoted to security functionality.

It also has to be noted that the business models involved in wireless differ from the normal Internet case. This is most clearly shown by the fact that network operators pay most of the cost of the mobile equipment.

Finally, and still not well understood, is the fact that a lot of the current Internet PKI work has implicitly assumed that the end users are adults. At least in the GSM "world", it is now common for minors to have their own phones. This means that we can no longer depend on drivers' licenses, passports, bank account details, etc. as a means of establishing identity and poses an interesting challenge to PKI vendors.

Some vendors (Baltimore Technologies included) have deployed solutions (such as Baltimore Telepathy) that address these issues and are involved in working with partners to prove these innovations and extensions to traditional PKI in large scale environments.

¹ The opinions expressed in this paper are those of the author and not necessarily those of Baltimore Technologies nor of the WAP Forum.

WAP Security Services

In this section, we very briefly describe the set of security services defined by the WAP Forum, which led to the need to define a Wireless PKI for WAP. This section assumes some familiarity with the overall WAP architecture, for details see [WAP-ARCH], and only outlines the WAP wireless transport layer security [WAP-WTLS] protocol, the WAP cryptographic API [WAP-CRYPTO] and the concept of the wireless identity module [WAP-WIM].

When a WAP device connects to a WAP gateway, it can connect in “secure mode” which means using the WTLS protocol. This protocol is modelled on the current Internet transport layer security TLS protocol [RFC2246], which, in turn is an open standards version of the Secure Sockets Layer [SSL] protocol. WTLS is almost the same as TLS in the sense that the exchanges involved in the handshake protocol, the application and alert protocols all fulfil the same functions, and are the same even in many details. If you understand SSL or TLS, you will have no difficulty understanding WTLS.

WTLS does differ from TLS in a few important ways: the sets of cryptographic algorithms supported are different, it uses more compact encodings for over-the-air (OTA) data and it supports some new models for the efficient re-use of previously established session keys. WTLS also defines classes of WTLS connection: class I in which both parties are anonymous; class II in which the WAP gateway is authenticated; and class III in which both parties are authenticated. Confidentiality is provided for in all cases. Note that these classes could also have been defined for TLS or SSL (in both cases class II is most commonly used). For our purposes we only need note that at class II, servers must be part of a PKI and for class III, both clients and servers must be part of a PKI.

One significant way in which WTLS differs from TLS is that it includes the definition of a new public key certificate format, the WTLS certificate, of which more later.

In addition to securing bytes sent between the WAP device and gateway, there was also an appreciation that some security services would be needed for transaction data. To this end the concept of a cryptographic API was defined. This defines a set of functions (currently only one) that can be called from WML scripts (which can be thought of as being analogous to JavaScript).

Today the only function defined in the WAP cryptographic API is a text signing function named `signText()`, which offers the end user the ability to sign data in response to calls issued from content downloaded to the WAP device. The specification of `signText()` is such that the output data can be repackaged, (on the wired side), to be PKCS#7 [PKCS#7] compliant. The end user is also to be prompted for a special PIN, for each call to `signText()` as a means of ensuring that users are not easily tricked into digitally signing data – an action that may in the near future have legal consequences.

Of course, in order to use the `signText()` function the user needs a private key. In order for the signature to be checked, the public key is normally certified in a PKI.

Finally, the WAP Forum has defined a WAP Identity Module [WAP-WIM], which in a sense extends the GSM concept of Subscriber Identity Module (SIM) to include providing interfaces for cryptographic operations. The net result is that in WAP devices that do have WIMs (they are optional), the most sensitive cryptographic keys (private keys, pre-master-secrets) are never exposed outside the WIM.

It is also possible to have a single smartcard providing both SIM and WIM functionality (a so-called S/WIM). The three models – no WIM, separate WIM and SIM, and single S/WIM – can have a significant effect on the business models and PKI policies that apply.

In particular, as many near-term WAP devices are unlikely to include asymmetric key generation functionality, the WIM or S/WIM production facilities and practices become an important item to consider when deploying a WAP PKI.

WAP PKI Specifications

So, in September 1999, the WAP Forum's security group (WSG), realising that WAP security services required the provision of a wireless PKI in order to scale to the expected sizing, began serious work on the WAP PKI. At this point a number of PKI vendors also got involved in WSG, bringing to bear an in-depth knowledge of on-going efforts such as the IETF's PKIX WG [PKIX].

The prior expectation in WSG had been that the wireless PKI would be based on a fairly simple extension of the WTLS certificate format. However, it quickly became clear that PKI involved a lot more than this. In fact, it is probably accurate to compare a PKI built on the WTLS certificate format as being similar to the 1988 version of the X.509 [X.509] standard. For a description of X.509 and how it has been built into a PKI over the past five years in the IETF, see [ROADMAP]. It is an understatement to say that WSG was not keen to repeat all this work, particularly given the timeframe (6-9 months) in which it had to produce a workable PKI specification.

One facet of the evolution of X.509 worthy of mention is the movement towards making digital signatures legally significant, in particular in the EU and the USA. The fact is that most of this work had been based on an implicit assumption that there would be an underlying X.509 based PKI. In fact, current technical and policy work in this area is largely aimed at teasing out the detailed requirements so that digital signatures produced by end-entities in an X.509 PKI can be legally significant.

All this had to be matched against the existing WAP WTLS, WIM and cryptographic API specifications and the limitations inherent in wireless networks. Of course, none of these limitations had been considered when developing the existing X.509 based PKI specifications.

The end result was the development of two specifications: the WAP PKI definition [WAP-WPKI] and a profile of the X.509 public key certificate format suited for certificates that are to be stored on a WAP device or transmitted over the air [WAP-CERTPROF]. It is important to realise that these specifications are intended to be a pragmatic response to the situation presented above.

We will now look at the most important aspects of these specifications, but will not give full details – the reader is referred to the WAP Web site [WAP], where the specifications themselves are available for download.

PKI Portal

According to [RFC2459] PKIs consist of a number of components: Certification Authorities (CAs), Registration Authorities (RAs) and end-entities (EEs). The WPKI specification does not aim at defining any new components but does define a PKI portal as being a PKI component (either a CA or RA) that is "WPKI aware". This is the PKI entity that is contacted by WAP devices and gateways when they use any WAP specific PKI mechanism. Typically, the PKI portal will be an RA or the on-line interface to a CA.

Certificate Formats

The core of the current WAP PKI specification is the realization that the WTLS certificate format is supported in current devices and will have to continue to be supported for some time. However, currently deployed devices only support WTLS class II and do not support either WTLS class III or the signText() function. This meant that existing server (more properly gateway) certificates had to support the WTLS certificate format, but that the situation with client certificates was still open as there was no existing deployment.

Also worth noting is the fact that `signText()` output might well “escape” from WAP’s current walled garden. This means that the signature verification code could not be expected to be WAP-aware in all cases. These signatures are also much more likely to be required to be “legally significant” than those used in WTLS.

In addition, there will be many more WAP devices than gateways, and this means that it is much more important to re-use existing infrastructure for the many (clients), as opposed to the few (gateways).

The end result is that the current WPKI specification effectively mandates that current WAP gateways will be certified using the WTLS certificate format and that WAP clients will be certified using X.509 public key certificates. Note, however, that all of the current specifications allow either certificate format to be used in the protocol. So the above fact is only really apparent from the WPKI specification.

Following on from this, there are consequences for the formats of CA information (or CA certificates). Clearly if WAP devices are to verify WTLS certificates for WAP gateways, it makes sense that they handle CA information in the same format. This means that the CAs that issue WTLS certificates (for WAP gateways) should make their public keys available using the WTLS certificate format.

In fact, as the WTLS certificate format is very limited, it is also probable that only a single level of PKI can be usefully supported, that is, it is very likely that CAs that are considered (by the WAP device) as being “root” CAs will directly certify WAP gateways.

Similarly, CAs that certify WAP clients using the X.509 public key certificate format should publish their CA information using the X.509 format.

In summary then: WTLS certificates are to be used for WAP gateways and X.509 public key certificates are to be used for WAP devices and end-users.

Root CA Handling

As the root CA information on the WAP device is based on the WTLS certificate format, which includes a validity period, there is a need for a way to update this information over-the-air before existing settings “expire”. Of course, there is also a need to be able to introduce new root CA information over-the-air and the WPKI specification addresses these two needs by defining secure schemes for sending new root CA information over-the-air.

This led to some debate within WSG, as the solution adopted would have significant impact on security, usability and the “open-ness” of the wireless PKI in terms of business models². Eventually, two different schemes were defined which have fairly different characteristics as described below.

The first scheme is based on the out of band distribution of a hash of the root CA information³. In this scheme the user can simply download the root CA information over an insecure channel, but to “activate” the CA, the user must enter a 30 decimal digit value, distributed out of band. Of course, this is not very good for users, but it was felt that something better than the current Internet model of “just clicking ok” should be provided. It should be noted that this scheme allows the independent introduction (or recovery!) of new CAs to the device, without, for example, operator control. In certain situations, e.g. in a bank branch, the user could be assisted with entering this value.

² Effectively, this could have closed off some answers to the “who’s the CA?” question.

³ Actually “derived from a hash” would be more accurate as the value includes effectively 80 bits of hash and some checkdigits.

The second scheme was based on the idea of leveraging existing CA information to “introduce” new CAs. Basically, an existing CA digitally signs the new CA information. This scheme is much better for the user, as no interaction is required, but, on the other hand, has unknown implications for the two CAs’ policies! Clearly, in order for one CA to be willing to “introduce” another CA, it has to know quite a lot about the new CAs policies and may, in theory at least, be exposing itself to some liability.

There is also a method defined to allow a CA to “roll-over” its CA information, which is syntactically identical to the signature scheme above, but where the new and old CA information relate to the same CA. Finally, the specification also allows for other mechanisms to be used, for example, where an operator has a secure provisioning capability in place this could be used to update CA information on the WAP device.

Revocation and Short-Lived Gateway Certificates

X.509 and PKIX define a number of mechanisms for revoking public key certificates when the binding between an identity and a public key should no longer be considered “safe”⁴. Unfortunately there is currently no scheme which is suitable for use in a WAP client. The reason is that the existing schemes either require a potentially large certificate revocation list (CRL) [RFC2459] to be sent to the client or require the client to contact an on-line certificate status responder using, for example, the OCSP protocol [RFC2560].

The bandwidth available to current WAP clients precludes the use of CRLs and the requirement for a new connection to the OCSP responder cannot be met with current devices which typically only support a single connection to a single gateway at any given moment. Resolving this is an area of active work in WSG, but, in the meantime, a scheme of “short-lived” gateway certificates has been developed that can provide similar functionality in the near term.

The idea is that the typical WAP gateway will generate a key pair, produce a PKCS#10 [PKCS#10] certification request and send this to a WTLS certificate enabled CA. The CA, having validated the request according to its certification practice statement, can issue a WTLS certificate for the gateway and send this back to the gateway.

The WTLS certificate produced by the CA might be valid for either a long (say one year) or a short (say two day) period. If the certificate is only valid for a short period, the WAP clients are less exposed to “bad” gateway certificates, because the “badness” will presumably only continue undetected for a short period. There is also a specification of a simple way for the gateway to “pick up” newly issued certificates (presumably also short-lived). This allows a CA to produce a new certificate for a WAP gateway, say every day, which is automatically picked up by the WAP gateway, while allowing the CA to limit its exposure as usual. The net effect is roughly the same as if the CA produces a CRL at the same frequency.

Certificate URLs

As WAP devices have limited storage, and can be quite hard to contact from a server, it is obvious that storing a client’s own certificate on the client’s device might not always be a good idea. After all, a client may have many certificates (but still have limited storage) and certificates expire and must be renewed, which is a problem (for users and PKI operators) if you have to get rid of the old one and replace it each year. In addition, if a certificate is stored on the client, it can only be used if it is sent over-the-air, and bandwidth is another limited resource!

The solution adopted first recognizes that clients that only sign, never need to see their own certificates! That is, they would only use their certificates to send to signature verifiers, whether for WTLS class III or signText().

⁴ For some definition of “safe”!

Given these facts, it was suggested that instead of storing their certificates, clients could store a certificate URL that they then send over-the-air to verifiers. The verifier, presumably having fewer bandwidth limitations, can de-reference the URL and retrieve the client's certificate.

Doing this requires that the URL has a format that allows the verifier to check that the retrieved certificate and URL "match" and such a format is defined in the WPKI specification.

Client Certificates

There exist a variety of models which result in WAP clients possessing certificates, ranging from cases where a new WIM is produced after the client's identity is established to cases where the client only registers in a PKI over-the-air. Since the WPKI specification is aimed at providing interoperability between different WAP vendors' products this over-the-air registration case is clearly within the scope of the WPKI specification.

The PKIX group have defined a number of fairly feature-rich protocols that can be used for such registration cases [RFC2510, RFC2797], however WSG felt that it was unreasonable to expect that device manufacturers would include support for such complex protocols in their current devices. At this point WSG faced the prospect of inventing a new format, or adopting a pragmatic near-term solution – of course the latter approach prevailed⁵.

The idea is rooted in noting that essentially two different things happen in most registration cases – the client "proves" its identity and also "proves" that it possesses the private key corresponding to the public key which is to be certified⁶. So, if a client contacts a PKI portal and does these two things, even if it does them in a proprietary fashion, then the PKIX protocols [RFC2510, RFC2797] can be used by the PKI portal to create a standard registration message for transmission to a PKIX CA or RA. Stated another way: WPKI defines neither registration, nor certificate request message formats, but does allow standard, PKIX, mechanisms to be used beyond the PKI portal.

What WPKI does specify (loosely) is that the client can use either WTLS Class III or signText() as the mechanism for proving possession of the relevant private key. In other words: "prove it by using it".

In the WTLS case, the proof of identity can be established under the protected WTLS session. For signText() the proof of identity can be established by choosing the to-be-signed text appropriately. For example, if a client has been provided with a one-time use "password" via some out-of-band mechanism, then the use of this password indicates to the PKI portal that the correct client is at the other end of the connection.

This creates a "bootstrap" issue – what certificate does the client use when proving possession to a PKI portal? There are two relatively simple solutions available: the client can create a "self-signed" certificate for this purpose when it "knows" that it is dealing with a PKI portal, or perhaps better, the client's device can be provided with a "device certificate"⁷ that "proves" that the relevant private key is present in a device of the indicated type.

Registration into a PKI can produce results immediately, or after an extended period (say up to a week), depending on the certification policy in force, and how the proof of identity is handled. This leads to a requirement that the PKI portal is able to indicate to the client that a definitive response is not ready, but may be ready after a delay. The WPKI defines a response type for this purpose which allows the return to the WAP client of: a "check-back" instruction, the actual certificate, or perhaps more commonly, the certificate URL.

This then is the approach taken for over-the-air registration in the current WPKI specification.

⁵ It is also true to say that the expectation is that the device and networks will improve to the extent that this will no longer be an issue, and that the "wired" world protocols will in future take wireless cases into account.

⁶ Where "proves" is to be interpreted liberally!

⁷ This option is suitable for the common case where the keys are generated before the user acquires the device, which will be the most common case in the near future.

Certificate Profile Specification

For those cases where X.509 certificates (say WAP device certificates) are to be stored in clients, or transmitted over-the-air, we again face the typical wireless constraints of storage and bandwidth limitations. Typical X.509 certificates can be large (given that they are very extensible), with 2,000 byte certificates not being uncommon. Using 2,000 bytes for each certificate would not be the wisest use of the limited storage available on current clients – people far prefer to be able to store phone numbers!

In order to provide guidance for developers, the certificate profiles specification [WAP-CERTPROF] defines a subset of PKIX [RFC2459] certificates that must, or should be supported on clients. The main constraint is that these certificates must be smaller than 700 bytes. It is important to note that this specification only applies to certificates that are stored on clients or sent over-the-air, so that the certificates referenced via certificate URLs are not constrained and may use all of the X.509 extensions that are appropriate.

Status of the Specifications

At the time of writing (July 2000) the WPKI definition and certificate profile specifications are at the “Proposed” stage in the WAP Forum’s process. This means that some minor changes are considered likely before the specifications reach the “Approved” stage. In fact, since the Proposed specifications have been published, on-going work within WSG has identified a number of minor errors and clarifications which will be applied in due course (probably becoming visible outside WAP Forum in Q42000). The specifications are currently open for wider review and comments from interested parties are welcomed. The WAP Forum’s Web site [WAP] provides a mechanism for submission of comments.

Future developments

The current specifications are aimed at supporting the overall WAP 1.3 conformance level (although vendors are introducing some WPKI support even in WAP 1.2 conformant products). As such it is not expected that any major changes will be made to them at this point, however minor changes and clarifications will be applied as implementation experience warrants.

The current focus within WAP Forum, and WSG, is on the WAP 2.0 specifications which are intended to open up the walled garden and provide much greater end-to-end interoperability between WAP devices and the Internet in general. To this end, existing Internet security specifications such as TLS and IPSec [RFC2401] are being examined. The likely outcome is that WSG will specify profiles of these specifications (i.e. subsets) which WAP 2.0 devices will implement. WSG will also seek to influence the further development of these Internet specifications so that the needs of wireless devices are handled better. For example, the certificate URL concept can provide benefits to wired, as well as wireless client devices.

In addition, WSG is examining the needs for additional application layer security functionality, for example, it is possible that more functions like signText() will be defined for the WAP cryptographic API.

In terms of the impact on the WPKI, a similar approach may be expected, aiming at interoperability with current Internet specifications and attempting to influence the evolution of those specifications.

Conclusions

In conclusion we should say something about the importance of wireless and more particularly WAP and about the lessons learned from involvement in this process.

The main lesson learned is that there was quite a gap between those involved in wireless and those involved in the more general Internet – often the two groups did not appreciate the constraints and requirements under which the other had to operate. The good news is that this gap is significantly smaller than it used to be, at least in the WSG.

Finally, from a PKI perspective it is clear that wireless and WAP has the potential to have a very significant impact on the deployment of PKI. Literally millions of devices have already been manufactured meeting the current WAP specifications (1.1) and market forecasts expect that hundreds of millions of devices will be WAP compliant over the period up to 2003. The combination of so many devices able to generate digital signatures, and the newly introduced digital signature legislation in many countries, makes it more and more likely that the use of PKI and digital signatures will become the dominant authentication technology of the next decade.

December 2000 Update

Since the original article was written, the WSG have decided that for WAP 2.0 (expected to be fully defined during 2001), WAP clients will run TCP stacks and will use TLS for security. To that end WSG have proposed some (backwards compatible) extensions to the IETF TLS working group. The actual changes proposed are very similar to those between WTLS and TLS.

Introducing Baltimore Technologies

Baltimore Technologies develops and markets security products and services to enable companies to develop trusted, secure systems for e-business, the Internet and mobile commerce. Its products include a wide range of [Public Key Infrastructure \(PKI\)](#) products and services, wireless e-security solutions, cryptographic toolkits, access control and authorization, content security (MIMEsweeper products) security applications and hardware cryptographic devices.

Baltimore Technologies employs over 1,000 people worldwide and operates from over 30 cities, with headquarters in Dublin, Ireland; London, UK; Boston, USA and Sydney, Australia. For further information and press releases on Baltimore Technologies, please visit www.baltimore.com.

About Baltimore Telepathy

Baltimore Telepathy (<http://www.baltimore.com/telepathy>) is the most comprehensive open standards based wireless security solution infrastructure, designed to meet the most demanding security needs of : Wireless Network Operators, Content Providers (including Financial Institutions and Mobile Enterprises) and Technology Vendors. Comprising of powerful Developer Toolkits and innovative infrastructural components, Baltimore Telepathy enables secure and trusted services to mobile customers by securing communications and transactions from the end-user to a company's e-commerce and IT systems. Baltimore's Telepathy is fully compatible with the award-winning Baltimore UniCERT PKI architecture to provide security for wireless e-business and e-commerce systems.

References

- [IAB00] *"Overview of 2000 IAB Wireless Internetworking Workshop"*, RFC 3002, D. Mitzel, December 2000.
URL: <ftp://ftp.isi.edu/in-notes/rfc3002.txt>
- [PKCS#10] *"PKCS #10: Certification Request Syntax Version 1.5"*, IETF RFC 2314, B. Kaliski, March 1998.
URL: <ftp://ftp.isi.edu/in-notes/rfc2314.txt>
- [PKCS#7] *"PKCS #7: Cryptographic Message Syntax Version 1.5"*, RFC 2315, B. Kaliski, March 1998.
URL: <ftp://ftp.isi.edu/in-notes/rfc2315.txt>
- [PKIX] IETF Public-Key Infrastructure (X.509) Working Group
URL: <http://www.ietf.org/html.charters/pkix-charter.html>
- [RFC2401] *"Security Architecture of the Internet Protocol"*, IETF RFC 2401, S. Kent, R. Atkinson, November 1998.
URL: <ftp://ftp.isi.edu/in-notes/rfc2401.txt>
- [RFC2246] *"The TLS Protocol Version 1.0"*, IETF RFC 2246, T. Dierks, C. Allen, January 1999
URL: <ftp://ftp.isi.edu/in-notes/rfc2246.txt>
- [RFC2459] *"Internet X.509 Public Key Infrastructure Certificate and CRL Profile"*, R. Housley, et al, January 1999.
URL: <ftp://ftp.isi.edu/in-notes/rfc2459.txt>
- [RFC2510] *"Internet X.509 Public Key Infrastructure Certificate Management Protocols"*, IETF RFC 2510, C. Adams and S. Farrell, March 1999.
URL: <ftp://ftp.isi.edu/in-notes/rfc2510.txt>
- [RFC2560] *"Internet X.509 Public Key Infrastructure – On-line Certificate Status Protocol – OCSP"*, IETF RFC 2560, M. Myers, R. Ankney, A. Malpani, S. Galperin, C. Adams, June 1999.
URL: <ftp://ftp.isi.edu/in-notes/rfc2560.txt>
- [RFC2797] *"Certificate Management Messages over CMS"*, IETF RFC 2797, M. Myers, et.al., April 2000.
URL: <ftp://ftp.isi.edu/in-notes/rfc2797.txt>
- [ROADMAP] *"Internet X.509 Public Key Infrastructure PKIX Roadmap"*, draft-ietf-pkix-roadmap-05.txt, work-in-progress, A. Arsenaut, S. Turner, March 2000
URL: <http://www.ietf.org/internet-drafts/draft-ietf-pkix-roadmap-06.txt>
- [SSL3] *"The SSL 3.0 Protocol"*, A. Frier, P. Karlton, and P. Kocher, Netscape Communications Corp., Nov 18, 1996.
URL: <http://home.netscape.com/eng/ssl3/draft302.txt>
- [WAP] The WAP Forum Web site.
URL: <http://www.wapforum.org/>
- [WAP-ARCH] *"WAP Architecture Specification"*, WAP Forum, 30 April 1998.
URL: <http://www.wapforum.org/>
- [WAP-CERTPROF] *"WAP Certificate and CRL Profiles Specification"*, WAP Forum, 18 February 2000
URL: <http://www.wapforum.org/>
- [WAP-CRYPTO] *"WML Script Crypto API"*, WAP Forum, 5 November 1999.
URL: <http://www.wapforum.org/>
- [WAP-WIM] *"WAP Identity Module Specification"*, WAP Forum, 5 November 1999.
URL: <http://www.wapforum.org/>
- [WAP-WPKI] *"WAP Public Key Infrastructure Definition"*, WAP Forum, 3 March 2000.
URL: <http://www.wapforum.org/>

- [WAP-WTLS] *"Wireless Transport Layer Security Specification"*, WAP Forum, 5 November 1999.
URL: <http://www.wapforum.org/>
- [X.509] *"Information Technology – Open Systems Interconnections – The Directory: Authentication Framework"*, ISO/IEC International Standard 9594-8 | ITU-T Recommendation X.509 (2000).