AUTHORIZATION OF SDR SOFTWARE USING COMMON SECURITY TECHNOLOGY

Rainer Falk (Siemens Corporate Technology, Munich, Germany, rainer.falk@siemens.com); François Haettel (Thales, Neuilly-sur-Seine, France, francois.haettel@fr.thalesgroup.com); Ulf Lücking (Nokia, Bochum, Germany, ulf.lucking@nokia.com); Eiman Mohyeldin (Siemens, Munich, Germany, eiman.mohyeldin@siemens.com)

ABSTRACT

A central issue for reconfiguration is authorization (or certification) of reconfiguration software, defining which software is accepted from whom. A well-known and widely used security mechanism to protect software download is signed content. The paper describes how standard digital signatures based e.g. on PKCS#1 or DSA signatures and X.509 certificates can be used for certification of radio software. Depending on how they are employed, different certification models can be realized (vertical market model, horizontal market model).

1. INTRODUCTION

Reconfiguration allows modifying the configuration of communication equipment as mobile devices and base stations during their operation by software download or by parameter changes. This allows adapting the equipment flexibly to changing conditions depending on the current environment and user and operator preferences.

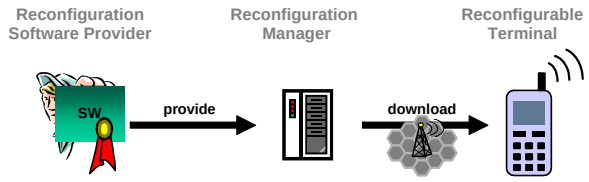

Figure 1: Reconfiguration Software Download

Figure 1 shows the download of reconfiguration software: It originates from a software provider that attaches a digital signature to the actual software. This software is downloaded to a reconfigurable terminal from a network-based reconfiguration manager.

The improved flexibility poses the threat of downloading wrong or malicious software, i.e. of reconfiguration software that is not working as expected or that causes harm by intention or that changes the configuration of a device in a way that contradicts the interests and expectation of end users, network operators and service providers, equipment manufacturers and also regulatory authorities. Such malicious radio software could invalidate essential radio conformance properties, and it could also lead to other types of harm: Security mechanisms required e.g. for secure network access could be weakened or circumvented, a user's private data (contacts, appointments) could be sent to unauthorized parties, or expensive premium rate numbers could be called in the background.

A central concept for secure download of reconfiguration software is the authorization (or certification) of reconfiguration software: A reconfigurable device accepts only duly authorized reconfiguration software. The authorization of reconfiguration software can be classified according to the stakeholder that is the origin of restrictions:

- Regulator (radio software relevant for conformity of radio emissions, e.g. transmission frequency, emission power, product responsibility),
- Manufacturer (bug fixes, enhancement or change of baseband algorithms, installation of optimized protocol stack components),
- Network operator (e.g. monitoring and selection of most suitable radio technology, handover decisions, and medium access algorithms),
- Service provider (e.g. "branding" of user interface, software needed for service-provider specific services),
- End user (e.g. applications, user interface themes, background images, ring tones).

A well-known and widely used security mechanism to protect software download is digitally signed content. The provider of a software module attaches a digital signature to the module that can be verified by the receiving device. The digital signature ensures that the module cannot be modified (integrity) and it attests the software provider (authentication of origin). By attaching a digital signature computed with his private key, the software provider certifies the software.

That means that he makes a statement that this is "good software". The receiving device validates the signature of a received software module and checks whether it originates from a trusted provider. The software provider is the entity that acts in the role to authorize a piece of software. In real world, it could be – depending on the chosen market model – be the device manufacturer, a testing house (industry consortium or approved regulatory body), the user's service provider, or the software developer directly.

The paper describes how standard digital signatures based e.g. on PKCS#1 or DSA signatures and X.509 certificates can be used for certification of radio software (software defined radio, SDR). Depending on how these known mechanisms are employed, different certification models can be realized:

- Vertical market model with certification by device manufacturer
- Horizontal market model with certification of each radio hardware-software combination by an approval authority (industry consortium or regulatory body)
- Horizontal market model with separate certification of radio hardware and software (as above by an industry consortium or a regulatory body)

Section 2 summarizes relevant commonly used security technology (digital signature, certificates, signed content), and section 3 describes how this technology can be employed to realize different market models for radio software authorization, the required security infrastructure The required steps will be described to certify a new hardware device or a new piece of software.

2. RELEVANT SECURITY TECHNOLOGY

This section gives an overview of the basic underlying security technology: Digital signatures based on public-key cryptography, and how to apply them for signing a software module (signed content).

2.1. Digital Signature

Public key cryptography employs an algorithm using two different but mathematically related "keys", one called the private key for creating a digital signature (or transforming data into a seemingly unintelligible form), and another key called public key for verifying a digital signature (or returning the message to its original form). The two keys are called private key and public key. The private key needed to compute a digital signature is the one that has to be kept secret. The operation of computing a valid digital signature can be performed only by the entity that possesses the private key. The digital signature, i.e. the result of the former operation, can be verified using the corresponding public key. Everyone who has access to the public key can perform the verification step. This step checks whether the digital signature has in fact been computed using the corresponding private key. Due to the properties of the public-key cryptosystem, it is not practically possible to determine the private key from a known public key. In practice, the asymmetric cryptographic signing operation is not applied to the data to be signed, but to a cryptographic hash value of that data that can be computed efficiently. A

In an asymmetric cryptographic system, only the private key has to be kept secret, while the public key can be distributed freely. But the parties using the public key must use the correct, authentic public key, i.e. the one that really corresponds to the private key of the intended peer entity. For this purpose, public-key certificates issued by a Certification Authority (CA) bind a public key to an identity and possibly further data. Certificates according to the X.509 standard [6] are used most commonly in practice. The infrastructure for public-key certificates is called public key infrastructure (PKI). It is also possible that a certificate is revoked. To check the revocation status, a certification revocation list (CRL) can be used containing a list of revoked certificates, or the status of a single certificate can be checked online using the online certificate status protocol (OCSP) [10].

Well known algorithms for digital signatures are RSA/PKCS#1 (Rivest-Shamir-Adleman, named after its inventors; Public Key Cryptography Standard) [8] and DSA (Digital Signature Algorithm) [4]. First a digest value of the content to be signed is computed with a cryptographic hash function as MD5 or SHA-1 (but note that collisions have been found for MD5). Then the actual asymmetric digital signature algorithm is computed of the digest value. A widely used format for cryptographic messages as signed content is PKCS#7 respectively cryptographic message syntax (CMS) [5]. The CMS/PKCS#7 format supports inclusion of certificates needed to verify the signature, it supports multiple signers, and the signed content can be contained, but it is not required to be. So the signature and the signed content can be encoded as a single data structure, but it is as well possible that the actual software and the signature are separate (detached signature).

2.2. Signed Content

In the case of signed content, a digital signature is added to a piece of content such as a software module. It is used to attest that a certain software module originates from a trusted provider. This provider is the one who has computed the digital signature. The receiving device is now capable of verifying the digital signature to check whether the software module has been tampered with (integrity) and whether it

originates from a known, trusted provider (authentication, authorization).

Signed content can be used for example for signing MIDlet suites in MIDP2.0 [15]. What is actually signed is the Java archive (JAR file), where data contained in the JAR file is protected by the digital signature, including – besides the actual Java code – also meta information that is part of the Manifest file. Instead of relying on PKCS#7, here the RSA PKCS#1 signature is encoded directly in the Java application descriptor. Also the certificates needed for signature verification are embedded in the application descriptor. This has the advantage that the application descriptor contains security information that can be verified even before the actual Java archive file is downloaded. To complete the verification, the actual digest of the downloaded JAR file has to be verified to match the reference digest as asserted by the digital signature. Further examples for utilizing signed content are Microsoft Authenticode [9] and the Symbian operating system [14].

Four protection domains are distinguished by the "recommended security practice for GSM/UMTS compliant devices" if MIDP2.0: manufacturer domain, operator domain, trusted third party domain, and the untrusted domain. A downloaded MIDlet suite is put in one of the domains depending on whether it is signed and by whom. The restrictions enforced by the execution environment depend on the domain in which a MIDlet has been put.

3. SOFTWARE DOWNLOAD AUTHORIZATION

This section describes how the basic mechanism of signed content can be employed to realize different market models for reconfiguration software. The market model defines who is a legitimate, authorized provider of reconfiguration software:

- Vertical market model: radio software certification by device manufacturer
- Horizontal market model with certification of each radio hardware-software combination by an approval authority (industry consortium or regulatory body)
- Horizontal market model with separate certification of radio hardware and software (as above by an industry consortium or a regulatory body)

While the basic mechanisms for secure software download are well known, specific for secure download of radio reconfiguration software is the enforced policy that defines which party can create an accepted signature and thereby indicate towards the target device that the radio software module is authorized (certification, approval). This digital signature attests here that conformance properties are not invalidated. Please note that this meaning of the digital signature is not a direct consequence of the cryptographic digital signature, the meaning comes from the policies followed when computing this digital signature.

In the vertical market model, radio-related software is accepted only if it is authorized by the device manufacturer. Here, the device manufacturer can ensure that conformance properties are met, but also that a proper, reliable operation is ensured as he is still in control concerning which radio software is accepted on devices he brought into the market. Alternative approaches for secure radio software download that are suitable for horizontal market models are a current research topic. Approaches include combined or separate approval for radio hardware and radio software, moving the responsibility to validate a radio configuration to a network-based function, or to supervise radio emissions during operation and to perform reactive measures if a malfunction is detected.

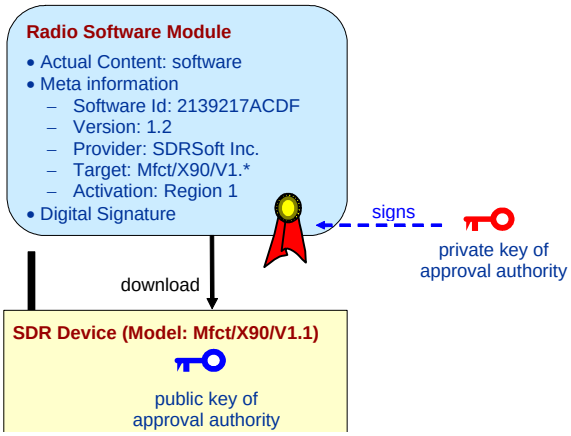

Figure 2: Signed Radio Software Module

Figure 2 illustrates the basic concept of radio software authorization based on signed content by an approval authority: An approval authority authorizes a radio software module by computing and attaching its digital signature using its private key. The module's meta information contains entries to identify the module, the authorized target device, and restrictions on the activation (to allow for different regulations for radio equipment that vary on the region). The device verifies the signature using the public key of the approval authority. Depending on the desired policy, this approach can be used to realize a vertical market model where only software authorized by the device manufacturer is accepted as well as to realize a horizontal market model where each hardware-software-combination requires authorization from a trusted approval authority. The target devices for which a radio software module is authorized to be used on can be encoded as part of the meta-

information of the radio software module (the meta information is covered by the digital signature as well as the actual software). What distinguishes these cases is which entity is acting as approval authority in the real world (e.g. single manufacturer, industry consortium, testing house, regulatory approval body).

When a single reconfigurable device is used in different regions (global roaming), variations in radio certification rules have to be distinguished: The policy for radio software authorization followed by the reconfigurable equipment may vary. For example, some regions might require approval by a regulatory authority while other regions follow a more open approach. Therefore, the reconfigurable device needs to be aware of the region/location in which it is currently used, and switch to and enforce the corresponding radio software authorization policy. This policy defines in particular the trusted parties who can authorize a software module (root certificate, public keys) and possibly restrictions based on evaluation of the software module's meta information.

3.1. Vertical Market Model

As long as the downloaded software modules are specific to a single device type, e.g. a firmware update, patch or an additional feature such as support for an additional radio standard, it seems to be most natural that a reconfigurable device accepts only software modules authorized by its device manufacturer (vertical market model). Here, the device manufacturer can not only ensure that conformance properties are met, but also ensures a proper operation as he is still in control on which radio software is accepted on devices he brought into the market. This approach has already been followed by the 3GPP 23.057 MExE (Mobile Execution Environment) standard [2]. It requires that the support of core software download functionality – e.g. a codec, a new air interface, software defined radio – in a MExE device shall only be under the control of the MExE device manufacturer. Similar to MExE, also the MIDP2.0 recommended security practice for GSM/UMTS compliant devices distinguishes manufacturer, operator, third party and untrusted domains. Although MIDP 2.0 [15] considers actually only Java MIDlets (applications), the security infrastructure (keys, certificates) could be re-used also for other types of manufacturer-signed software.

A manufacturer signed software module is authorized for a specific target device. For simplicity, no certificates are used; the software module is digitally signed using directly the private key of the device manufacturer. The target device is indicated as part of the meta information, here as combination of manufacturer name (Mfct) and model (X90). But in general, other information, e.g. an approval number, could be used to identify the target device for which the software module is authorized. It could also be implicitly encoded by using a different manufacturer signing key for each device model, but this would lead to unnecessary complexity. Optionally, also a data element can be included to uniquely identify each software module, but even without an explicit identification, each software module can be identified uniquely by its cryptographic digest computed with a one-way hash function as e.g. SHA-1.

The receiving device verifies the digital signature, i.e. it ensures that the received software module has been approved (authorized) by its manufacturer and that it has not been manipulated. It also compares whether the software module is indeed intended for the type of the target device. For this purpose, the device compares the indicated target (Mfct/X90) with its own reference identifier. The digital signature could be attached to the download module or it could be a separate, detached signature. The latter case can be advantageous when a large software module is authorized for several target devices: Download servers have to store the software module only once, and an additional small detached signature file for each target device.

The following subsections describe the steps that have to be performed in the case that a new software module shall be approved, or when a new hardware model is brought into market.

When the device manufacturer wants to authorize an additional reconfiguration software module, he attaches his digital signature to this module. With a valid signature, a target device will recognize the module as authorized.

3.1.1. New Software Module

When the device manufacturer wants to authorize an additional reconfiguration software module, he attaches his digital signature to this module. With a valid signature, a target device will recognize the module as authorized.

An updated software module can be handled in the same way. The module has to be signed by the device manufacturer. The version number can be encoded also as part of the software module's meta information. While the same software identifier would be used, the version number would be increased.

3.1.2. New Hardware Device Type

A new hardware device model needs to store the trusted public key of its manufacturer. This key is stored on the device by the device manufacturer during manufacturing. The manufacturer also has to compute a digital signature for

all those reconfiguration software modules that shall be authorized for the new device model. The new device model is indicated as target device in the meta information. For older devices, the already computed signatures remain valid. When the new device is fully backwards compatible with some older device types, it could also be programmed to accept radio software modules targeted at these older device types. In this case, the older software modules do not have to be signed explicitly for the new device type.

3.2. Horizontal Market Model

It is interesting to note that the well-known solution for secure software download based on signed content as described above is sufficient also to realize a vertical market model where each hardware-software-combination requires authorization from an approval body, a model underlying e.g. the "Tally" download system [16]. When using the well-established signed content approach, the digital signature would be computed by a trusted approval authority, e.g. an industry consortium or a regulatory body instead of the device manufacturer. The reconfigurable device would have to store the approval authority's public key to be able to verify the signature, and to compare the target identifier with its own identifier.

3.1.1. New Software Module

When a radio software provider would like to get approval to use his software on a hardware device of a specific type, he would request authorization from the approval authority for this hardware-software-combination. When granted, the approval authority would compute a digital signature of the software module, including as target the intended hardware model and its manufacturer. Optionally, also an approval number could be added as part of the meta information.

Also other parties could apply for an approval, for example a service provider or device manufacturer who wants to make a reconfiguration software module available for download.

3.1.2. New Hardware Device Type

A new hardware device needs to store the trusted public key of the approval authority. This key is stored on the device by the device manufacturer during manufacturing. It can also be required to store an approval number to ensure unique identification of the device type.

Already existing, older reconfiguration software can be authorized for a new hardware model in the same way as described above in section 3.1.1: The approval authority digitally signs the already existing reconfiguration software, indicating the manufacturer and type of the new hardware model in the meta information. The hardware manufacturer, the software manufacturer, or also an independent party as a service provider or network operator could apply for approval of a hardware-software combination.

3.1.3. Variant for Independent Approval of Hardware and Software

If independent approval of radio hardware and software should be deemed acceptable in the future, also this approach could be realized using signed content as described above. A basic prerequisite to be practical is that different device models support the same reconfiguration software execution environment, enabling that the same software module can be executed on different device types. The only required change to the software authorization scheme would be that the meta information of a signed software module indicating the target device type is used in a different way: Instead of indicating a single target device type, an identifier of the intended target radio execution environment would be used. The reconfigurable device would compare this identifier with the reference identifier of the implemented execution environment. Another possibility would be to use a wildcard expression matching all intended target device models.

The two extremes of independent authorization of radio hardware and radio software on the one side and authorization of each hardware-software-combination can be combined in a two-step solution:

- Software is authorized for an open radio platform
- In addition, a compatibility check of reduced complexity is required for each intended hardware model.

A potential advantage of this combined approach could be that possibly complex and sumptuous checks of the software module against a standardized open radio platform can be combined with efficient compatibility checks to be performed for each hardware device. The combined approach could be implemented only organizationally to make conformance checks more efficient, but it could also be mapped on a technical download solution, allowing that both steps are performed independently. In particular, they could be performed by different entities, so that e.g. only the first step would involve a trusted radio software approval body, while the compatibility check could be performed by a network operator in order to ensure a reliable operation.

4. CONCLUSIONS

Existing security technology implementing security services like encryption, authentication, non-repudiation is on the one hand available and has only to be used, on the other hand developing secure security mechanisms has proven to be challenging to do right, see for example the disastrous insecurity of the WEP encryption mechanism intended to protect wireless communication. So not only for reasons of efficiency, also for reasons of security it is advantageous to use existing security technology as far as possible. This affects not only the very basic cryptographic algorithms themselves, but also security standards built on top of them (as for example digital signature and signed content).

Using a digital signature for secure content download is a well-known security mechanism that can also be used to protect download of radio software. The specifics of radio reconfiguration (software defined radio) do not require new cryptographic mechanisms. Instead it needs to be defined how the known security mechanisms have to be used to implement the policy which shall be followed:

- what needs to be signed (what to include in meta information: e.g. authorized target device(s), software identifier or approval number, region where may be used; further conditions that have to be met for activation of software module)
- who shall be authorized (trusted) to sign a download software module and thereby authorize/approve its use.
- required public key infrastructure
- exact format (e.g. PKCS#7 with RSA signature 1024 bit, use attached or detached signature)

It is important to notice is that the policy to be followed will vary most probably not only with local regulations, but will depend also on the evolvement of regulatory rules, the specific market for which a reconfigurable device is intended, and the underlying business model. Furthermore, it will vary depending on reconfiguration classes that can be used to distinguish the properties that are to be modified respectively defined by the downloaded software module (e.g. relevant for regulatory conformance, relevant for reliable and efficient network operation, relevant for the end user). For example, while applications for a controlled execution environment might by accepted from any source under user decision, low-level radio software modules could only be accepted when approved by the device manufacturer or another trusted approval authority, without the user having a possibility to override this policy. Algorithms for cell selection and medium access could be approved by a network operator to ensure a correct, fair, and efficient operation of the mobile communication system.

ACKNOWLEDGEMENT

This work has been performed in the framework of the EU funded project E^2R. The authors would like to acknowledge the contributions of their colleagues from E^2R consortium.

REFERENCES

[1] IST-2003-507995 E^2R Project, http://www.e2r.motlabs.com/

[2] 3GPP 23.057 “Mobile Execution Environment (MExE)”, V6.2.0, 2003-9.

[3] P. Cook: “Wireless Download Security”, SDRForum Document, 2004-I-0069, 14 June 2004. http://www.sdrforum.org/MTGS/mtg_39_jun04/04_i_0069_v0_00_wireless_security_06_14_04.pdf

[4] Digital Signature Standard (DSS), Federal Information Processing Standards (FIPS) Publication. 186, May 1994. http://www.itl.nist.gov/fipspubs/fip186.htm

[5] R. Housley: “Cryptographic Message Syntax (CMS)”, RFC3369, Aug. 2002.

[6] R. Housley, W. Polk, W. Ford, D. Solo: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile”, RFC3280, April 2002.

[7] IST-2003-507995 E2R Project, http://www.e2r.motlabs.com

[8] B. Kaliski, J. Staddon: “PKCS #1: RSA Cryptography Specifications Version 2.0”, RFC2437, Oct. 1998.

[9] Microsoft MSDN: “Introduction to Code Signing”, Microsoft Authenticode, http://msdn.microsoft.com/library/default.asp?url=/workshop/security/authcode/authenticode.asp

[10] M. Myers, R. Ankney, A. Malpani, S. Galperin, C. Adams: “X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP”, RFC2560, June 1999.

[11] Sun Micros.: Java Tutorial: Signing and Verifying JAR Files – Understanding Signing and Verification, 2005. http://java.sun.com/docs/books/tutorial/jar/sign/intro.html

[12] L. Michael, M. Mihaljevic, S. Haruyama, R. Kohno: “Security Issues for Software Defined Radio – Design of a Secure Download System”, IEICE Trans.Comm, Vol.E85-B No 12, Dec. 2002, pp. 2588–2600.

[13] K. Sakaguchi, C. Fung Lam, T. Doan, M. Togooch, J. Takada, K. Araki: “ACU and RSM Based Radio Spectrum Management for Realization of Flexible Software Defined Radio World”, IEICE Trans. Commun., Vol. E86-B, No. 12, pp. 3417–3424, Dec. 2003.

[14] Sander Siezen: “Symbian OS Version 9.1 functional description”, version 1.1, Feb. 2005. http://www.symbian.com/technology/symbos-v91-det.html

[15] Sun Micros.: “Mobile Information Device Profile, v2.0”, JSR-118, 2002.

[16] Y. Suzuki, K. Oda, R. Hidaka, H. Harada, T. Hamai, T. Yokoi: “Technical Regulation Conformity Evaluation System for Software Defined Radio”, IEICE Trans. Commun., Vol. E86-B, No. 12, pp. 3392–3400, Dec. 2003.