Securing Reconfigurable Terminals – mechanisms and protocols

Stoytcho Gultchev[1], Chris Mitchell[2], Klaus Moessner[1], Rahim Tafazolli[1]

Mobile VCE Teams at: [1]Centre for Communication Systems Research
The University of Surrey, Guildford, Surrey, GU2 7XH, UK
[2]Information Security Group, Royal Holloway, University of London, Egham TW20 0EX, UK

email: {S.Gultchev, K.Moessner, R.Tafazolli}@eim.surrey.ac.uk, C.Mitchell@rhul.ac.uk

ABSTRACT

Software reconfigurability of air interfaces, the actual reconfiguration processes and the procurement of reconfiguration software are posing substantial threats to the system integrity of wireless communication system. These threats are investigated and reported in this paper and mechanisms to ensure secure reconfiguration procedures are described. The security mechanisms documented form part of the MVCE Reconfiguration Management Architecture (RMA).

I. INTRODUCTION

Future multi-function and multi-mode mobile terminals, based on SDR technologies, will consist of individual dynamically configurable components that may use different access technologies (i.e. wireless LAN, cellular or satellite air interfaces or wireline/Bluetooth connections) to download configuration and radio software components that are necessary to reconfigure from one air interface standard (Radio Access Technology - RAT) to another. Reconfigurability will depend on three main enablers:

a) an open programmable software radio platform,
b) downloadable radio configuration software and
c) an architecture to manage reconfiguration processes and to ensure compliance with the radio standards and regulations.

Therefore, reconfigurable terminals will require a secure environment to support their communications and their access to reconfiguration related information within the network. One of the main challenges facing reconfigurability is in understanding the security requirements and potential threats that the heterogeneity in the access networks introduces, encompassing networks on the scale of conventional cellular networks down to personal networks. The security issues arising are considerable, especially given that some of the local, personal traffic may be within a configuration of multiple devices or components belonging to a single user, and reconfiguring such a (multi-device-)'terminal' for that user. This leads to conclusion that many of the applications and services, needed for reconfiguration of the 'terminal', accessed by the user are likely to require additional security measures over and above what is provided by the underlying network infrastructure [1]. This paper provides a solution to these security issues, focusing on the security management tasks of reconfigurable terminals.

II. SECURITY & RECONFIGURABILITY

Software Radio Terminals being able to reconfigure to various/any radio access systems [2], will require robust security mechanisms to protect the mobile terminal from unauthorised access and to ensure the integrity of the internal processes involved in reconfiguration when exchanging information with the network. Different types of reconfigurations of a terminal will lead to different reconfiguration procedures (i.e. the complexity of a reconfiguration process may vary from small adaptation of parameters to a complete reconfiguration of the RAT) but all of them will have at least one common feature – each of these reconfiguration procedures has to implement configurations and to process messages that are passed into or out of the terminal. This includes the exchange of signalling messages between terminal and network, and the download of reconfiguration software modules from network to terminal node.

Reconfigurability of radio terminals intrinsically provides the possibility of unintentionally but also deliberately corrupting the radio environment. To prevent such situations, reconfigurable systems will require security policies and a security entity, able to cope with a number of threats, including:

- unauthorised interception of data (the user may wish to keep the precise configuration of its terminal private, and the transferred software may itself be confidential);
- malicious modification to data (unauthorised changes to software could cause a terminal to be rendered inoperable);
- impersonation of terminal or network (a malicious terminal impersonator may be able to persuade a terminal to use defective software).

To meet these security threats, a number of security services are required:

- confidentiality (to prevent unauthorised interception),
- integrity (to prevent unauthorised modification),
- origin authentication (to detect impersonation).

The security protocol and mechanisms for securing the reconfigurable terminal as part of the *Reconfiguration Management Architecture (RMA)* [3], are described in sections III and IV of this paper.

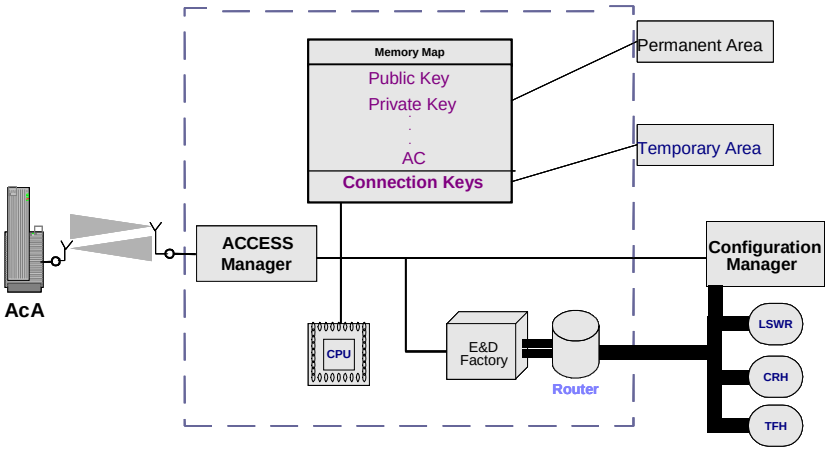

Figure 1 Security Manager block diagram

The RMA is an architecture that provides the infrastructure for reconfiguration of SDR terminals, performs the reconfiguration processes and implements the actual configurations on the reconfigurable radio platform (information can be found in [3-5]). It also provides the required security services and addresses the above-mentioned security threats. To manage and control reconfiguration procedures, the RMA consists of multiple functional parts, which are located within the network and the terminal, respectively. Within the terminal resident part of the RMA, a functional entity implements and provides the security related tasks; it ensures that only authorised access to the reconfiguration management system takes place and that code downloaded had not been corrupted during the download. The functionality of the Security Manager module within the RMA, responsible for establishment, maintenance and termination of the secure connection to the outside world (i.e. the network resident parts of the reconfiguration management architecture) and for the connections within the 'terminal', is described. This Security Manager (SECMAN) is defined as an independent process within the RMA; it requires its own processor and memory space.

III. RECONFIGURATION MANAGEMENT ARCHITECTURE (RMA) – Security Manager

The Security Manager (SECMAN) is responsible for safeguarding access to the functions of the reconfiguration management part within the terminal and also to prevent attempts at fraudulent access from the outside world. Additionally, it provides the required security-related information to the network, stores the security information of the terminal (i.e. public keys and private keys) and the necessary external encryption keys. The SECMAN is responsible for establishing secure connections between the terminal and the network using the security protocols and mechanisms described below.

The Security Manager's internal architecture is depicted in Figure 1. It consists of an 'Access Manager Entity', which deals with the establishment of secure connections between the terminal and the network. Also, it processes the messages between Configuration Manager, the network resident parts of the RMA (i.e. the AcA) and the other functional entities within the Security Manager (see Figure 1). The second module is an 'Encryption & Decryption Factory' (EDF), which implements the security features (as defined in the section IV), the EDF (and its equivalent within the AcA) encrypts both messages and reconfiguration software, before any transmission between terminal and network can take place. For messages and software transmissions originated within the network, the EDF decrypts the streams and passes the data to the 'Router' (i.e. the third functional module within the SECMAN). This router's main task is to direct packets to the various internal entities within the RMA (i.e. to the Configuration Manager, Local Software Repository, Configuration Rule Handler and Tag File Handler, see [4]). The router de-multiplexes the incoming message flow and redirects messages to the appropriate target (as a side effect, the router removes much of the complexity from the EDF, distributes the tasks within the Security Manager and introduces through its modularity additional flexibility to the structure). In summary, the Security Manager performs the following tasks:

- Establishment of secure connections within the RMA;
- Encryption of message and data transfer within the RMA;
- Distributing of reconfiguration messages and software between the modules within the RMA;
- Encryption and decryption of messages and data based on the Security Protocols and Mechanisms;
- Performing the communication/reconfiguration management related signalling with the other modules within the RMA, see [5].

Once the Security Manager receives a request, from the Configuration Manager, to establish a secure connection between the terminal and the network, it follows the 'Security Protocols and Mechanisms' outlined in section IV. It first performs a mutual authentication procedure between terminal part of the RMA and the network part(i.e. the AcA server). The next step is to respond to the Configuration Manager; this response contains a flag indicating whether a successful authentication has taken place and, if so, a connection may subsequently be established. If the network sent the request, a similar procedure has to be performed (however in the opposite direction).

If messages are exchanged between LSWR, CRH, TFH, CM and the network, they are processed and packaged

into a secure frame by the SECMAN and then transmitted to the network. If a packet arrives from the AcA server, the Access Manager authenticates and forwards it to the EDF where the necessary decryption is performed; after that the message is passed to the 'Router' for further distribution to the appropriate functional entity. The secure connection remains open until the Configuration Manager requests its termination.

IV. SECURITY PROTOCOLS AND MECHANISMS

A PKI-based security scheme

We next outline a way in which the three security services described above are provided by the RMA's Security Manager module. This security scheme is PKI (Public Key Infrastructure) based, in that it assumes that all terminals and AcAs are equipped with an asymmetric key pair and a certificate for their public key. This has the advantage that it will enable a terminal and an AcA to secure exchanged data even if they have no prior 'security context'; moreover this can be achieved without any need for on-line communications with third parties. This is likely to be particularly advantageous in the future wireless environment.

This will require each terminal and each AcA to have an associated Certification Authority (CA). This has two immediate consequences:

- Each terminal and each AcA will have a trusted copy of the public signature verification key of their CA;
- Each terminal and each AcA will possess a certificate for their own public key, signed by their CA.

When the terminal and AcA conduct the first phase of the protocol described below, they will need to have a reliable copy of each other's public key. This will typically be achieved by exchanging public key certificates and then verifying them. If the two entities share the same CA then there will be no problem. However, if the two entities use different CAs then they will first need to obtain trusted copies of their respective CA's public key. This can be achieved by the use of *cross-certificates*.

The security protocol

The security solution itself can be divided into two phases. In the first phase the terminal security manager (SECMAN) and the AcA set up a shared secret session key. This key is then used in the second phase to protect the data exchanged between the two parties.

Phase 1 – Session key establishment

We propose the use of a key establishment mechanism standardised in ISO/IEC 11770-3, [6], namely what is specified there as *Key agreement mechanism 5*. Whilst this seems a reasonable approach, this could be replaced by many other protocols with similar properties.

This protocol requires use of a function F: $H \times G \rightarrow G$, with properties as specified in [6]. The scheme also requires the choice of an element $g \in G$, known by all parties. Both entities are required to have a private key agreement key $h \in H$, and a public key agreement key $p = F(h, g)$. That is, we assume that the terminal has private key agreement key $h_T \in H$ and public key agreement key $p_T = F(h_T, g)$, and that the AcA has private key agreement key $h_A \in H$ and public key agreement key $p_A = F(h_A, g)$.

As discussed above, we assume that the terminal and AcA have trusted copies of each other's public key agreement keys, i.e. the terminal has a trusted copy of p_A and the AcA has a trusted copy of p_T. This can be achieved by exchanging public key certificates.

The protocol operates as follows.

1. The terminal randomly and secretly generates a value $r_T \in H$, computes $F(r_T, g)$, and sends $F(r_T, g)$ to AcA.
2. The AcA randomly and secretly generates a value $r_A \in H$, computes $F(r_A, g)$, and sends $F(r_A, g)$ to the terminal.
3. The terminal computes the shared secret key as $w(F(h_T, F(r_A, g)), F(r_T, p_A))$.
4. The AcA computes the same shared secret key as $w(F(r_A, p_T), F(h_A, F(r_T, g)))$.

Note that steps 1 and 2 can be performed in either order (or in parallel). The same remark applies to steps 3 and 4.

Phase 2 – Data protection

As a result of phase 1, the Security Manager and the AcA will share a secret key. This secret key can now be used to protect all data transferred between AcA and terminal. Note that we assume that this secret key is actually used to derive two distinct keys: one for data encryption and one for MACing of data.

Both AcA and Security Manager will need to store a number of security-related variables for the lifetime of the 'security session'. This will include:

- The shared secret keys;
- A counter for traffic sent from terminal to AcA (initialised to zero during phase 1);
- A counter for traffic sent from AcA to terminal (initialised to zero during phase 1).

These variables form the 'security context'.

When a data string is sent from one party to the other the following steps shall be performed.

The data string shall be prefixed with a security header. This shall include the current counter value and a direction indicator (e.g. '0' for AcA to terminal and 1 for terminal to AcA). The counter in the security context shall then be incremented.

The security header and data string shall be input to the MAC function (using the shared secret MACing key). The output MAC shall be adjoined to the data, e.g. as a 'security trailer'.

The entire string resulting from the previous step shall be encrypted using the shared secret encryption key. When a data string is received it shall be processed as follows.

The received string shall be decrypted using the shared secret encryption key. The decrypted string shall be parsed into a security header, a data string, and a MAC value (if this parsing fails then the string shall be rejected).

The recovered security header and data string shall be input to the MAC function (using the shared secret MACing key). The output MAC shall be compared with the received MAC value; if the two values disagree then the received data shall be rejected.

The security header shall be parsed to recover the counter value and the direction indicator. If the direction indicator is incorrect then the data shall be rejected. The counter value shall be compared with the expected counter value in the security context. If the value is smaller than the expected value then the data shall be rejected. If the value is larger than the expected value then a warning shall be generated to indicate that data has been lost. If the data is accepted then the counter in the security context shall then be reset to one larger than the received counter value.

Some suggestions for algorithms

In order to implement the protocol proposed in the previous section, a number of cryptographic algorithms will need to be used. In this section we propose choices for these algorithms.

Signature scheme, as required to sign certificates used in Phase 1. It is suggested that an RSA-based signature scheme can be used, namely the scheme standardised in ISO/IEC 9796-2, [7].

Cryptographic hash-function, required as part of the signature scheme. Almost all signature schemes require the use of a cryptographic hash-function. The scheme in ISO/IEC 9796-2 [7] is no exception. It is suggested that the SHA-1 hash-function is used, as standardised in ISO/IEC 10118-3, [8].

Function F, sets G and H, and special element $g \in G$, as used in Phase 1. It is suggested to use the multiplicative group of integers modulo a large prime (see also Annex B of [6]). This requires the selection of a large prime p, with the property that p-1 has a large prime factor q[1]. The element g is then chosen to be an integer satisfying $1 < g < p$-1 such that g has multiplicative order q modulo p (i.e. $g^q \bmod p = 1$ and $g^s \bmod p \neq 1$ for any $s<q$). We then put

$G = \{g \bmod p, g^2 \bmod p, g^3 \bmod p, \ldots, g^q \bmod p = 1\}$,i.e. $|G| = q$, and $H = \{1, 2, \ldots, q\text{-}1\}$, and hence $|H| = q$-1.

Finally we define $F(h, g) = g^h \bmod p$.

One-way function w, as used in Phase 1. It is suggested that this is implemented simply by concatenating the inputs to the function and applying the hash-function SHA-1 (see above).

1 Here large is user-selectable, but for the purposes of an experimental implementation choosing p to be of the order of 21000 is probably appropriate, with q almost as large.

Encryption function e, as used in Phase 2. It is suggested that the AES block cipher should be used for data encryption, in the CBC mode. AES (Rijndael) will be specified in ISO/IEC 18033-3 [9], and the cipher block chaining mode is specified in ISO/IEC 10116 [10].

MAC function f, as used in phase 2. It is suggested that MAC Algorithm 2, as specified in ISO/IEC 9797-1 [11], is used. The block cipher used as part of this construction should be the AES (Rijndael) algorithm.

IV. CONCLUSIONS

This paper discussed the reconfigurability related security threats in the SDR terminal, and the various security issues, such as where additional security features were required, and solutions to address these threats were proposed and described. It presented the security entity of MVCE's RMA and described the implementation of the RMA – Security Manager module. Finally, Security Protocols and Mechanisms for the Security Manager's functionality were described.

ACKNOWLEDGMENTS

The work presented in this paper has been a part of Core 2 Research Programme of the Virtual Centre of Excellence in Mobile & Personal Communications, Mobile VCE, www.mobilevce.com, whose funding support, is gratefully acknowledged. More detailed technical reports on this research are available to Industrial Members of Mobile VCE.

REFERENCES

[1] IST-2000-25350 project SHAMAN, 26-Feb-2001 http//www.ist-shaman.org

[2] Mitola J, "Software Radio Architecture–Object Oriented Approaches to Wireless Systems Engineering", Wiley-Interscience, ISBN 0-471-38492-5, 2000.

[3] Moessner K, Vahid S, Tafazolli R, "Reconfiguration Management Architectures", UK patent application no. 0028463.8, 22 November 2000.

[4] Moessner K, Gultchev S, Tafazolli R, "Software Defined Radio Reconfiguration Management", PIMRC, San Diego, California, USA, 30 September -04 October 2001.

[5] Gultchev S, Moessner K, Tafazolli R, "Management and Control of Reconfiguration Procedures in Software Radio Terminals", WSR2002, Karlsruhe, Germany, 20-21 March 2002.

[6] ISO/IEC 11770-3: 1999, Information technology – Security techniques – Key management – Part 3: Mechanisms using asymmetric techniques.

[7] ISO/IEC 9796-2: 1996, Information technology – Security techniques – Digital signature schemes giving message recovery – Part 2: Mechanisms using a hash-function.

[8] ISO/IEC 10118-3: 1998, Information technology – Security techniques – Hash-functions – Part 3: Dedicated hash-functions.

[9] ISO/IEC 18033-3 (to appear), Information technology – Security techniques – Encryption algorithms – Part 3: Block ciphers.

[10] ISO/IEC 10116: 1997, Information technology – Security techniques – Modes of operation for an n-bit block cipher.

[11] ISO/IEC 9797-1: 1999, Information technology – Security techniques – Message Authentication Codes (MACs) – Part 1: Mechanisms using a block cipher.