PROTECTION OF DOWNLOADABLE SOFTWARE ON SDR DEVICES

Eimear Gallery (Mobile VCE Research Group, Information Security Group,
Royal Holloway, University of London,
Egham, Surrey TW20 0EX; E.M.Gallery@rhul.ac.uk);
Allan Tomlinson (Mobile VCE Research Group, Information Security Group,
Royal Holloway, University of London,
Egham, Surrey TW20 0EX; Allan.Tomlinson@rhul.ac.uk).

ABSTRACT

This paper addresses the problem of configuring mobile devices over the air. A protocol is described that applies the concepts of trusted computing to allow a mobile host to demonstrate that it is secure, before any software is securely downloaded to it. This enables the source of the software to be given guarantees about the way the software will be handled by the recipient mobile host.

1. INTRODUCTION

It is envisaged that mobile communications systems will soon be sufficiently advanced to make use of SDR techniques. These techniques will be used to reconfigure the air interface, providing the consumer with greater flexibility in the choice of access networks. While the concept of a reconfigurable air interface holds considerable promise, there are several fundamentally important security issues to consider.

The security of downloaded code has been considered by the SDR Forum[1], which has produced a set of requirements for software downloads [6], and described the security considerations for SDR [7]. This paper addresses some of the security issues identified by the SDR forum, and describes a protocol that may be used to meet a number of the requirements identified.

The protocol described in this paper originated from recent research into securing broadcast content delivered to mobile receivers [3]. The solutions developed to address that problem require the secure download of a conditional access application to a mobile receiver. This paper takes these concepts and adapts them specifically to provide secure solutions for SDR.

[1] www.sdrforum.org

2. SECURITY REQUIREMENTS

Security issues arise in relation to both the SDR terminal host on which the software will execute, and the RF reconfiguration software itself. Both the host and the application need to be protected.

Earlier work presented to the SDR forum [2] focused on protecting the SDR terminal from malicious applications through the deployment of a policy-based authorisation framework for implementation within the mobile environment. The objective of this was to provide mobile devices with the ability to assign appropriate privileges to software. The authorisation decision is based on both where the software originates from and the attributes it possesses.

By contrast, the work described in this paper aims to fulfil some of the requirements associated with the SDR application, as defined in [6]. These application-oriented requirements are all associated with the download process.

1. The download process shall employ means, such as encryption, of protecting proprietary radio software and data during download to prevent unauthorised parties from gaining access to or altering this proprietary data or software. It is important to ensure that when this proprietary intellectual property is being downloaded to the SDR device, it is adequately protected from unauthorised access.

2. The software shall be downloaded to a buffer area in the SDR device and verified for integrity and authenticity.

3. The download process shall employ an effective authorisation procedure to verify that the entity requesting the radio software download has the authority to receive, install, and utilise the radio software download.

4. A capability exchange shall take place between the network and the SDR device prior to radio software download to enable the network to select appropriate

software entities and parameter sets for the SDR device. If the network server finds no matching software entities and parameter sets, the download process shall be terminated with a failure message back to the SDR device.

5. Finally, it must be determined whether the configuration of the SDR device remains acceptable for correct operation of the downloaded radio software. If a mismatch occurs, the installation process shall be terminated with feedback to an appropriate network entity.

In the long term, the SDR environment may become more open, with RF configuration software widely and freely available. In the short to medium term, however, it is envisaged that IPR associated with radio software will remain a protected commodity. In the architecture and the associated download protocol proposed here, both the confidentiality and integrity of the software are protected while it is in transit from the software provider to the SDR terminal, and when it is on the terminal itself, including when in storage and when executing.

While SDR will provide greater flexibility in the system, it will still have to remain under some degree of control. In order to comply with user safety controls and regulator guidelines, it may be in the interest of the software or network provider to validate that the device has a specific set of OS controls in place before allowing the release and execution of SDR software, to prevent damage occurring to either the user or the network.

It is also important to ensure that the capabilities reported by the SDR device are indeed accurate, so that the wrong SDR software will not be transmitted to the terminal for execution. Mechanisms fulfilling these requirements will help prevent any legal issues arising with respect to dangerous radio emissions from user devices. In conjunction with this, there are commercial benefits to be gained by providing additional protection to consumers.

In consideration of requirements 3 and 4, although the capability response from the SDR device may represent a specific state at the time of transmission, this is of little help if the terminal configuration can be changed before the downloaded software is installed. If this happens, then software, which may work as intended with the presumed configuration, could cause the terminal to function incorrectly or maliciously. The requirements relating to integrity verification of the SDR terminal are met by our proposed protocols.

It is also clear that the download of malicious code has emerged as one of the most significant security issues surrounding SDR. In the long-term scenario mentioned above, where RF reconfiguration software is freely available, a device may be reconfigured in an ad hoc manner. In this scenario there can be no guarantees that, when a device leaves a controlled network, malicious code will not be downloaded either accidentally or maliciously. Thus, rogue terminals may be created. Such terminals may subsequently request to be reconfigured to rejoin a commercial network. In order to prevent damage that may be caused by such rogue terminals, it is in the operator's interest to ensure that these devices can be detected prior to their access to the required SDR software, and re-connection to the network. In order to achieve this, it must be ensured that only uncompromised terminals are authorised to install the required SDR software.

4. DOWNLOAD PROTOCOL

The protocol we describe to address the above concerns has its basis in trusted platform technology. The Trusted Computing Group[2] (TCG) is an industry forum which is defining standards for trusted platform technology. This forum has a Mobile Phone working group, which is developing trusted computing standards specifically for mobile devices. It is therefore reasonable to expect that future mobile devices will be able to make use of trusted computing technology. The protocol described in this paper protects the downloaded application during transport to, and execution on, the mobile receiver. Trusted platform components specified by the TCG and used in our proposal include the Core Root of Trust for Measurement (CRTM), the Trusted Platform Module (TPM) [9,10,11] and the TPM Software Stack (TSS) [8].

In parallel to the Trusted Computing Group, Microsoft is developing the Next Generation Secure Computing Base (NGSCB) [1,5]. NGSCB mandates the presence of a Security Support Component (SSC) which may be implemented by a TCG version 1.2 compliant TPM. NGSCB also describes modifications to be made to the CPU, memory controller, graphics adaptor, and keyboard in order to facilitate the protected execution of an isolation kernel, which enables several operating systems to execute in parallel on the same machine, and controls access to system resources.

4.1. Application of Trusted Computing to Secure Downloads

Potential issues may arise with a trusted platform that deploys only the mechanisms described by the TCG. The main issue is that there are no mechanisms described by the TCG to partition the system into trusted and untrusted environments, nor is there any mention of trusted operating systems or applications which may exist within these

[2] www.trustedcomputinggroup.org

environments. On the face of it, one could take this to imply that the "protected execution environment" we speak of encompasses *the entire platform*. Consequently, platform use could become very restricted. If the platform state is to be considered trustworthy, the challenger may insist that only a specific operating system and limited set of applications are running.

If platform use is to remain unrestricted, the challenger could be faced with the task of verifying a large set of complex integrity metrics. It is clear from the way PCR values are calculated that verification of an integrity challenge could become a difficult and overly complex task.

A question also arises in relation to the protection of binary objects after they have been decrypted. When an application is decrypted we must consider the threats it may be exposed to during execution, for example tampering or replication.

Finally, there exists the possibility that malicious software may penetrate the system and bypass the operating system kernel via Direct Memory Access (DMA). In this way, applications that have been measured by the CRTM may be overwritten by malicious code. Therefore, during platform attestation, a false impression of the software environment may be given to a challenger.

To overcome these limitations, the protocol proposed in section 4.6 is based on a combination of the security mechanisms offered by NGSCB, LT hardware extensions [4], and the CRTM and TPM as defined by the TCG. This combination of security mechanisms is being proposed here for a number of reasons.

The CRTM and the TPM described in the TCG specifications are well defined, in the public domain, and have undergone a review process leading to the recent release of version 1.2 of the main TPM specification [9,10,11]. They facilitate authenticated boot, secure storage and platform attestation.

Memory protection mechanisms provided by the NGSCB isolation kernel and LT hardware extensions guarantee that the downloaded application can execute without interference and without external monitoring. By implementing an isolation kernel, as described by Microsoft, trusted partitions can be separated from the untrusted system partitions. Thus any legacy operating system and any untrusted applications may still run in parallel to trusted applications. Furthermore, the process of platform attestation or software integrity verification also becomes much simpler. Chipset extensions, such as those described by Intel can prohibit system vulnerabilities caused by physical attacks (DMA) bypassing the integrity measuring mechanisms.

4.2. Notation

S	denotes the SDR application server
TPM	denotes a TPM embedded in the mobile receiver M
A_{SDR}	denotes the SDR application
A_D	denotes a trusted application/agent responsible for the secure download of SDR applications
$Cert_X$	is a public key certificate for entity X
R_X	is a random number issued by entity X
$E_K(Z)$	is the result of the symmetric encryption of data Z using the key K
H(Z)	is a one-way hash function of data Z
$MAC_K(Z)$	is a Message Authentication Code, generated on data Z using key K
X(public)	is the public asymmetric key of X
X(private)	is the private asymmetric key of X
$S_X(Z)$	is the digital signature of data Z computed using entity X's private signature transformation
X||Y	is the result of the concatenation of data items X and Y in that order
Id_X	is the identity of X

4.3. Model

The model under consideration is illustrated in figure 1 and involves three parties: the mobile receiver, M; the SDR application server, S; and the attacker, A.

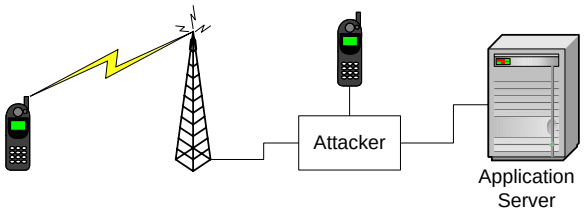


Figure 1: Model

Assumptions about M: We assume the presence of a TPM as described in version 1.2 of the TCG specifications. We also assume the presence of chipset extensions which allow memory protection to prevent physical attacks via DMA, and we assume the presence of an isolation kernel. Thus the mobile receiver M can support multiple execution environments or system partitions.

Within a protected partition, a trusted download application, A_D, executes upon a trusted operating system. It is this protected environment into which the SDR application, A_{SDR}, will be downloaded and executed.

The download application, A_D, will perform two fundamental tasks. It will complete the protocol described below. It will also prevent the potential interference of any other application with A_{SDR} while it is executing. It may, for

example, incorporate a monitoring function which adheres to a specified policy, such that once the conditional access application is running on the device, any attempt by another application to start up will fail; alternatively the start-up of any additional applications will result in A_D stopping A_{SDR}, and erasing it from memory.

We assume that all secret keys required by the mobile receiver are protected by and are only accessible via the TPM.

A unique asymmetric encryption key pair, called the endorsement key pair, is associated with the TPM. The private endorsement key is securely stored in the TPM. The associated public key is certified, and the certificate contains a general description of TPM and its security properties.

Credentials are generated indicating whether the particular design of TPM in a particular class of mobile platform meets specified security requirements, and whether a particular mobile host is an instance of this class of mobile platform and incorporates a specific TPM which meets this design.

A private signing key, an attestation key, is securely stored by the TPM. The public signature verification key corresponding to this private key is certified by a certification authority, CA. The certificate issued, $Cert_{TPM}$, binds the identity of the TPM (the trusted platform containing the module) to a public key used for the verification of digital signatures. This certificate must be obtainable by the software provider S.

The TPM is capable of generating an asymmetric encryption key pair, where the public encryption key is signed/certified using the signature key described above. This thwarts the privacy and security threats surrounding routine use of the public asymmetric encryption key. The private decryption key from this pair is bound to a particular environment configuration state.

Assumptions about S: We assume that the software provider, S, has a private signing key that is securely stored within its environment and that this key is used only for entity authentication. We also assume that S has a certificate, $Cert_S$, issued by the certification authority, CA. This certificate associates the identity of the S with a public key value for the verification of digital signatures. This certificate must be available to, and verified by, the mobile platform, M. Finally, we assume that S is able to verify the claims made by the platform containing a particular TPM regarding the platform's state.

4.4. TPM Commands

We make use of the TPM command set and data structures as specified by the TCG. The $TPM_{CreateWrapKey}$ command is used in step 3 of the public key protocol to instruct the TPM to generate the asymmetric key pair A_D(public) and A_D(private). The input parameters associated with the $TPM_{CreateWrapKey}$ command include information about the key to be created, e.g. key length, and authorisation data necessary to access the parent wrapping key. Encrypted authorisation data for the newly generated key pair may also be input if required.

For this particular use case we require that the key to be created is non-migratable. This implies that the key cannot be migrated from the TPM in which it is created. Alternatively, using a new command described in the TPM version 1.2 specifications, a certifiable migratable key may be created using the $TPM_{CMKCreateKey}$ command. This creates a migratable key which may be certified by the TPM and migrated only under strict controls. We will focus, however, on the case where the key to be created is non-migratable.

In response to the $TPM_{CreateWrapKey}$ command, the TPM returns a *TPM_Key* data structure. This data structure contains A_D(public) and the encrypted private key, A_D(private). The data structure also identifies the operations permitted with the key, and contains flags to indicate whether or not the key is migratable. The data structure may also identify the PCRs to which A_D(private) is bound, and may include the PCR digests at key creation and the PCR digests required for key release. The PCR data provides the integrity metrics, I, used in the protocols.

In our application, S will require that the returned PCR digest at creation reflects a trusted execution environment which consists of a correctly functioning download application running on a trusted operating system. The required PCR digest at release could be inserted by A_D as an input parameter to the $TPM_{CreateWrapKey}$ command. Verification of the returned PCR digest at creation (reflecting a correctly functioning A_D running on the trusted operating system) would ensure that all protocol steps, and the value input by A_D as the digest at release, may be trusted to be correct. The value input into the digest at release by A_D should be checked by S before A_{SDR} is dispatched.

The PCR digest at release may also reflect a state in which a particular operating system and a particular download application alone are running. If this PCR data is communicated to the challenger, S, proof must exist that the data originated from a genuine TPM and that it has not been replayed.

The final part of the *TPM_Key* structure to consider is the *TPM_Auth_Data_Usage* field. This field may take on one of the following values: *TPM_Auth_Never*; *TPM_Auth_Always*; or *TPM_Auth_Priv_Use_Only*. In our scenario, it is A_D that must access the private key to decipher A_{SDR}. The first option is to permit A_D to load the private key without the submission of any authorisation data. In this case the *TPM_Auth_Data_Usage* field is set to *TPM_Auth_Never*. Alternatively, the *TPM_Auth_Data_Usage* field could be set to

TPM_Auth_Always or *TPM_Auth_Priv_Use_Only*, where, on key pair generation, 20 bytes of authorisation data are associated with the public/private key pair, or with just the private decryption key, respectively.

To facilitate this, before a request for key pair generation, A_D could request that the user provides a password, from which the authorisation data for private key use would be derived. Thus, when access to the private decryption key is required, the correct password would have to be re-entered by the user. This option is acceptable provided that user interaction with A_{SDR} is feasible.

Once a key pair has been created using the $TPM_{CreateWrapKey}$ command, the handle associated with this key can be given to the TPM in a $TPM_{CertifyKey}$ command. 160 bits of externally supplied data which, in this protocol, is used to submit a one way hash of R_S, and Id_S may also be given as an input parameter to this command. In response, the TPM returns a *TPM_Certify_Info* data structure. This structure describes the key that was created, including authorisation data requirements, a digest of the public key, and a description of how the PCR data is used. In addition to this structure, the TPM also signs and returns a hash of the public key digest, the 160 bits of external data, and the PCR data contained in *TPM_Certify_Info*, respectively.

4.5. Protocol Initiation

Both protocols begin when the user makes a request for a specific SDR application. It is initially determined whether the required A_{SDR} has previously been downloaded, and is still available in secure storage. If so, the download application, A_D, is called to retrieve A_{SDR} from secure storage and execute the application. If A_{SDR} is not available on the mobile device, then A_D is called to download the application, which can be accomplished by deploying the following protocol. This protocol is completed every time SDR software is to be downloaded. Thus, the asymmetric encryption key pair generated is unique to each protocol run.

4.6. Protocol Description

The protocol is illustrated in Fig.2 and described in the following.

1. $A_D \rightarrow S$: Request for A_{SDR}
2. $S \rightarrow A_D$: R_S
3. $A_D \rightarrow$ TPM: $TPM_{CreateWrapKey}$
4. TPM $\rightarrow A_D$: TPM_{Key}
5. $A_D \rightarrow$ TPM: $TPM_{CertifyKey}$
6. TPM $\rightarrow A_D$: TPM_ Certify_ Info$||$ $S_{TPM}(H(A_D(public)) || H(R_S||Id_S) || I)$
7. $A_D \rightarrow S$: $R_S || Id_S || TPM_{Key} ||$ TPM_ Certify_ Info $|| S_{TPM}(H(A_D(public)) || H(R_S||Id_S) || I)$

S now verifies the signature on the data received, $S_{TPM}(H(A_D(public))||H(R_S||Id_S)||I)$, checks R_S to ensure the message has not been replayed and Id_S to ensure that the message was destined for S. Assuming that the signature, nonce, and Id_S are correct, S then verifies the integrity metrics, I. If I describes a trustworthy platform, then S generates $K1_{S,AD}$ used for data encryption, and $K2_{S,AD}$ used for data integrity protection.

8. $S \rightarrow A_D$: $E_{AD(public)}(K1_{S,AD} || K2_{S,AD}) ||$ $S_S(E_{AD(public)}(K1_{S,AD} || K2_{S,AD})) ||$ $E_{K1S,AD}(MAC_{K2S,AD}(A_{SDR}))$

On receipt of the above message, A_D verifies $S_S(E_{AD(public)}(K1_{S,AD}||K2_{S,AD}))$ and if the signature is valid, instructs the TPM to decipher $E_{AD(public)}(K1_{S,AD}||K2_{S,AD})$. A_D then deciphers $E_{K1S,AD}(MAC_{K2S,AD}(A_{SDR}))$ and verifies $MAC_{K2S,AD}(A_{SDR})$. Once the MAC is verified, the application can be executed. During execution, A_D precludes the potential interference of any other application with A_{SDR}, and after execution A_D deletes A_{SDR}, and all other keys, when they are no longer required. The encrypted copy of A_{SDR} may remain stored for future use, space permitting.

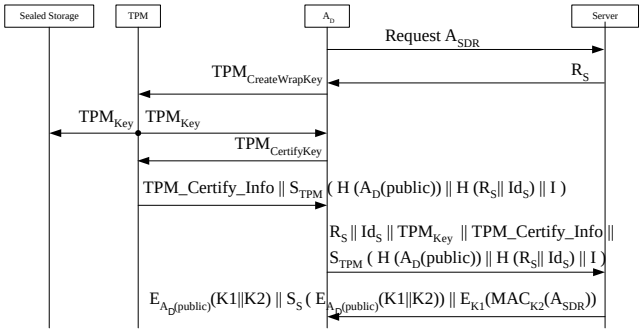


Figure 2: Secure Download Protocol

4.7. Security Analysis

A detailed security analysis of the protocol proposed above, is contained in [3]. In summary, the protocol provides secure transmission and secure execution of A_{SDR}. Secure transmission of A_{SDR} is defined by the following security services which are inherent when using the protocols: confidentiality; integrity; entity authentication; origin authentication; and protection against replay attacks. Secure

execution relates to securing A_{SDR} on the host itself, i.e. through the secure storage and execution of the application. A formal analysis of the above protocol is also given by Delicata in [3].

5. REFERENCES

[1] P. England, B. Lampson, J. Manferdelli, M. Peinado, and B. Willman, "A trusted open platform," *IEEE Computer*, 36(7):55-62, July 2003.

[2] E. Gallery, "A Policy-Based Framework for the Authorisation of Software Downloads in a Mobile Environment," In *2nd Software Defined Radio Technical Conference (SDR03)*, Orlando, Florida, USA, 17-19 November 2003.

[3] E. Gallery, A. Tomlinson, and R. Delicata, "Application of Trusted Computing to Secure Video Broadcasts to Mobile Receivers," Technical Report RHUL-MA-2005-11 (http://www.rhul.ac.uk/mathematics/techreports), Department of Mathematics, Royal Holloway, University of London, June 2005.

[4] Intel, "LaGrande Technology Architectural Overview," Technical Report 252491-001, Intel Corporation, September 2003.

[5] M. Peinado, Y. Chen, P. England, and J. Manferdelli, "NGSCB: A Trusted Open System," In Huaxiong Wang, Josef Pieprzyk, and Vijay Varadharajan, editors, *Proceedings of 9th Astralasian Conference on Information Security and Privacy, ACISP 2004*, volume 3108 of *Lecture Notes in Computer Science*, pages 86-97, Berlin, Germany, July 2004. Springer-Verlag.

[6] SDRF, "Requirements for Radio Software Download for RF Reconfiguration," Technical Report SDRF-02-A-0007-V0.0, SDR Forum, November 2002.

[7] SDRF-DL-SIN, "Security Considerations for Operational Software for Software Defined Radio Devices in a Commercial Domain," Technical Report SDRF-04-A-0010-V0.0 (Ballot Version), SDR Forum, October 2004.

[8] TCG, "TCG Software Stack (TSS) Specification," TCG Specification Version 1.1, The Trusted Computing Group, Portland, OR, USA, August 2003.

[9] TCG, "TPM Main, Part 1 Design Principles," TCG Specification Version 1.2 Revision 62, The Trusted Computing Group, Portland, OR, USA, October 2003.

[10] TCG, "TPM Main, Part 2 TPM Data Structures," TCG Specification Version 1.2 Revision 62, The Trusted Computing Group, Portland, OR, USA, October 2003.

[11] TCG, "TPM Main, Part 3 Commands," TCG Specification Version 1.2 Revision 62, The Trusted Computing Group, Portland, OR, USA, October 2003.

ACKNOWLEDGEMENTS

The work reported in this paper has formed part of the PDE area of the Core 3 Research programme of the Virtual Centre of Excellence in Mobile and Personal Communications, Mobile VCE, www.mobilevce.com, whose funding support, including that of EPSRC, is gratefully acknowledged. Fully detailed technical reports on this research are available to Industrial Members of Mobile VCE.

MOBILE

VCE

MOBILE VCE

November 2005

Protection of downloadable software on SDR devices

Eimear Gallery,
Allan Tomlinson,
Mobile VCE Research Group,
Information Security Group,
Royal Holloway, University of London.

Royal Holloway
University of London

Contents

Motivation for research

Security requirements

- Security requirements and considerations for SDR surrounding the protection of the RF reconfiguration software.

Download protocol

- Key exchange protocol based on trusted computing technologies

Security analysis

MOBILE VCE

www.mobilevce.com

Motivation for research

- It is envisaged that mobile communications systems will soon be sufficiently advanced to make use of SDR techniques.
- These techniques will be used to reconfigure the air interface, providing the consumer with greater flexibility in the choice of access networks.
- While the concept of a reconfigurable air interface holds considerable promise, there are several fundamentally important security issues to consider.

Motivation for research

- Both the host and the application need to be protected.
- Earlier work presented to the SDR forum (Tech conf 2003) focused on protecting the SDR terminal from malicious applications through the deployment of a policy-based authorisation framework for implementation within the mobile environment.
- By contrast, the work described in this paper aims to fulfill some of the requirements associated with the SDR application.

Security requirements

(SDR Tech report: SDRF-02-A-0007-v0.0)

- The download process shall employ means, such as encryption, of protecting proprietary radio software and data during download to prevent unauthorised parties from gaining access to or altering this proprietary data or software. It is important to ensure that when this proprietary intellectual property is being downloaded to the SDR device, it is adequately protected from unauthorised access.

- The software shall be downloaded to a buffer area in the SDR device and verified for integrity and authenticity.

Security requirements

- The download process shall employ an effective authorisation procedure to verify that the entity requesting the radio software download has the authority to receive, install, and utilise the radio software download.
- A capability exchange shall take place between the network and the SDR device prior to radio software download to enable the network to select appropriate software entities and parameter sets for the SDR device. If the network server finds no matching software entities and parameter sets, the download process shall be terminated with a failure message back to the SDR device.

Security requirements

- It must be determined whether the configuration of the SDR device remains acceptable for correct operation of the downloaded radio software. If a mismatch occurs, the installation process shall be terminated with feedback to an appropriate network entity.

Security considerations

Intellectual property rights (IPR) on the device

- Long term: more open environment.
- Medium to short term: IPR associated with radio software.

Safety controls and regulator guidelines

- SDR offers greater flexibility.
- However, operator may wish to validate the device configuration before release of SDR software.

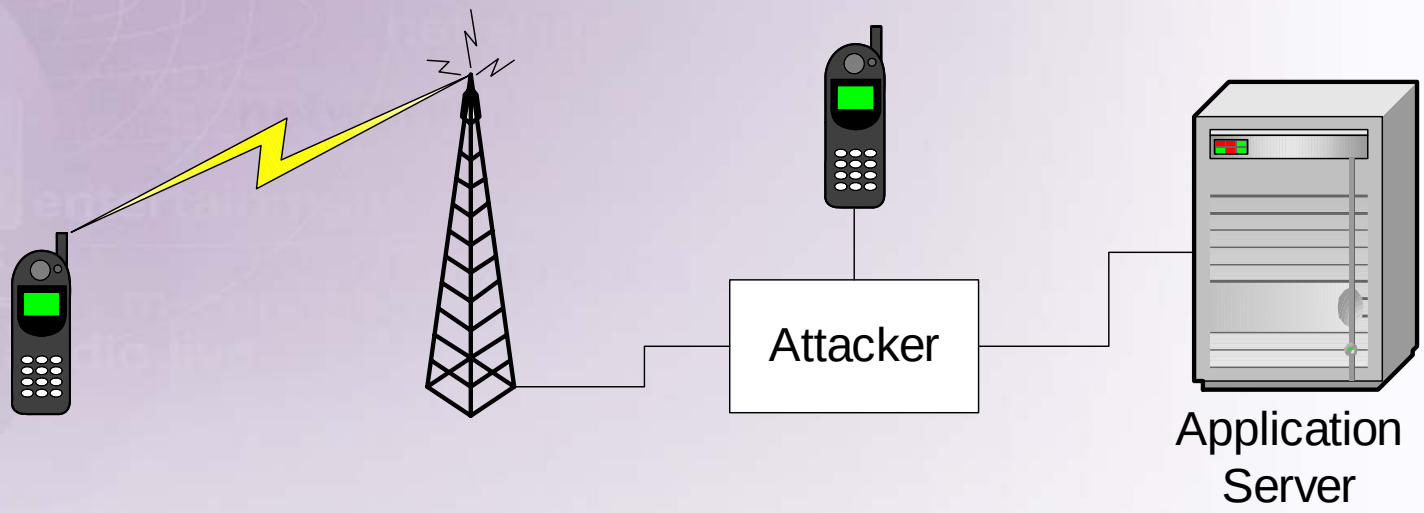
- It is also important to ensure that the capabilities reported by the SDR device are indeed accurate, so that the wrong SDR software will not be transmitted to the terminal for execution.

Security considerations

Malicious code

- Long term: more open environment where SDR software is freely downloadable.
- No guarantees that, when a device leaves a controlled network, malicious code will not be downloaded either accidentally or maliciously.
- Thus, rogue terminals may be created.
- Such terminals may subsequently request to be reconfigured to rejoin a commercial network.
- In order to prevent damage that may be caused by such rogue terminals, it is in the operator's interest to ensure that these devices can be detected prior to their access to the required SDR software, and re-connection to the network (ensure that only uncompromised terminals are authorised to install the required SDR software).

Download protocol: Model



Download protocol: Assumptions (M)

1. We assume the presence of a TPM as described in version 1.2 of the TCG specifications.
2. We also assume the presence of chipset extensions which allow memory protection to prevent physical attacks via DMA, and we assume the presence of an isolation kernel.
3. Thus the mobile receiver M can support multiple execution environments or system partitions.
4. Within a protected partition, a trusted download application, A_D, executes upon a trusted operating system. It is this protected environment into which the SDR application, A_{SDR}, will be downloaded and executed.

Download protocol: Assumptions (M)

5. **The download application, A_D, will perform two fundamental tasks.**
   - It will complete the protocol described below.
   - It will also prevent the potential interference of any other application with A_{SDR} while it is executing.
     - It may, for example, incorporate a monitoring function which adheres to a specified policy, such that once the conditional access application is running on the device, any attempt by another application to start up will fail;
     - Alternatively the start-up of any additional applications will result in A_D stopping A_{SDR}, and erasing it from memory.

Download protocol: Assumptions (S)

1. We assume that the software provider, S, has a private signing key that is securely stored within its environment and that this key is used only for entity authentication.
2. We also assume that S has a certificate, $Cert_S$, issued by a certification authority, CA.
3. This certificate must be obtainable by the mobile receiver.

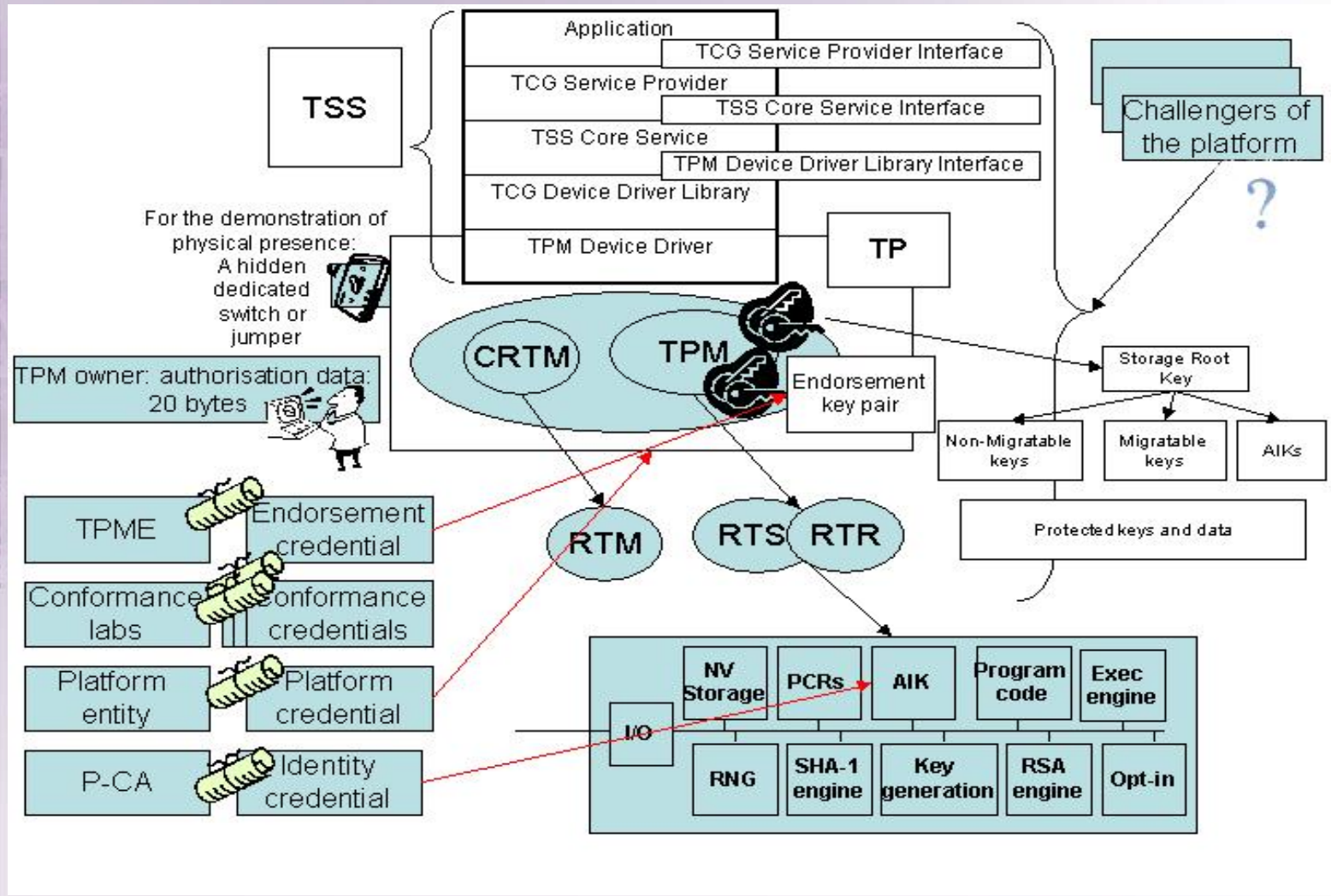
Download protocol: Security services

1. **Confidentiality of the application in transit.**
2. **Integrity of the application in transit.**
3. **Entity authentication:**
   - Host;
   - Application provider.
4. **Origin authentication of SDR application.**
5. **Freshness of messages.**
6. **Confidentiality and Integrity of application in while in storage on the device (AC mechanisms to protect the application on the device).**
7. **Confidentiality and Integrity of application while executing on the device.**

Download protocol: Corresponding security mechanisms

1. **Symmetric encryption.**
2. **MACing of the SDR application.**
3. **Entity authentication protocol runs as described in ISO 9798-3 (Host and application provider)**
   - Attestation (Host) as described within TCG TPM specification set.
4. **Digital signature of the application provider on the symmetric keys used in 1. and 2.**
5. **Nonces/ timestamps.**
6. **Protected/secure storage, as described in TCG TPM v1.2 specification set.**
7. **Memory isolation techniques, as described by Microsoft with respect to their NGSCB.**

Download protocol: TCG Trusted Platform Architecture



Download protocol: Application of TCG Trusted Platform Technology

- All secret keys required by the mobile receiver are protected by and are only accessible via the TPM.
- A unique asymmetric encryption key pair, called the endorsement key pair, is associated with the TPM.
  - The private endorsement key is securely stored in the TPM.
  - The associated public key is certified, and the certificate contains a general description of TPM and its security properties.
- TPM credentials.

Download protocol: Application of TCG Trusted Platform Technology

- A private signing key, an attestation key, is securely stored by the TPM.
  - The public signature verification key corresponding to this private key is certified by a certification authority, CA.
  - The certificate issued, $Cert_{TPM}$, binds the identity of the TPM (the trusted platform containing the module) to a public key used for the verification of digital signatures.
  - This certificate must be obtainable by the software provider S.
- The TPM is capable of generating an asymmetric encryption key pair, where the public encryption key is signed/certified using the signature key described above.
  - This thwarts the privacy and security threats surrounding routine use of the public asymmetric encryption key.
  - The private decryption key from this pair is bound to a particular environment configuration state.

Download protocol: Application of TCG Trusted Platform Technology

Integrity measurements

- Root of trust for measurement (RTM): CRTM : Configuration/Integrity measurements
- Root of trust for storage (RTS): TPM : Configuration/Integrity measurement storage – PCRs
- Root of trust for reporting (RTR): TPM: Reporting of Configuration/Integrity measurements: Attestation
  - current platform configuration

Demonstration of trustworthiness

- Integrity verification mechanism
  - Validation certificates
  - Conformance, endorsement and platform credentials

Download protocol: Application of TCG Trusted Platform Technology

Secure Storage

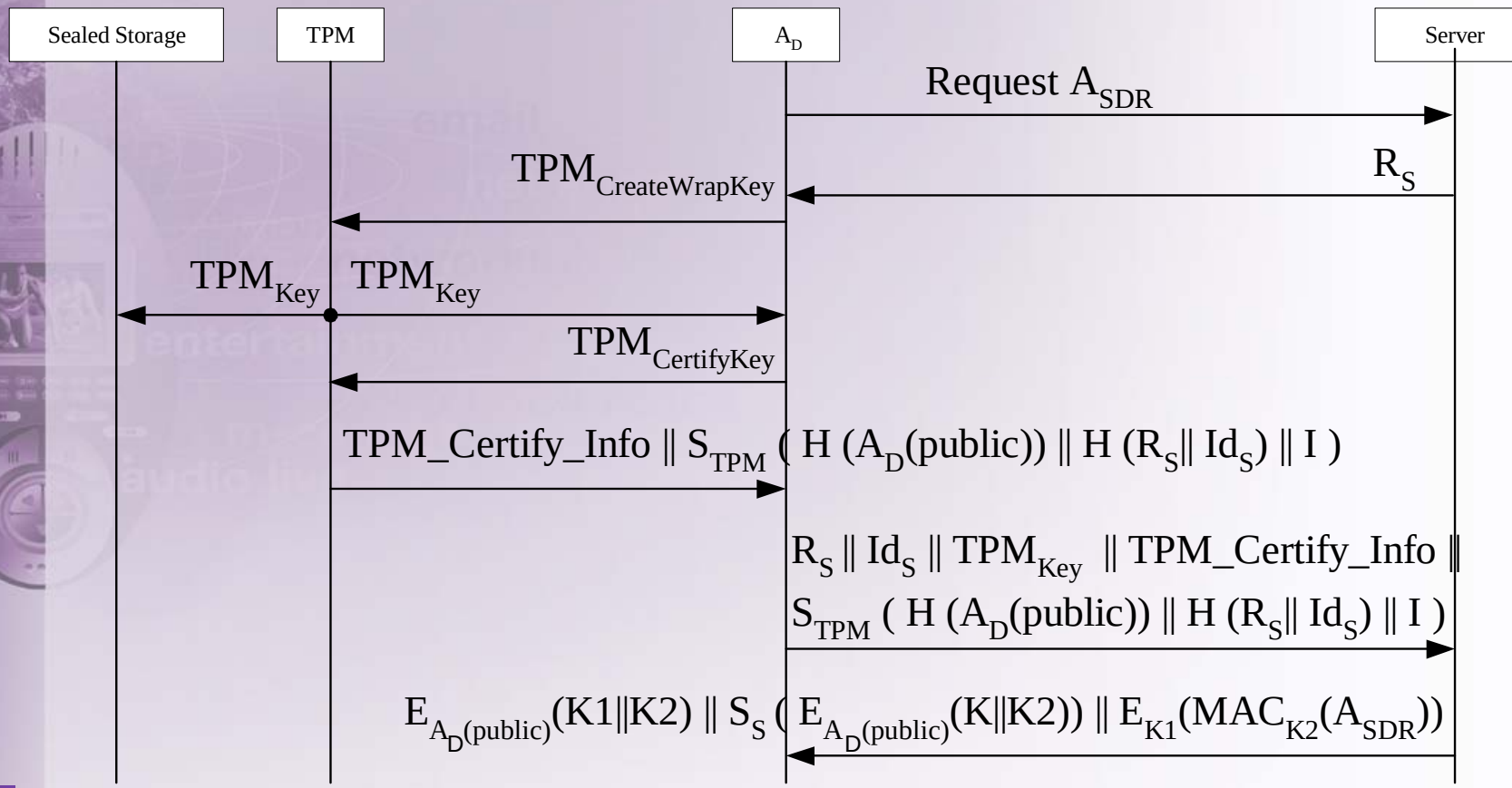
- Key generation
- Binding of keys to specified platform configuration (PCR values) such that a particular key is only accessible and useable when platform is in a specified state

Download protocol: Application of other trusted platform technologies

Secure execution environment

- Memory protection
  - NGSCB, LaGrande (support efficient isolation kernel/machine monitor implementation)
    - Provides benefits with PCR verification.
    - Aids with sealed storage realisation.
    - Protection of application while executing.

Download protocol



Download protocol: Analysis

(RHUL-MA-2005-11 Available: www.rhul.ac.uk/mathemathics/techreports)

Informal analysis

- Completed against the seven security services required of the protocol.

Formal analysis

- Completed by Rob Delicata, University of Surrey.
- In the first instance, the scope of the informal analysis is wider than that of the formal analysis. The informal analysis also serves to present the reasons for the protocol's correctness as opposed to the pleasing, but rather underwhelming, boolean response you get from a formal analysis.

Thank you !

MOBILE VCE

For further information please contact:

Eimear Gallery, Dr. Allan Tomlinson

E-mail: {e.m.gallery,allan.tomlinson}@rhul.ac.uk

Tel: +44 1784 414345

WWW: www.mobilevce.com