Proceedings of the IASTED International Conference
Communication, Network, and Information Security
October 9–11, 2006, MIT, Cambridge, MA, USA
ISBN CD: 0-88986-636-8

INTRODUCING TRUSTED EAP MODULE FOR SECURITY ENHANCEMENT IN WLANs AND VPNs

[1]Pascal Urien, [2]Mesmin Dandjinou, [1]Mohamad Badra

[1]Ecole Nationale Superieure des Telecommunications (ENST)
37/39 rue Dareau 75014 Paris
France
Pascal.Urien@enst.fr, Mohamad.Badra@enst.fr

[2]Universite Polytechnique de Bobo-Dioulasso
Burkina Faso
Msmin.Dandjinou@voila.fr

ABSTRACT
The Extensible Authentication Protocol (EAP) is a kind of *Esperanto* used for access control in various network technologies such as WLAN or VPN. We introduce the *trusted EAP module*, a tamper resistant chip that computes the EAP protocol. Its functional interface is compatible with IETF emerging specifications. We present an open smartcard platform which enables the design of cheap components, both on client and server side; furthermore we describe a management model that remotely modifies embedded credentials and applications. An implementation of a RADIUS server working with EAP server modules is detailed and analyzed. Finally experimental performances are commented and we underline that today EAP modules compute complex protocol like EAP-TLS in less than 5s, and therefore may be deployed in existing networks.

KEY WORDS
Security, Smartcard, PKI, WLAN, EAP, VPN, RADIUS, TLS.

1. Introduction

The Extensible Authentication Protocol (EAP) [11] appeared in 1999, in order to solve authentication issues induced by an increasing number of users who intended to reach their internet service provider (ISP) through modems, dealing with the point to point protocol (PPP) [31]. It was then introduced in the IEEE 802.1x [6] architecture, defining an AAA infrastructure (Authentication, Authorization and Accounting) for wired and wireless networks. The emerging IEEE 802.16e standard [22], which may be understood as a mobile digital line subscriber technology, also uses EAP through the PKM-EAP (Privacy Key Management EAP) protocol [9] in order to authenticate ISP customers. Lastly it can be used to open secure tunnels, like PPTP [3] or IPSEC links, since it is supported by the next generation of the Internet Key Exchange protocol (IKEv2) [13].

In this paper we introduce the trusted EAP module (TEAPM) e.g. a tamper resistant device, typically a smartcard, that computes the EAP protocol. The main idea is to confine critical credentials in a trusted environment, that moreover autonomously processes EAP messages.

This paper is organized according to the following plan. Section 2 recalls some basic features of the EAP architecture. Section 3 proposes a TEAPM functional interface, compatible with IETF emerging specifications. Section 4 outlines main characteristics of smartcards, and recalls principles of classical remote management. Section 5 introduces the TEAPM smartcard, an open java platform that enables the design of cheap components, both for client and server sides. Furthermore it describes a management model for embedded credentials and applications. We detail and analyze a RADIUS [8] server implementation dealing with EAP-TLS [4] server modules. Finally, experimental performances are commented, and we demonstrate that EAP modules may be deployed in today networks.

2. About the EAP protocol

An EAP dialog (see figure 1) occurs between two entities, the Authentication Server (EAP server) and the Peer (EAP Client). The Authentication Server generates requests processed by the Peer, which returns responses. At the end of this exchange these two entities usually calculate a common shared secret (referred as the AAA-key) involved in calculations of various cryptographic materials.

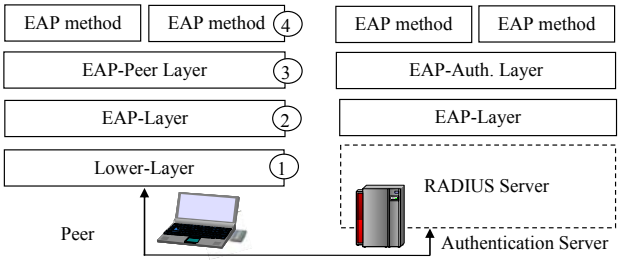

Figure 1. The EAP model

Conceptually [11], an EAP entity consists of four components:

1- The lower layer responsible for transmitting and receiving EAP frames exchanged between the Peer and the Authenticator.

2- The EAP layer that implements duplicate detection and retransmission, and delivers and receives EAP messages to and from the EAP peer and authenticator layers.

3- The EAP peer that acts as a router forwarding EAP packets to the appropriate EAP method identified by a specific byte type field.

4- EAP methods layer that realizes particular authentication scenari.

In next sections we shall introduce a tamper resistant device (the *Trusted EAP module*) that securely runs one or several EAP methods. Although this module is well suited for the Peer side, where it acts as the supplicant's electronic karma, it is also helpful on the Authentication Server side. This aspect is more precisely detailed in section 5.

Figure 2 illustrates TEAPMs deployment in a Wi-Fi context. EAP methods are running in highly secure computing environments, which are not controlled by classical operating systems such as Windows or Linux.

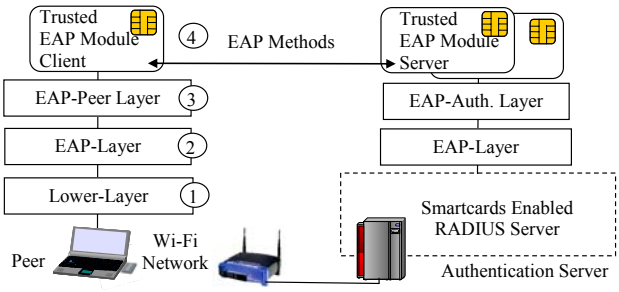

Figure 2. TEAPM deployment in a Wi-Fi context

3. Trusted EAP Module

The trusted EAP module is a tamper resistant device that delivers EAP services in a trusted computing environment. Although this paper is focused on smartcards, other physical implementations could be considered.

3.1 Network services

Network services comprise two kinds of functional interfaces, described in [19] and [23], that we refer as Peer-Layer Interface and Exported-Parameters (see figure 3).

The interface between EAP methods and Peer layer is described in [19] and comprises two main procedures (left part of figure 3):

- methodState() which initializes a method (INIT) or gets its current state (a choice among CONTinue, MAY_CONTinue or DONE)

- eapReqData() which forwards EAP messages to methods, in particular the EAP identity request.

Three variables are internally managed by methods; *ignore* indicates that incoming packet has been dropped; *allowNotifications* indicates that EAP notifications are handled; *decision* gives information about current status (CONDitional_SUCCess, UNCONDitional_SUCCess, FAILure) of authentication scenario. According to this value, an EAP response (eapRespData) may be available. In case of success, the method computes a set of values, whose use is more precisely defined in [23], and which are made available for other EAP layers:

- The Master Session Key (MSK) used as a shared secret, involved in cryptographic material generation, according to various standards [5] [10] [22].

- An additional key, the Extended Master Session Key (EMSK), never shared with a third party.

- Application Master Session Keys (AMSK) introduced by in [26] and obtained with a key distribution function (KDF) using EMSK and other values as input parameters.

- Method-ID used as an unique identifier of an EAP conversation. It's typically obtained by the concatenation of two random values generated by server and client entities.

- Server-ID corresponding to the identity, if any, of the server. For example it's the subject field of an X.509 certificate.

- Peer-ID used for the identity, of the client, if any. It could be the subject field of an X.509 certificate.

- Channel Bindings used as elements of information, typically relative to the IEEE 802.1x access point (Called-Station-Id, , Calling-Station-Id, NAS-Identifier, NAS IP-Address, etc.). They are optionally mirrored during an EAP session, from server to client.

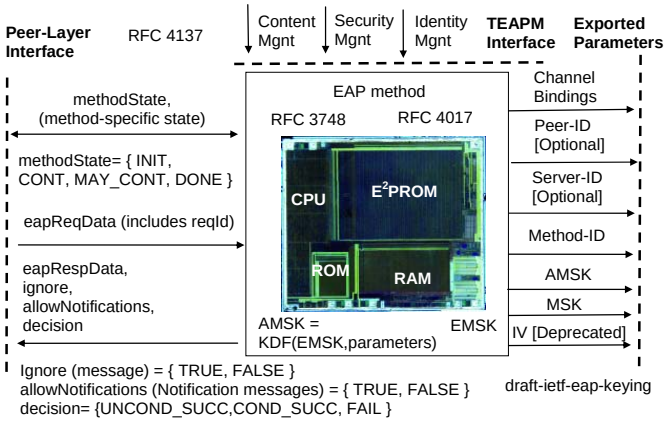

Figure 3. The Trusted EAP module architecture

3.2 Other services

A TEAPM device has a physical interface with the EAP-Peer layer and produces output values as described in the previous section. In this paper we define TEAPM services as applications executed in a (JAVA) smartcard which offers a highly secure computing environment.

However other management services are required for TEAPM practical deployment:

- Content Management. It's the set of operations needed to download credentials, required by a given method (X.509 certificates, cryptographic key, …).

- Security Management. This service manages mechanisms (PIN code, biometric techniques, …) that restrict TEAPM use to authorized users.

- Identity Management. When several methods are available, this service allows to select one of them.

These aspects are more precisely detailed in [21].

4. Smartcard Overview

4.1Smartcard technology

Smartcard is a tamper resistant chip. As an illustration, many governments enhance their identity protection and verification by deploying ID-cards [16] or e-Passports [14], including such a component. It comprises a CPU, often a crypto-processor, and various kind of memories: ROM storing the operating system code, RAM used for stacking operations, and non volatile memory (E^2PROM, FLASH, …) acting as a mass storage (see figure 4). Security is enforced by various hardware and software countermeasures controlled by the embedded operating system. As an illustration internal data bus is encrypted in order to avoid eavesdropping, or cryptographic calculations are protected against differential power attacks. In classical ISO 7816 smartcards [1], commands are carried through a serial link, whose data rate may reach 230,000 bauds; but other communication interfaces are also available such as USB [18] (with a throughput of 12 Mbits/s) and MMC (Multi Media Card).

Smartcards work according to a remote procedure call (RPC) paradigm. They process requests (named APDUs) and return responses, whose size are typically less than 256 bytes. Binary encoding rules are precisely defined in the ISO 7816-4 standard. Furthermore, in most cases, these devices run a Java Virtual Machine, and securely execute applications, e.g. software written with a subpart of the well known Java language.

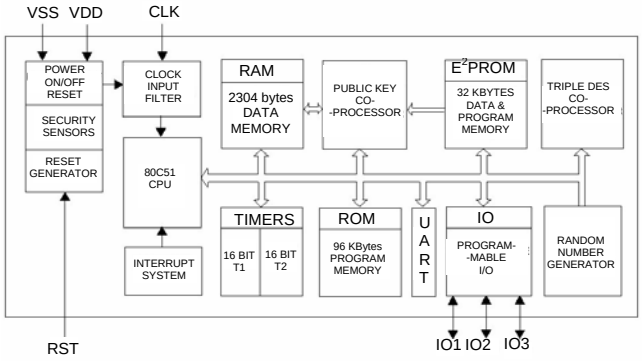

Figure 4. Internal structure of a classical smartcard chip

In 2005, 1,8 billion of smartcards were produced and about 70% of them in the form of SIM [29] modules, that enforce security procedures for GSM phones. Most SIMs include a Java Virtual Machine and are remotely managed by mobile operators.

4.2 SIM remote management

The basic idea behind SIM remote management is to shuttle meaningful commands ([1] ISO 7816-4 APDUs) thanks to the popular Short Message Service (SMS). A SMS message [27] is made of two part: a header and a payload. According to [28] SIM commands are transported in one or several concatenated SMSs. This standard also provides classical security services, such as data privacy (symmetric keys, KIC are used for command encryption/decryption) and information integrity (message authentication codes are computed/checked according to symmetric KID keys).

The remote management of a SIM module is divided in two classes of operations, remote access to files and applet management. So:

- In accordance with [1], SIM files are hierarchically organized according to a tree scheme including a root

repertory (Master File), sub repertories (Dedicated Files) and files. Files operations are controlled by a small set of commands (SELECT FILE, READ, WRITE).

- Applet management includes the opportunity to load, install and remove embedded applications. For that, three main commands are needed: LOAD, INSTALL and DELETE.

These procedures are more precisely defined by [7]. Usually a loading session is accomplished in two steps. Firstly a mutual authentication is performed between a Security Domain entity stored in the SIM, and the remote management entity; two cryptographic keys (S-ENC and S-MAC) are computed at the end of this operation. Secondly an application is downloaded under the control of the Security Domain; S-ENC is used for data encryption and S-MAC for MAC calculation.

5. The TEAPM Smartcard

The TEAPM smartcard is described by an internet draft [21]. All services previously mentioned (network interface, content management, security management, identity management) are accomplished via a set of APDUs [1] defined in this document. Example of performances are given, later in this section.

5.1 OpenEapSmartcard

As we underlined it before, we believe that TEAPM application is a secure embedded software. In order to make it widely available, we introduced the OpenEapSmartcard (see figure 5) platform [12] [17] [20] [24] [25] whose target is to release an open environment, working with most of commercial devices. The standard Java Card Forum (JCF) framework supports a cryptographic package that includes cryptographic resources (Random Number Generator, SHA1, MD5, RSA, DES, 3xDES, AES, etc.) required by classical authentication protocols (TLS, AKA, …).

The OpenEapSmartcard application comprises four java components:

1- *The EapEngine*. It manages several methods and/or multiple instances of the same one. It implements the EAP core, and acts as a router that sends and receives packets to/from authentication methods. At the end of authentication process, each method computes a key (the Master Session Key, MSK) which is read by the terminal operating system.

2- *The Authentication interface*. This component defines all services that are mandatory in EAP methods, in order to collaborate with the EapEngine. The two main functions are Init() and Process-Eap(). The first one initializes methods and returns an Authentication interface; the second function processes incoming EAP packets.

3- *Credential objects*. Each method is associated to a Credential object that encapsulates all information required for processing a given authentication scenario.

4- *Methods*. Each authentication scenario is processed by a specific method class. Once initialized, it analyses each incoming EAP request and delivers corresponding response.

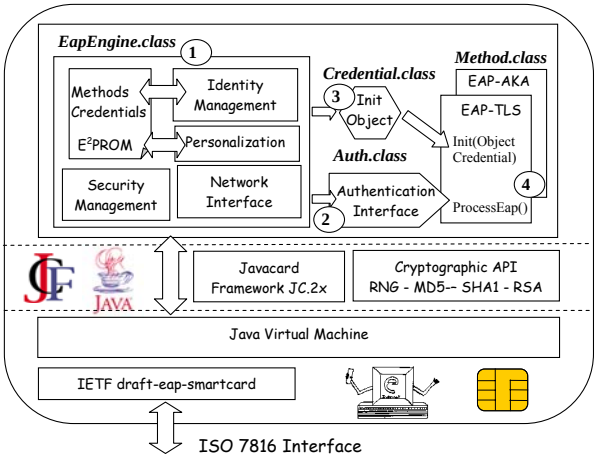

Figure 5. OpenEapSmartcard

5.2 Example of TEAPM deployment

Figure 6 presents a Wi-Fi infrastructure working with TEAPMs (client and server). We give a short description of the operations that take place during the client's authentication.

1- A client's PC, that intends to access to resources offered by the wireless network, periodically transmits an EAP-Start frame in order to initiate an EAP authentication scenario.

2- The access point then produces and EAP identity request that is forwarded by the PC operating system to an EAP smartcard. This device delivers an EAP identity response.

3- This response is encapsulated by the AP in a RADIUS [8] packet, and sent to the RADIUS server. The RADIUS software checks the availability of a smartcard EAP server, and forwards the EAP identity response, that begins a new session.

4- The EAP server returns an EAP request message, encapsulated again in a RADIUS packet sent to the AP, and then to the user's computer. This request is then forwarded to the client's smartcard, which processes the message and outputs a response, sent again to the EAP server smartcard.

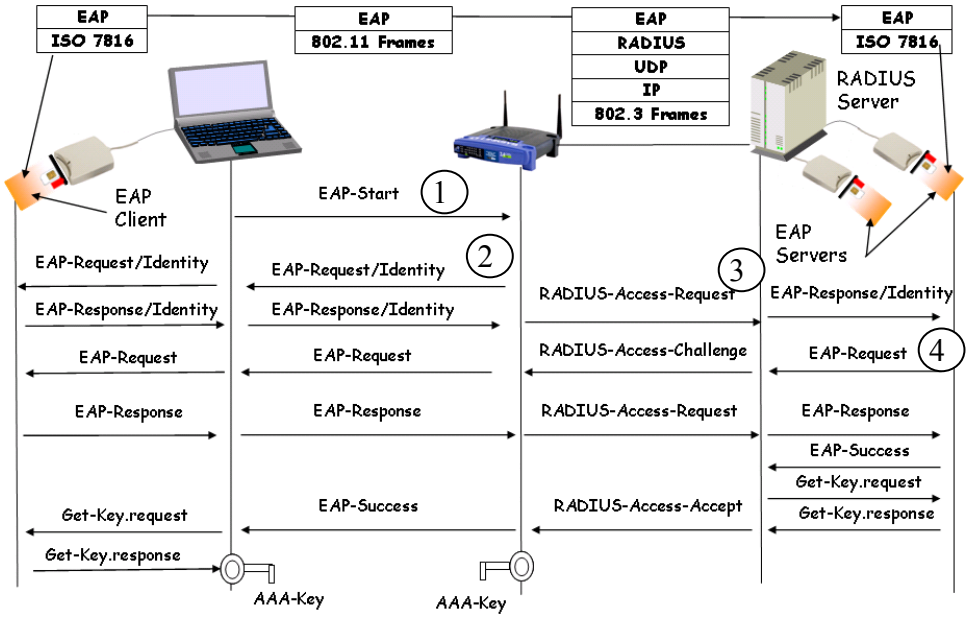

Figure 6. TEAPM deployment illustration

As illustrated by figure 6, an authentication dialog is a set of requests and responses exchanged between EAP server and EAP client. At the end of this process, the server produces an EAP-Success message, as a result the RADIUS entity gets a master cryptographic key (MSK) needed for all security operations between the AP and the client's PC.

The MSK key is securely pushed to the AP, in conjunction with the EAP-Success message, which is forwarded again to the client.

Upon reception of this event, the client retrieves the MSK key from its EAP smartcard. Everything is now ready for computing all cryptographic elements, consumed by radio security protocols such as WEP [5] or IEEE 802.11i [10].

5.3 Performances

They are some timing constraints, the delay between EAP requests and responses must be less than 30 s [6]. Furthermore the total authentication duration must be less than 60 s, in order to avoid the client's network interface reset, generated by the DHCP timeout.

With commercial javacards devices, we observed performances of about 5 s for clients or servers; a detailed analysis is described in [24]. These figures are compatible with previously mentioned requirements, and therefore our platform may be deployed in today network.

5.4 Remote Management

As previously mentioned, remote management of smartcards is a classical feature in today GSM networks and facilitates data and application administration. In a TEAPM context, management functions (see figure 7) could realize the following services:

- *Cancellation of credentials*, such as X509 certificates and associated private keys. Smartcards cloning is extremely difficult, in consequence there is only one physical instance of these entities. The ability to remotely block their use, is an important security requirement, in a distributed PKI environment.

- *Updating of credentials*. There is a need to guaranty continuity or extension of customer subscriptions. This demand is fulfilled by replacing or adding information elements that control services availability.

- *Downloading of new applications*. Authentication protocols may evolve and include new functionalities. In that case, the software is transparently updated, e.g. without TEAPM bearers interaction.

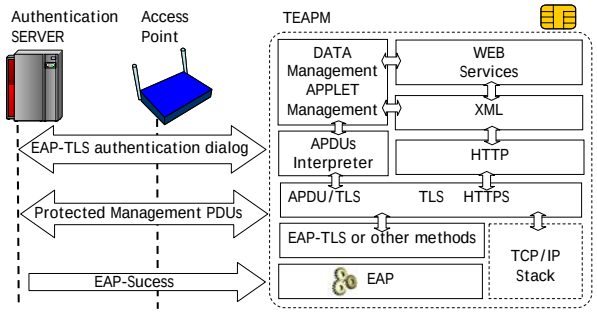

Figure 7. TEAPM remote management

These useful functionalities require a protected data transfer between the administration server and the TEAPM. Most of authentication methods establish secure links that may transport ciphered and signed messages.

Thus, the record layer of EAP-TLS [2] may shuttle encrypted AVPs (*Attribute Value Pair*), and EAP-AKA [30] can transport ciphered payload. Although the DHCP protocol induces timing constraints (usually 60 s) during the first authentication occurrence, the re-authentication process, as defined in [6], doesn't suffer from this issue, and therefore is a practical mean for refreshing information embedded in a tamper resistant device.

The legacy syntax used for management of classical smartcards is based on APDU encoding. Because all TEAPM functional interfaces are described as a set of APDUs, our current TEAPM administration works with an APDUs interpreter (see figure 5), which parses and executes incoming commands. However, other techniques are also suitable and could avoid the deployment of unusual management tools; for that we propose two potential candidates:

- *LDAP* (Lightweight Directory Access Protocol) [32]. The TEAPM content is interpreted as an entry belonging to a portable directory. Management entities act as LDAP clients which create, update or delete attributes by means of messages coded according to the ASN.1 syntax, and typically encapsulated in TLS packets.

- *WEB services*. New generation of smartcards, for example the *Trusted Personal Device* introduced by the European project Inspired [35], offers sufficient resources, in terms of computing capacities and memory sizes, in order to support a TCP/IP stack [18] and the HTTP protocol. Such devices may include WEB server and additional XML facilities (see figure 8), that enable the remote control of embedded resources through WEB services interfaces. An early form of these services could be realized by a specific DTD, translating a collection of APDUs in an XML document.

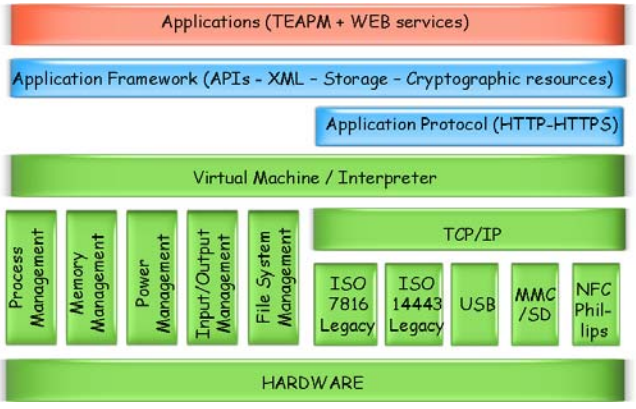

Figure 8. The Trusted Personal Device project

5.5 EAP-server design

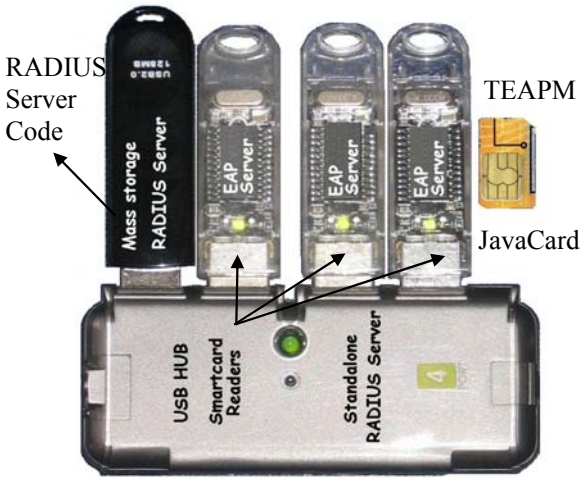

Figure 9. Practical realization of TEAPM servers

EAP servers are dual forms of the EAP clients. They process EAP responses and return EAP requests, while clients do the opposite. These components may be included in various classes of servers, such as access server (authentication in PPP), IKE server (authentication in VPN), or RADIUS server (authentication in Wi-Fi infrastructure). They enhance the security by processing the EAP protocol in a highly trusted computing environment, that equally holds cryptographic credentials. Even if the server operating system is compromised, critical data like RSA keys resist to eavesdropping. In that case, and in similarly to TEAPM clients, the EAP server can be remotely deactivated thanks to management facilities. Furthermore, when client's identity is hidden by cryptographic shields, they may secretly check this parameter and notify the success of an authentication without divulging the true identity of network user [37]. This property facilitates PET (Privacy Enhancing Technologies) deployment in emerging pervasive networks.

In an IEEE 802.1x [6] infrastructure, EAP messages are transported between a network access server (NAS) and an authentication server by the RADIUS protocol. The security of this transport is based on a secret (the RADIUS secret) shared between the AP and the RADIUS server. In most cases the physical security of an access point is difficult to evaluate; the secret is permanently stored in the AP, and perhaps without any cryptographic protection. For that reason, we designed a RADIUS server (see figure 9), running in a personal computer (what means that RADIUS security is controlled by the computer which also holds the RADIUS secret), but that delegates all EAP operations to EAP servers (see figure 10). Each of them processes the EAP-TLS protocol associated to an unique X.509 certificate and its RSA secret key. These choices greatly simplify the software design. More details may be found in [36].

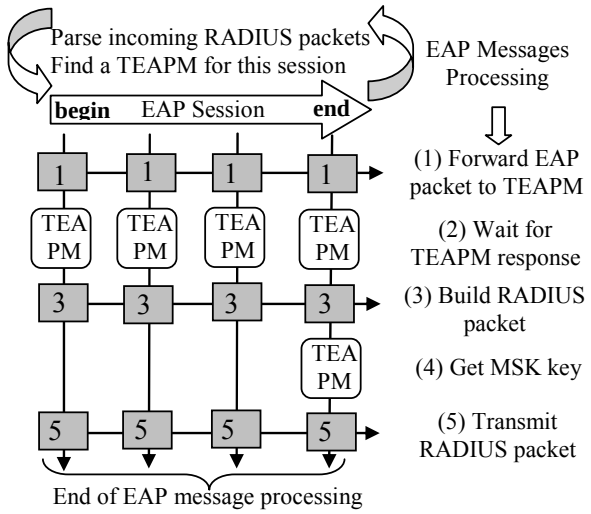

Figure 10. logical management of TEAPMs

Each EAP session is uniquely identified by the concatenation of two values, the NAS-Identifier (RADIUS attribute n°32 in [33]) and the Calling-Station-Id (the client's MAC address, corresponding to RADIUS attribute n°31 in [33]) as follows:

Session-Id = NAS-Identifier | Calling-Station-Id.

When a new session is started (reception of a RADIUS packet containing an EAP-Identity.response indication) the RADIUS server software attempts to find a free EAP-Server which will manage this session until its end. If no device is available the RADIUS packet is silently discarded.

EAP message processing is rather slow, and may require a few seconds; while a complete EAP-Session typically costs 5 s of TEAPM time. Therefore, each EAP packet is handled by a software thread, that waits for TEAPM response, which is afterwards encapsulated in a RADIUS message and transmitted to the right access point (see figure 10).

When no EAP server is available, the incoming RADIUS packet is silently discarded. This mechanism is similar to the classical blocking algorithm used in circuit-switching, where an incoming call is ignored if no output trunks are available. Under the hypothesis the system is working according to an M/M/c/c queue, it follows the Erlang-B formula:

$$p_c = \frac{(\lambda/\mu)^c}{c!}\left[\sum_{k=0}^{c}\frac{(\lambda/\mu)^k}{k!}\right]^{-1}$$

where

- pc is the probability of blocking (e.g. a RADIUS packet is silently discarded),
- c is the number of smartcards (EAP servers),
- λ is the rate of authentication sessions, and
- $1/\mu$ the mean time of an authentication session.

With our best couple of devices (client and server), we measure an authentication duration of about 5 + 5 = 10 s; therefore $1/\mu = 10$. Let's assume a network with 1000 users, with an authentication session every hour, then we deduce:

$\lambda = 1000 / 3600$, and $\lambda/\mu = 10 \times 1000/3600 = 2{,}8$.

The probability of blocking (pc) is about 50% with 2 smartcards (c = 2) and only 1% with 8 smartcards (c = 8).

6. Conclusion and Future Work

In this paper we have presented a new class of cheap authentication modules dealing with WLAN and VPN technologies. We believe that they will improve the security by first of all computing critical protocols in a trusted computing platform, and secondly introducing useful management services.

In a not too distant future, our works will address the TEAPMs management, by means of WEB services facilities. We plan to develop TEAPM applications with the next generation of smartcards, such as the *Trusted Personal Device* [35], which includes TCP/IP facilities and multiple Virtual Machines, providing software tools dedicated to XML processing.

Furthermore, as the memory sizes of today tamper resistant device are dramatically increasing, and are reaching the gigabyte horizon [34], future TEAPMs could manage a large amount of information.

References

[1] ISO 7816, "Cards Identification - Integrated Circuit Cards with Contacts".
[2] RFC 2246, "The TLS Protocol Version 1.0", January 1999.
[3] RFC 2637, "Point-to-Point Tunnelling Protocol (PPTP)", July 1999.
[4] RFC 2716, "PPP EAP TLS Authentication Protocol". October 1999.
[5] Institute of Electrical and Electronics Engineers, "Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications", IEEE Standard 802.11, 1999.
[6] Institute of Electrical and Electronics Engineers, "Local and Metropolitan Area Networks: Port-Based Network Access Control", IEEE Standard 802.1X, September 2001.
[7] GP 2.1.1, "Global Platform Card Specification Version 2.1.1", March 2003.

[8] RFC 3579, "RADIUS (Remote Authentication Dial In User Service) Support For Extensible Authentication Protocol (EAP)", September 2003.
[9] Institute of Electrical and Electronics Engineers, "IEEE Standard for Local and Metropolitan Area Networks, part 16, Air Interface for Fixed Broadband Wireless Access Systems", IEEE Standard 802.16, 2004.
[10] Institute of Electrical and Electronics Engineers, "Supplement to Standard for Telecommunications and Information Exchange Between Systems - LAN/MAN Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specification for Enhanced Security", IEEE standard 802.11i, 2004.
[11] RFC 3748, "Extensible Authentication Protocol, (EAP)", B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, H. Levkowetz, Ed., June 2004.
[12] Urien, P, Badra M, Dandjinou, M, "EAP-TLS Smartcards, from Dream to Reality", 4th Workshop on Applications and Services in Wireless Networks, ASWN'2004, Boston University, Boston, Massachusetts, USA, August 8-11, 2004.
[13] RFC 4306, "Internet Key Exchange (IKEv2) Protocol", C. Kaufman, December 2005.
[14] Libon, O, Vandendooren, H, "The Belgian BELPIC project, a double approach: e-Government & Smartcard", International Meeting 2004, Sophia Antipolis, France, September 22-24, 2004.
[15] RFC 4017, "Extensible Authentication Protocol (EAP) Method Requirements for Wireless LANs", March 2005.
[16] National Institute of Standards and Technology, "Interfaces for Personal Identity Verification", NIST Special Publication 800-73, April 2005.
[17] Urien, P, Dandjinou, M, "The OpenEapSmartcard project", short paper, Applied Cryptography and Network Security 2005, ANCS 2005, Columbia University, New York, USA, June 7-10, 2005.
[18] Tual, J.P, Couchard, A, Sourgen, K, "USB Full Speed enabled smart cards for Consumer Electronics applications", Consumer Electronics, ISCE 2005, Proceedings of the Ninth International Symposium on 14-16 June 2005, pp.230-236.
[19] RFC 4137, "State Machines for Extensible Authentication Protocol (EAP) Peer and Authenticator", August 2005.
[20] Urien, P, Dandjinou, M, Badra, M, "Introducing micro-authentication servers in emerging pervasive environments", IADIS International Conference WWW/Internet 2005, Lisbon, Portugal, October 19-22, 2005.
[21] Internet draft, "EAP-Support in Smartcard", draft-eap-smartcard-09.txt, October 2005.
[22] Institute of Electrical and Electronics Engineers, "Draft IEEE Standard for Local and metropolitan area networks part 16: Air Interface for Fixed and Mobile Broadband Wireless Access Systems Amendment for Physical and Medium Access Control Layers for Combined Fixed and Mobile Operation in Licensed Bands", IEEE 802.16e/D12, October 2005.
[23] Internet draft, "Extensible Authentication Protocol (EAP) Key Management, Framework", draft-ietf-eap-keying-09.txt, January 2006.
[24] Urien, P, Dandjinou, M, "Designing smartcards for emerging wireless networks", Seventh Smart Card Research and Advanced Application IFIP Conference, CARDIS 2006, Tarragona-Catalonia, April 19-21, 2006.
[25] OpenEapSmartcard, http://www.enst.fr/~urien/openeapsmartcard.
[26] Internet draft, "Extensible Authentication Protocol (EAP) Key Management Extensions", draft-aboba-eap-keying-extens-00.txt, April 2005.
[27] GSM 03.40, "Technical realization of the Short Message Service (SMS) Point-to-Point (PP)", ETSI.
[28] TS 03.48, "Security Mechanisms for the SIM application toolkit", 3GPP.
[29] GSM 11.11, "Specification of the Subscriber Identity Module - Mobile Equipment (SIM - ME) interface", ETSI.
[30] RFC 4187, "Extensible Authentication Protocol Method for 3rd Generation Authentication and Key Agreement (EAP-AKA)", January 2006.
[31] RFC 1661, "The Point-to-Point Protocol (PPP)", STD 51, July 1994.
[32] RFC 2251, "Lightweight Directory Access Protocol (v3)", December 1997.
[33] RFC 2865, "Remote Authentication Dial In User Service (RADIUS)", June 2000.
[34] MegaSIM, http://www.m-systems.com/
[35] FP6-IST, "Integrated Secure Platform for Interactive Personal Devices", INSPIRED project, Information Society Technologies, 2004-2006.
[36] Urien P., Dandjinou M., "Introducing Smartcard Enabled RADIUS Server", CTS 2006, the 2006 International Symposium on Collaborative Technologies and Systems, Las Vegas, USA.
[37] Urien, P, Badra, M. "Secure Access Modules for Identity Protection Over the EAP-TLS. Smartcard benefits for user anonymity in wireless infrastructures.", Secrypt 2006, the International Conference on Security and Cryptography, Setubal Portugal, 7-10 August 2006