Project Number	AC095
Project Title	ASPeCT: Advanced Security for Personal Communications Technologies
Deliverable Type	Intermediate
Security Class	Public

Deliverable Number	D14
Title of Deliverable	Trusted third parties: evaluation report
Nature of the Deliverable	Report
Document reference	AC095/KUL/W23/DS/P/14/1
Contributing WPs	WP2.3
Contractual Date of Delivery	May 1997 (Y03/M03)
Actual Date of Delivery	23 June 1997
Editor	Keith Martin

Abstract	This document provides a report on the evaluation of the initial ASPeCT end-to-end security service demonstration involving trusted third parties.

Keywords	ACTS, ASPeCT, TTP, security, integrity, key, escrow, recovery

1. Executive summary

Deliverable D14 is the latest deliverable produced by ASPeCT Work Package 2.3 (WP2.3). This Work Package is concerned with the research and development of appropriate measures to meet the needs for security and protection in future systems for mobile telecommunications services, and more specifically UMTS. In particular, WP2.3 is concerned with the use of Trusted Third Parties (TTPs) in providing such services.

The initial ASPeCT demonstration is of a working TTP service, with generation and distribution of key material to support protected communications between two users in different domains. This demonstration was specified in ASPeCT Deliverable D07 [D07] and a demonstrable version was provided in ASPeCT Deliverable D09 [D09]. Deliverable D14, provides an evaluation of this demonstration.

The **objective** of Deliverable D14 is to review and assess feedback from users and other parties on the initial end-to-end security service demonstration.

NOTE: The following **limitations and restrictions** apply. The **public** nature of this deliverable is restricted to this document.

No general rights to the programmes and the libraries which constitute the demonstrator/prototype are given or implied. Certain components may be

- proprietary,
- subject to non-disclosure agreements,
- claimed and acknowledged as background material,
- subject to governmental controls on export or re-export.

Specific enquiries or requests for clarification may be addressed, in the first instance, to the editor.

2. Document control

2.1 Document history

Version 0: 14th May (not issued)
Version 1: 9th June (first draft)
Version 2: 16th June (second draft)
Version 3: 18th June (third draft)

2.2 Changes Forecast

No changes are forecast to this document unless errors are noted subsequent to delivery.

2.3 Change Control

In conformance with the ASPeCT Quality Plan.

3. Document Cross References

[D02]	ASPeCT Deliverable D02 - Initial Report on Security Requirements. Ref. AC095/ATEA/W21/DS/P/011/B
[D07]	ASPeCT Deliverable D07 - Security Services: First Specification. Ref. AC095/RHUL/W23/DS/I/07/1
[D09]	ASPeCT Deliverable D09 - Trusted Third Parties: First Implementation. Ref. AC095/RHUL/W23/DS/I/09/1
[D11]	ASPeCT Deliverable D11 - Trusted Third Parties: Limiting Smart Card Constraints. Ref. AC095/GD/W24/DS/P/11/1
[AR97]	R J Anderson and M Roe, The GCHQ protocol and its problems. Advances in Cryptology - EUROCRYPT '97, Lecture Notes in Comput. Sci. 1233:134-148, 1997.
[CGM96]	L Chen, D Gollman and C J Mitchell, Key escrow in mutually mistrusting domains, Proceedings of 1996 Cambridge Workshop on Security Protocols, Lecture Notes in Comput. Sci., 1189:139-153, 1996.
[CM96]	L Chen and C J Mitchell, Key escrow in multiple domains. Pre-print, 1996.
[D95]	D E Denning, Critical factors of key escrow encryption systems, Proceedings of the 18th National Information Systems Conference, 10-13 October 1995, Baltimore, Maryland, pp384-394.
[DB76]	D E Denning and D K Branstad, A taxonomy for key escrow encryption systems, Communications of the ACM, 39(3):33-40, 1976.
[DH76]	W Diffie and M Hellman, New directions in cryptography, IEEE Transactions on Information Theory, 22:644-654, 1976.
[ElG85]	T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory, 31:469-472, 1985.

[ETSI97a]	ETSI Draft prEG 201 057, Telecommunications Security; Trusted Third Parties TTPs); Requirments for TTP services, Edition 1.1.1, May 1997.
[ETSI97b]	ETSI SEC Draft ETS, Specification for Trusted Third Party Services - Part 1 : Key management and key escrow / recovery, Version 5.0, June 1997.
[GCS97]	Global Cellular Service - Service Description, Vodafone Ltd and Swiss PTT, Issue 1, April 1997.
[IBM97]	IBM Secure Way Key Recovery Technology, White Paper.
[IETF96]	IETF, Internet Public Key Infrastructure Part III : Certificate Management Protocols, Internet Draft, November 1996.
[ITU93]	ITU-T Recommendation I.350 (03/93), General aspects of quality of service and network performance in digital networks, including ISDNs
[ITU94]	ITU-T Recommendation E.800 (04/94), Terms and definitions related to quality of service and network performance including dependability
[JMW96a]	N. Jefferies, C. Mitchell, M. Walker. A proposed architecture for trusted third party services, Cryptography : Policy and Algorithms, Lecture Notes in Comput. Sci., 1029:98-104, 1996.
[JMW96b]	N. Jefferies, C. Mitchell, M. Walker. Combining TTP-based key management with key escrow, Royal Holloway Computer Science Department Technical Report CSD-TR-96-10, 1996.
[KP96]	L R Knudsen and T P Pederson, On the difficulty of software key escrow, Advances in Cryptology - EUROCRYPT '96, Lecture Notes in Comput. Sci. 1070:237-244, 1996.
[Mic93]	S Micali, Fair cryptosystems, MIT Technical Report, MIT/LCS/TR-579.b, November 1993.
[Mit96]	C J Mitchell, Private Communication, 6th November 1996.
[UKC96]	UK CESG, Securing Electronic Mail within HMG - Part 1 : Infrastructure and Protocol, Draft C T/3113/TL/2776/11, 21 March 1996.
[VKT97]	E.R. Verheul, B. Koops, H.C.A. van Tilborg, Binding cryptography - a fraud-detectable alternative to key-escrow proposals, Computer Law and Security Report, vol. 13, No 1, 1997.
[VT97]	E.R. Verheul and H.C.A. van Tilborg, Binding ElGamal: A Fraud -Detectable Alternative to Key-Escrow Protocols. Advances in Cryptology - EUROCRYPT '97, Lecture Notes in Comput. Sci. 1233:119-133, 1997.

4. Abbreviations and glossary of terms

4.1 Abbreviations

ACRYL	Advanced Cryptographic Library (Siemens)
API	Applications Programming Interface
ASN.1	Abstract Syntax Notation (version 1)
ASPeCT	Advanced Security for Personal Communications Technologies
CA	Certification Authority
ETSI	European Telecommunications Standards Institute
EXODUS	Experiments on the Deployment of UMTS
FSM	Finite State Machine

GUI	Graphical User Interface
GSM	Global System for Mobile Communications
IETF	Internet Engineering Task Force
IP	Internet Protocol
ISDN	Integrated Services Digital Network
KRSP	Key Recovery Service Provider
LAN	Local Area Network
MAC	Message Authentication Code
MoU	Memorandum of Understanding
PAC	Privilege Attribute Certificate
PC	Personal Computer
QoS	Quality of Service
TCP	Transmission Control Protocol
TTP	Trusted Third Party
UMTS	Universal Mobile Telecommunications System
VASP	Value Added Service Provider
WP	Work Package

4.2 Glossary of terms

agent	see interception authority/agent
certificate	a collection of unforgeble information, signed by a CA, conveying trusted information about the entity to which it relates.
certification authority	an authority trusted by one or more users to create and assign certificates.
digital signature	data appended to, or a cryptographic transformation of, a data unit that allows a recipient to prove the source and integrity of the data unit and protect against forgery.
escrow	to lodge in a safe place for future reference by authorised owner or other authorised agent
interception authority/agent	an agency authorised by law, and with appropriate and specific legal warrant, to intercept or recover communications traffic or related material which may be judged to be prejudicial to the preservation of law and order.
key escrow/recovery	the process of allowing authorised persons under certain prescribed conditions to decrypt ciphertext with the help key escrow/recovery information supplied by one or more trusted parties.
repudiation	denial by one of the parties involved in a communication of having participated in all or part of the communication.
user	human user or an application using a service or network.

5. Introduction

5.1 Background to ASPeCT and WP2.3

The ASPeCT project is concerned with the security of future mobile telecommunications systems. Security features must form an integral part of future systems in order to

- to protect user traffic and terminals, and to give confidence that information and communication can be private, and free from undetected corruption;
- to protect the communications system itself against accidental or malicious abuse; this includes the operations and services, the network components and equipment, together with the commercial and business aspects necessary to maintain required levels of service.

Security features based on cryptographic techniques in second generation systems such as GSM have enabled systems to be much less susceptible to abuse than their predecessors. The increasing requirements from users, operators and regulatory bodies for appropriate security measures call for more advanced features in third generation systems, in particular UMTS. It is the goal of ASPeCT to develop and demonstrate solutions for UMTS.

Trusted Third Parties (TTPs) provide the means to allow users to establish confidential communication channels with other users, possibly in different countries, whilst being able to satisfy law enforcement requirements at both the national and international level. These type of services can also be used for data recovery in the event of lost or damaged keys. TTP-based services allow the recovery of confidentiality keys under appropriate controls - such as an extension of a search warrant.

The general objectives of Work Package 2.3 can be summarised as follows.

- to provide a Europe-wide solution to the problem of managing keys to provide security services for mobile telecommunications use;
- to carry out a prototype implementation and demonstration of the feasibility of the solution;
- to implement the solution as part of a wider UMTS trial to obtain results on the acceptability and performance of the solution in a real environment;
- to take steps to obtain appropriate European standards approval for the solution.

5.2 The first demonstrator

This document is a report on the completion of the first stage of the second of the WP2.3 objectives listed in Section 5.1, namely a first working demonstrator of TTP services. The first demonstrator was fully described in [D09] and publicly exhibited for the first time at the IS&N conference on the 28th May 1997 in Como, Italy. The demonstrator explicitly shows the use of TTPs in establishing an end-to-end encryption service, with the TTPs providing key distribution and certification services. The demonstrator also implicitly provides a key recovery service through the same TTPs. Later stages of the project will integrate the TTP services with trusted billing services based on secure micropayment techniques and with advanced authentication facilities based on smart cards. This integration will be the goal of the second demonstrator, to be completed in the next phase of the project.

5.3 Overview of demonstrator evaluation

The remainder of this document deals with a detailed evaluation of the first TTP demonstrator. This evaluation is divided into seven subsections as follows.

1. ***Key Escrow Protocol.*** An overview of the key escrow protocol implemented in the demonstrator, including analysis of protocol features, discussion of possible extensions and comparisons with alternative protocols.
2. ***TTP Services.*** A review of the services, security functions, internal operations and interfaces of the first TTP demonstrator.
3. ***Architecture.*** An overview of the entire architecture of the first demonstrator.

4. ***Appearance of Demonstration.*** An analysis of the first TTP demonstrator appearance.
5. ***User Friendliness.*** A discussion of the features and suitability of the Graphical User Interface.
6. ***Performance.*** An evaluation of the processing delays involved in the first demonstrator.
7. ***Applicability.*** A discussion on the application of a TTP infrastructure to support UMTS security services.

6. Evaluation of the demonstrator

6.1 Key escrow protocol

6.1.1 Overview of functionality provided by first demonstrator

The first TTP demonstrator provides UMTS users with a mechanism to support end-to-end confidentiality of communications. In our model two users, who wish to communicate with each other, make use of the key management services provided by a TTP infrastructure to support the establishment of a shared secret confidentiality key to be used in a symmetric cryptosystem. We assume that each user belongs to a *domain* (perhaps a country) and that they only directly communicate with a *home TTP*, which is a TTP associated with their domain.

An important feature of the mechanism is that some information used to generate the shared secret confidentiality key is escrowed to the TTPs. Thus, the demonstrator offers a mechanism whereby an *interception agent* can, under certain prescribed conditions, obtain access to communication between the users by presenting a valid warrant to an appropriate TTP. The TTPs facilitate interception by releasing certain information, which may be used by the agent to decrypt the targeted communications.

The protocol used to establish a shared secret confidentiality key in the first demonstrator is based on the JMW architecture [JMW96a, JMW96b]. The particular mechanism and protocols used in the ASPeCT demonstrator are described in ASPeCT Deliverables D07 and D09 [D07, D09]. A description of the main protocol is included in Appendix C (Section 8.3) for completeness.

Confidentiality services with key escrow may provide UMTS users with the means to establish confidential communications channels with other users, possibly in different countries, whilst being able to satisfy law enforcement requirements at both the national and international level. These type of services can also be used for data recovery in the event of lost or damaged keys.

6.1.2 Parameters and requirements of escrow protocols

We provide two lists that jointly provide a framework within which to analyse proposed escrow protocols. The first list contains *parameters* of an escrow protocol, which describe the relationship between entities in the protocol and particular properties of the protocol. The second list contains *requirements* of the protocol, which are necessary outcomes of the protocol. Note that while the requirements are distinct from the parameters of the protocol, some of the requirements are specified in terms of the protocol parameters. The two lists are partially compiled from previous lists in [JMW96a] and [VKT97]. The detailed taxonomy in [DB76] and the list of criteria in [D95] are also of interest, although somewhat broad to be of direct use in ASPeCT. In this section we also recall a couple of general problems from [KP96], which apply to all key escrow proposals.

6.1.2.1 Escrow parameters

The following parameters should be identified when proposing or analysing an escrow protocol.

I. **Communication structure:** *Who talks to whom?* This parameter includes a description of which entities are involved in the protocol, and what type of communication channel (if any) exists between them.
II. **Trust relationships:** *Who trusts who?* This parameter describes the degree of trust that entities involved in the protocol have for one another. When two entities partially trust one another the nature of this partial trust should be precisely described.
III. **Interception safeguards:** *Which communications can be intercepted, when and by whom?* This parameter details the scope of interception permissible with respect to precision of target and time length. It also describes which TTPs can assist in each type of interception.
IV. **Escrow type:** *What type of escrow?* This parameter identifies whether session keys are to be escrowed and whether the choice of encryption algorithm is to be fixed, or both.
V. **Cryptographic flexibility:** *How flexible?* This parameter describes how cryptographically flexible the protocol should be. In particular, how much choice should there be with respect to key update policies and choice and use of trusted third parties.
VI. **Communication Type:** *What type of communication?* This parameter describes the communication scenario that the key escrow protocol is to be applied to. For example, whether communication is one-way or two-way, for national or international networks.
VII. **Implementation:** *What implementation restrictions exist?* This parameter describes any relevant implementation restrictions that exist for the protocol environment. For example whether the solution is for hardware or software (or both), or whether public or secret key algorithms can be supported.

6.1.2.2 Escrow requirements

Any key escrow protocol should satisfy the following requirements.

1) **User completeness:** *Honest users succeed.* By following the protocol, honest users will succeed in establishing a session key for encrypting messages.
2) **Agent completeness:** *Honest agents succeed.* By following the protocol, an interception agent, in possession of an appropriate warrant, will be able to obtain plaintexts of any messages subject to the specifications for such interception detailed by the interception safeguards.
3) **User soundness:** *Dishonest users do not benefit.* Any user activity that is designed to misuse the protocol should at least be detectable. This includes using the framework of the protocol to establish a session key by some other means or encrypting by techniques not specified in the protocol.
4) **Agent soundness:** *Dishonest agents do not benefit.* Any agent activity that is designed to misuse the protocol should at least be detectable. This includes any activity not specified by the interception safeguards such as release of information by a TTP not designated to do so by the safeguards, release of information not specified by the safeguards, and release of information to an interception agent not in possession of an appropriate warrant.
5) **User acceptability:** *User approval.* The protocol should be acceptable to users. Factors that are likely to lead to acceptability include expert approval of the protocol, use of well-known cryptographic techniques, cryptographic flexibility and compatibility, efficiency of use and visible benefits of use.
6) **Agent acceptability:** *Agent approval.* The protocol should be acceptable to interception agents. The acceptability factors largely overlap those of user acceptability, however emphasis is different for some cases. For example *efficiency* in this case refers to efficiency of interception.
7) **Legality:** *Within appropriate laws.* The protocol should satisfy all relevant legal restrictions, including those concerning interception policy and cryptographic algorithm use and export. The protocol should also protect the relevant constitutional rights of all participating entities.

Note that while we claim that the above requirements are necessary for any escrow protocol, it may not be possible to verify that they all hold for the first demonstrator. For instance, to verify agent soundness we must ensure that interception agents can not present forged warrants. This lies outside the scope of our demonstration. Note also that *acceptability* is not well-defined. It may be the case that some participants find the parameters of the protocol unacceptable, before even considering the protocol itself. For a more detailed separation and discussion of issues concerning acceptability and legality see [VKT97]. Note that in assessing protocols we omit discussion of the last requirement, legality, as this lies somewhat beyond the technical scope of ASPeCT.

6.1.2.3 Basic problems with key escrow protocols

The following two problems allow users to exploit the framework of a key escrow protocol to send communications that cannot be subsequently accessed by interception authorities. The relevance of such problems should be taken into account when assessing user soundness of a key escrow protocol.

1. *Use public keys in other systems:* In a public key system two users A and B can use the public keys in a different system than the one specified by the protocol.
2. *Hash session keys:* Users A and B use the specified protocol to exchange keys k_i, however session keys K_i are computed $K_1 = H(k_1), \ldots, K_n = H(K_{n-1}, k_n)$, where H is a one-way hash function.

The first type of general problem could be avoided in certain implementations through the use of a tamper-resistant device. However the second type of problem seems very difficult to avoid. We suggest that it is thus quite hard to prevent dishonest users from abusing the framework of a protocol in ways such as those described here. In [Mit96] it was suggested that in any protocol where users are provided with an authenticated channel it is impossible to prevent such attacks. While acknowledging this point, it is still important to consider such problems when assessing user soundness of a protocol. Although almost impossible to prevent completely, it can be argued that a simple to use and efficient user protocol discourages this type of user abuse in a significant number of cases.

6.1.3 Desirable Parameters for the demonstrator

We will now consider the scenario assumed in the demonstrator, which we refer to as the *basic scenario*, and identify a set of desirable parameters for this basic scenario.

I. **Communication structure:** Two UMTS users A and B, register with separate home TTPs, denoted TA and TB respectively. User A and TA share a secure link, as do user B and TB. The TTPs TA and TB have access to a secure link but use of this link is restricted as it is regarded as expensive. Users A and B communicate over an insecure link. Interception authorities can communicate with either TA or TB.
II. **Trust structure:** Users and interception agents trust both TA and TB. Users and interception agents do not trust one another directly but rather trust the TTPs to act honourably in dealings between them. The TTPs do not need to trust the users or interception agents however they have some partial trust, at least to the extent that if either of these entities regularly abuse TTP services then the TTPs may lose their *trusted* status. This also applies to trust between TA and TB. Users A and B partially trust one another, at least not to subsequently reveal shared session keys.
III. **Interception safeguards:** Interception agents have the potential to access any message sent between A and B (if an appropriate warrant is obtained). Interceptions should however be *targeted* and *time-bounded* A targeted interception specifies precisely whether *all* messages from (or to) a specific user can be intercepted, or whether only messages from (or to) other specifies users can be intercepted. Time-bounded interceptions cover only a specified time, marked by dates or time-stamps. All messages sent outside the specified interception period should remain fully protected.
IV. **Type of escrow:** Only session keys are to be escrowed. Any encryption algorithm can be used.

V. **Cryptographic flexibility:** Users should be able to generate (or request) fresh keys as often as possible. The protocol should permit any symmetric encryption algorithm to be used for the encryption of messages. In this basic scenario users have one fixed home TTP.
VI. **Communication type:** One-way or two-way communication.
VII. **Implementation:** No restrictions for the basic scenario.

6.1.4 Evaluation of the demonstration

The protocol in the demonstrator is based on the *JMW protocol* [JMW96a, JMW96b], which has received widespread attention and has sometimes been referred to as the *Royal Holloway* protocol. Several variants of this protocol have appeared including those in [UKC96] and [CGM96]. We note also that in [AR97] a variant of the JMW scheme is referred to as the *GCHQ* protocol. All variants of the JMW scheme are based on the Diffie-Hellman key exchange protocol [DH76].

6.1.4.1 The protocol

Note: we describe here a simplified version of the JMW protocol. A full description of the protocol implemented in the first demonstrator can be found in Appendix C (Section 8.3).

Let p be a prime and g be a primitive element modulo p (these values are public). Let $K(TA,TB)$ be a secret key shared by TA and TB. Let f be a key generating function that takes as input a user identity and $K(TA,TB)$ and outputs a private receive key for that user. The protocol runs as follows:

1. A sends a message to TA requesting communication with B.
2. TA chooses a private send key a for A, and sends to A the values a, g^a, a signed copy of g^a and g^b, where b is the private receive key of B (which can be generated by both TA and TB).
3. A computes the session key g^{ab} and sends g^a and a signed copy of g^a to B.
4. B verifies g^a and computes g^{ab} (we assume that B has already received b from TB).

A simplified version of this protocol is given in Figure 6-1.

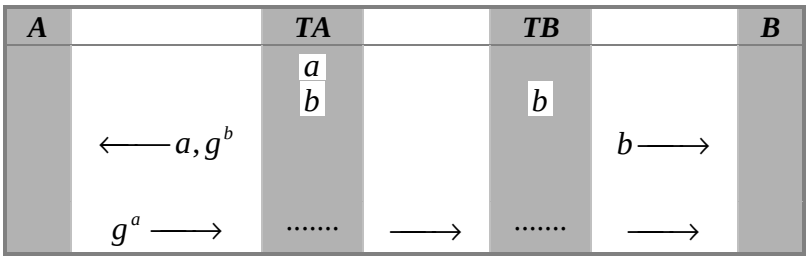


Figure 6-1: The basic JMW protocol of [D07]

We list the types of interception that are possible for the JMW protocol:

- TA releases a. All messages from A can be read for the lifetime of a.
- TA or TB releases b. All messages from users with home TTP TA to B can be read for the lifetime of b.
- TA or TB releases g^{ab}. All messages from A to B can be read for the joint lifetime of a and b.

6.1.4.2 Evaluation against parameters

We now check if the protocol has the parameters of our basic scenario. A quick check reveals that it *almost* does. The only parameter that is not agreed upon is the interception safeguards. As can be seen from the types of interception available, while precise targeted interception is possible by the third of the three interception types, there is no provision for time-bounded interception (except between private send and receive key updates). We discuss variants that permit time-bounded interceptions in Section 6.1.5.2.

6.1.4.3 Evaluation against requirements

We now test the protocol against our list of requirements.

1) **User completeness:** Yes.
2) **Agent completeness:** Yes.
3) **User soundness:** The general problems that apply to all key escrow protocols exist (see Section 6.1.2.3).
4) **Agent soundness:** No. Although this is a difficult requirement to ensure there should at least be provision for time-bounded interceptions before this can be satisfied.
5) **User acceptability:** No. Certainly not acceptable without clarification of agent soundness. The protocol does use established techniques however and does allow users to request fresh private send keys at any time. Private receive key generation is a more complex operation as updating of keys requires communication between TA and TB.
6) **Agent acceptability:** Needs clarification of the user soundness. Probably more acceptable to agents than users. Some efficient interceptions are possible through the release of private send or receive keys, however these types of interception should only be permissible when an interception is targeted in such a way that all affected users are covered by the warrant.

6.1.5 Enhancements

6.1.5.1 Interception options

In the evaluation of the protocol against our requirements we noted that that agent soundness should be clarified and user acceptability increased. Both these issues can be addressed by considering the exact way in which the TTPs provide warranted access to communications. The purpose of this discussion is to show what options are available.

There are two possible ways for a TTP to provide warranted access to communications. The TTP could pass the appropriate keys to the intercepting agent, or the TTP could use its escrowed key(s) to decrypt messages presented to it by the intercepting agent, without revealing the keys themselves. We discuss each approach in turn.

6.1.5.1.1 TTP releases escrowed keys

Three different keys can be released by TTPs involved in a particular communication.

- private send key
  The sender's TTP could release the private send key of the sender, which would allow all messages sent from the sender to be decrypted during the lifetime of the private send key.
- private receive key
  Either the sender's TTP or the receiver's TTP could release the private receive key of the receiver, which would allow all messages sent to the receiver from all users of the sender's TTP to be decrypted during the lifetime of the private receive key
- session key

Either the sender's TTP or the receiver's TTP could release the session key, which would allow all messages from the sender to the receiver to be decrypted during the joint lifetime of the sender's private send key and the receiver's private receive key.

The release of private send or receive keys allows some efficient interceptions to be made. However, care must be taken to ensure that these keys do not allow the agent to access communications which are not covered by the warrant. For some types of warrant it may not be permissible to release the private send key or private receive key. Instead, the TTP must release the session key for each message covered by the warrant.

In order to investigate the acceptability of the scheme to both users and agents, we consider some of the possible types of warrant that may be presented. First we note that the scheme supports both node-based interception (in which all communications involving a particular target can be decrypted) and edge-based interception (in which only communications between two targets can be decrypted). Both types of interceptions are now discussed in turn. For each type of interception we explain how warranted interception can be provided.

We consider four possible types of node-based interception:

N1. A TTP is warranted to provide access to all outgoing communications from a user for which it acts.

The TTP can provide the private send key(s) for the targeted user.

N2. A TTP is warranted to provide access to all incoming communications to a user for which it acts.

The TTP can provide the private receive key(s) for the targeted user.

N3. A TTP is warranted to provide access to all incoming communications (from users for which it acts) to a user for which it does **not** act.

The TTP can provide the private receive key(s) for the targeted user.

N4. A TTP is warranted to provide access to all outgoing communications (to users for which it acts) from a user for which it does **not** act.

The TTP can provide the session key(s) for outgoing communications (to users for which it acts) from the targeted user.

Interceptions N1, N2, and N3 can be provided with good agent acceptability, since the release of private send and receive keys increases the efficiency of interceptions. Interception N4 has a lower agent acceptability, since individual session keys must be obtained. However, although interception N4 is less efficient, it is likely to be much less common in practice than N1 and N2 (and possibly N3). Note also that in each case, user acceptability and agent soundness are maintained by ensuring that the agent can only access those communications covered by the warrant.

We consider two possible types of edge-based interception:

E1. A TTP is warranted to provide access to communications from a particular user for which it does act to a particular user for which it does **not** act.

The TTP can provide the session key(s) for communications between the two users.

E2. A TTP is warranted to provide access to communications from a particular user for which is does **not** act to a particular user for which it does act.

The TTP can provide the session key(s) for communications between the two users.

For edge-based interception, only individual session keys can be released. These can be obtained from either TTP and only provide access to communications between the two targeted users. Thus, user acceptability and agent soundness is maintained.

6.1.5.1.2 TTP performs decryption

If the TTP performs decryption then it is easy to ensure that interceptions are targeted, since the actual communications to be decrypted are presented to the TTP. Thus, agent soundness and user acceptability are maintained, and many of the problems associated with releasing keys are avoided. The main disadvantage of this approach is the increased amount of communication required between the TTP and the intercepting agent, and the potential delay in accessing communications. However, this disadvantage could be partially overcome if the key established in the protocol was used as a key encrypting key used to encrypt the actual session key. In this case the agent would present encrypted session keys to the TTP for decryption, rather than whole messages.

6.1.5.2 Time-boundness

In order to increase the agent soundness and user acceptability of the protocol it is worth incorporating the option of time-bounded interceptions. We consider a number of proposals.

6.1.5.2.1 Variant I

This variant was proposed in [JMW96b]. The protocol is identical to the JMW protocol except that the private receive key of B is regularly updated. Hence the key b of the JMW protocol becomes a *permanent* receive key, and a second key generating function G is applied to b and a date-stamp d to compute a *temporary* receive key $b' = G(b,d)$. Time-bounded interceptions are now possible by the following types of interception:

- TA or TB releases b'. All messages from users with home TTP TA to B can be read for the validity of date-stamp d.
- TA or TB releases $g^{ab'}$. All messages from A to B can be read for the joint lifetime of a and validity of date-stamp d.

6.1.5.2.2 Variant II

This variant is based on ideas in [UKC96]. This protocol is very similar to JMW variant II except that the private send key of A is also regularly updated. Thus the key a of the JMW protocol becomes a *permanent* send key, and a key generating function G is applied to a and a date-stamp d to compute a *temporary* send key $a' = G(a,d)$. For simplicity we assume that the date-stamp is updated with the same frequency for both temporary send and temporary receive keys, although this need not be the case. Time-bounded interceptions are now possible by the following types of interception:

- TA releases a'. All messages from A can be read for the validity of date-stamp d.
- TA or TB releases b'. All messages from users with home TTP TA to B can be read for the validity of date-stamp d.
- TA or TB releases $g^{a'b'}$. All messages from A to B can be read for the validity of date-stamp d.

6.1.5.2.3 Variant III

Proceed as in the JMW protocol until a session key g^{ab} has been established by user A. Now A takes a hash function H and a date-stamp d and computes a *session key of the day* $H(g^{ab},d)$ which is employed as the session key for encrypting messages from A to B. Time-bounded interceptions are now possible by the following type of interception:

- TA or TB releases $H(g^{ab},d)$. All messages from A to B can be read for the joint lifetime of keys a, b and validity of date-stamp d.

Note that it is most likely that the lifetimes of keys a and b span several date-stamps and so a time-bounded interception over an extended time period is likely to require the release of $H(g^{ab},d)$ for all date-stamps d corresponding to the time covered by the court order.

6.1.5.2.4 Variant IV

Proceed as in JMW variant II except replace date-stamp d with date-stamp D, where D lasts for a greater time period than d. User A then proceeds to establish the session key as in JMW variant II but then incorporates the ideas of JMW variant III. Thus user A takes a hash function H and date-stamp d and computes a session key of the day $H(g^{a'b'},d)$, which is then employed as the session key for encrypting the message from A to B. Time-bounded interceptions are now possible by the following types of interception:

- TA releases a'. All messages from A can be read for the validity of date-stamp D.
- TA or TB releases b'. All messages from users with home TTP TA to B can be read for the validity of date-stamp D.
- TA or TB releases $H(g^{a'b'},d)$. All messages from A to B can be read for the joint validity of date-stamps d and D.

Thus in JMW variant IV time-bounded interceptions for short time intervals and precise targets can be conducted by release of session keys by either TTP. For longer interceptions that apply to wider targets it is possible to release the private temporary send or receive keys that are date-stamped by all D covered by the appropriate court order. The exact granularity of the two different date-stamps can be adjusted to fit the application.

6.1.5.3 Two-way communication

Although we have discussed the basic protocol (and variants) in terms of one-way communication, there is no reason why session keys established by the protocol cannot be used for two-way communication. If it is preferred that both users should contribute to the establishment of a two-way session key then the one-way protocol can be performed twice, once in each direction, and the two resulting session keys could be combined. Note that if a session key has been agreed upon for two-way communication between user A and user B then, when a warrant is issued for interception of communication from A to B, it is unavoidable that all communication from B to A will also be intercepted. If this situation is not acceptable then such two-way keys should not be established and communication between A and B should take the form of two separate streams of one-way communication (one from A to B, and the other from B to A) with a different one-way session key protecting the information flow in each direction.

In Appendix A various merits of several variants of the basic one-way communication protocol extended to two-way communication are considered. These are based on several simplified protocols that had previously appeared in ASPeCT related documents. All these protocols are Diffie-Hellman [DH76] based protocols for establishing a two-way session key. The following general conclusions resulted from this study.

- *Combining two one-way keys does not seem profitable.* There does not seem to be any significant advantages in combining the two one-way keys as suggested as an option in [JMW96a] as this appears to result in a protocol that does not offer many obvious gains over a simple Diffie-Hellman key exchange [DH76] with escrowed secret keys..
- *It is possible to use "one-way" keys for "two-way" communication.* User security does not seem to be affected by the adoption of a one-way key for two-way communication. This allows the benefits of efficient warranted interception in some one-way protocols to be extended to two-way protocols. The basic one-way escrow scheme of [JMW96b] is the most efficient in this regard.
- *Directional targeting can not be done with a two-way key.* This somewhat obvious comment is made to highlight the fact that if sent and received communication is to be separated then a protocol that establishes a two-way key should not be used. Rather, a one-way key should be established in each direction and these keys should not be combined, but used separately, one for communication in each direction (the basic two-way version of [JMW96b]).

6.1.5.4 Escrow in multiple domains

The first demonstration assumes that each user belongs to a domain and only directly communicates with a *home TTP*, which is a TTP associated with the user's domain. Interception agents in either the sender's or the receiver's home domain can then access the communications by approaching the appropriate TTP under their jurisdiction.

However, in a multiple domain environment it may be required that interception agents in other domains, other than the home domain of the sender and receiver, have access to the communications. Consider for example a potential scenario in UMTS where two UMTS users A and B, who communicate with each other using end-to-end encryption, are citizens of countries C and D, respectively, work in countries E and F, are registered with two networks in countries G and H, and are roaming in two countries I and J. Their traffic might conceivably need to be intercepted by agents in any of the countries (domains) C-J. In this scenario the secret confidentiality key may need to be escrowed to TTPs in all the countries C-J.

Key escrow in multiple domains is easily catered for in key escrow schemes which are not combined with key distribution; keys are simply escrowed to all the required TTPs. In the JMW scheme, where key escrow is combined with key distribution, multiple domains can be catered for using a conference key distribution scheme, which allows the TTPs in the required domains to independently gain access to an escrowed key and make an updated contribution to the key without communication with TTPs in any other domain [CM96]. The use of key escrow in multiple domains may be used to increase agent acceptability in some scenarios.

6.1.5.5 Split escrow

In the demonstration a user can only register with one TTP. However, in practice a user may want the extra reassurance offered by having their keys shared between a number of independent keys. This idea is sometimes called split escrow. In order to support split escrow, a user *X* must be able to register with a set of TTPs $\{TX_i\}$, which the user is prepared to trust collectively, but perhaps not individually. The basic protocol could be adapted using the ideas of Micali [Mic96] or by using the solution proposed in [JMW96b]. In this scenario we must assume that each pair of TTPs (TA_i, TB_i), have access to a secure

link, and that in addition, interception agents can communicate with the sets of TTPs $\{TA_i\}$ *and* $\{TB_i\}$, in order to obtain the means to intercept communications.

6.1.5.6 Residual work factor

It could be argued that user acceptance would be increased if recovery of session keys required a considerable amount of effort on the part of the TTP and the interception agent. This approach would also help to discourage casual recovery requests.

Increasing the residual work factor associated with key recovery can be increased in some schemes quite simply. For instance, in key escrow schemes which are not combined with key distribution, the user can retain a variable amount of the session key, and escrow the rest to a TTP. Thus, an interception agent can only recover part of the session key from the TTP, the rest must be recovered as part of a labour intensive exhaustive key search.
An enhancement to the demonstrator would be to include a mechanism for increasing the residual work factor associated with key recovery.

6.1.5.7 Increased cryptographic flexibility

It is assumed that *g* and *p* are system wide parameters. However, in practice different integers *g* and *p* could be agreed between each pair of TTPs in the network. If this approach was adopted then each user must receive the value of *p* before it can carry out the required cryptographic operations. This could be achieved by ensuring that the appropriate TTP sends the value of *p* to the user as part of the protocol.

6.1.5.8 Implementation on trusted hardware

The security functionality in the demonstration is implemented in software on the demonstration PCs. However, the security of the protocol relies on a trustworthy implementation, which protects certain cryptographic information which must be stored by users' TTPs. In order to achieve a higher degree of security, parts of the security functionality on the user's terminal could be implemented on a separate, trusted, tamperproof security module or smart card. On the TTP side all the security functionality should be trusted, though implementation on trusted hardware within a tightly controlled operation and management regime.

The split of functionality between the terminal (PC) and the smart card will be limited by the processing and storage capabilities on the smart card. Thus, it is important to implement those functions on the smart card which provide the greatest increase in the level of security. For instance, it would be desirable in the first instance to store the public and private components of send and receive keys in a smart card and to calculate the session key from these values in the smart card. In addition it would be desirable to perform the encryption in the smart card, but the bandwidth limitations of the card interface are likely to make this difficult, especially when high bit rate services such as those proposed for UMTS are involved [D11].

It has been previously assumed that smart cards will be used in the second demonstrator and trial. However, the exact split of functionality has yet to be decided.

6.1.5.9 Confidentiality of user-TTP communications

An enhancement for the second demonstration would be to protect the user-TTP communications channel. In the simplest case this could be achieved using a static session key which could be distributed to the user manually when he first registers with his home TTP. An alternative is that the user and TTP establish this shared secret cryptographically. This would increase the flexibility and practicality of key management. The secret key shared between the user and his TTP could be released to an interception agent by the TTP under certain prescribed conditions.

6.1.5.10 Compliance with emerging standards

In order to aid interoperability, the demonstrator should be compliant with emerging standards, specifications and recommendations. ETSI are currently defining a set of European Telecommunications Standards for Trusted Third Party Services. These standards will be based on the set of requirements detailed in [ETSI97a]. The first part of these standards will specify key management services [ETSI97b]. The key management services will include support for the generation and distribution of session keys for use in symmetric encryption systems, including support for the escrow of session keys.

Furthermore, IETF have produced a set of four Internet Drafts which specify an architecture for a Public Key Infrastructure [IETF96]. The aim of the architecture is to provide support for privacy and digital signature services over the Internet in support of international commerce, balancing legitimate needs of commerce, governments and privacy of citizens. As such the draft specifications support the need for key recovery.

6.1.6 Other escrow protocols

Key escrow/recovery is currently a very active research area and there are many protocols and schemes under discussion. In Appendix B we describe three well publicised alternative protocols and briefly compare them to the protocol implemented in the first demonstrator.

6.2 TTP services

6.2.1 Introduction

One well established role for TTPs is the generation, distribution and management of public key certificates. Such certification services will become increasingly important in future mobile telecommunication services as public key-based security services become more widespread. Another important role for TTPs in future mobile networks will be in supporting end-to-end confidentiality services, with key escrow to enable lawful interception.

In the first demonstrator the TTP is used to support key management for confidentiality, with key escrow functionality, by implementing a protocol based on the JMW architecture [JMW96a, JMW96b]. Generally, the specifications in deliverables [D07] and [D09] have been implemented in the first TTP demonstration. In this demonstrator two ASPeCT defined TTPs (TA and TB) provide public key certification services to their two users (User A and User B respectively). Based on the certification services, these two users establish a shared key for end-to-end confidential communications and the key shared between two users is escrowed at both TTPs.

The public key certification services are implemented in the first demonstrator, including the demonstration of key generation, certification, distribution, storage and verification amongst User A, User B, TA and TB.

The key escrow service is held implicitly in this demonstration version, as either TTP has access to the secret key shared between User A and User B, however in this first demonstrator version, no specific interfaces (i.e., user interface, key escrow interface, inter-TTP interface and application interface) have been provided for an explicit key escrow/recovery service.
Comparing with the requirements for TTP services described in an ETSI Guide [ETSI97a], and with other requirements for TTP internal operations described in [D02], we shall explain what TTP services,

internal functions and operations, and external interfaces have been provided in the following subsections.

6.2.2 TTP Services

An ETSI Guide on Requirements for Trusted Third Party Services [ETSI97a] describes six TTP security services. The first ASPeCT TTP demonstrator includes two of them.

ETSI TTP services	Available in Demo 1	Consideration for Demo 2 & Trial
Key management services for symmetric cryptosystems	**N**	**N**
Key management services for asymmetric cryptosystems	**Y**	**Y**
public/private key pair generation	Y	Y
public key certification	Y	Y
public/private key pair distribution	Y	Y
public/private key pair revocation	N	Y
public/private key pair storage and retrieval	Y	Y
public/private key pair archival	Y	Y
Key escrow/recovery services	**Y**[1]	**Y**
Identification and authentication support services	**N**	**N**
Access control support services	**N**	**N**
Non-repudiation services	**N**	**N**

6.2.3 TTP security functions

An ETSI Guide on Requirements for Trusted Third Party Services [ETSI97a] describes sixteen TTP security functions. The first TTP demonstrator includes four of them.

ETSI TTP functions	Available in Demo 1	Consideration for Demo 2 & Trial
Key generation	**Y**	**Y**
Key distribution	**Y**	**Y**

[1] implicit recovery

Key revocation	**N**	**Y**
Key archival	**N**	**N**
Key storage/retrieval	**Y**	**Y**
Key reconstruction	**N**	**Y**
Public key certification	**Y**	**Y**
Certification for access control	**N**	**N**
Claimant/verifier exchanges	**N**	**N**
Evidence generation	**N**	**N**
Evidence recording	**N**	**N**
Evidence verification	**N**	**N**
Dispute resolution	**N**	**N**
Time-stamping	**N**	**Y**
Audit	**N**	**N**
Delivery authority	**N**	**N**

6.2.4 TTP internal operations

ASPeCT deliverable [D02] describes nine TTP internal operations. The first TTP demonstrator includes five of them.

TTP Internal Operations in [D02]	**Available in Demo 1**	**Consideration for Demo 2 & Trial**
Cryptographic computation	**Y**	**Y**
Cryptographic key storage	**Y**	**Y**
User information storage	**Y**	**Y**
Cryptographic key generation	**Y**	**Y**
Event information storage	**N**	**N**
Access control information generation	**N**	**N**
Certificate generation	**Y**	**Y**
Event analysis	**N**	**N**
Time-stamp generation	**N**	**Y**

6.2.5 TTP interfaces

An ETSI Guide on Requirements for Trusted Third Party Services [ETSI97a] describes three communication relationships between TTP and user. The first TTP demonstrator includes two of them.

ETSI communication relationships between TTP / user	Available in Demo 1	Consideration for Demo 2 & Trial
off-line	**Y**	**Y**
on-line	**Y**	**Y**
in-line	**N**	**N**

An ETSI Guide on Requirements for Trusted Third Party Services [ETSI97a] describes four TTP related interfaces. The first TTP demonstrator includes two of them.

ETSI TTP Interfaces	Available in Demo 1	Consideration for Demo 2 & Trial
User interface	**Y**	**Y**
Key escrow/recovery interface	**N**	**N**
Inter-TTP interface	**Y**	**Y**
Application interface	**N**	**Y**

6.2.6 Criteria

1. *Critique of Demo 1 itself*

   The TTP security services, security functions, internal operations and interfaces developed for the first TTP demonstrator meet the specifications of the demonstration in deliverables [D07] and [D09]. They all work well.

2. *General applicability*

   The TTP security services, security functions, internal operations and interfaces developed for the first TTP demonstrator contain only part of the basic requirements for TTPs defined in [ETSI97a], namely those concerned with key generation, certification, verification and escrow based on asymmetric cryptographic systems.

   These TTP security services, security functions, internal operations and interfaces implemented for the first TTP demonstrator are suitable for various application environments.

3. *Secure billing requirements*

   For the second demonstration of supporting a secure billing service, it is assumed that the certificates have yet to be retrieved and checked against a revocation list from a TTP, which is on-line during the charging procedure.

4. *Trial requirements*

   In the trial the ASPeCT TTP will provide on-line certification services to the mobile user and VASP (Value-Added Service Provider), who both require certificates as part of the secure billing service. The certification management services will include time-stamping and revocation list management.

6.2.7 Enhancements

The following enhancements are suggested for the specification and implementation of the second demonstrator and the trial.

- Determine a suitable TTP based protocol either by improving the protocol currently used in the first TTP demonstration or by choosing an alternative, in order to meet the requirements on both Secure Billing service and trial with EXODUS.
- Design and choose more general application program interfaces to meet the requirements of more general and varied application environments.
- Add more TTP security services, functions and internal operations to meet the requirements on both the secure billing service and the EXODUS trial, for example adding key revocation and certification for access control and time-stamping.

6.2.8 Remark on authentication

It is an assumption of the first demonstrator implementation that there exists a protected channel between the User and its TTP (channel-P, say). This assumption will be maintained in future versions of the TTP implementation. The rationale is as follows.

- In each of the two scenarios - EXODUS-based trial and Secure Billing Demonstrator - an earlier authentication of the User will have taken place before the TTP service is accessed.
- Details of the strength and longevity of the authentication between the User and TTP are outside the scope of this development.
- A further assumption is that there is trusted communication between the various relevant components of the demonstrator; hence an authenticated identity (or evidence thereof) from the earlier authentication can be passed between components.
- As the authenticated identity of the User is known, simple protection of channel-P might be the use of a previously-established long-term secret key between the User and TTP or use of each other's public-keys.
- A further autonomous authentication between the User and TTP is not necessary, and would be an unwelcome additional operation.

6.3 Architecture

6.3.1 Introduction

This section examines the architecture of the first TTP demonstrator against the relevant documents:

- D09 Specification [D09]
- D07 Specification [D07]
- D02 Requirements [D02]
- ETSI Requirements [ETSI97a]

The [D09] specification was restricted to a subset of features and facilities to be implemented as part of that deliverable, and hence the implementation closely conforms to this document.

Document [D07] described a broader perspective based on [D02] (which in turn formed one of the principal inputs to [ETSI97a]). It suggested a number of architectural views of a TTP Service.

1. Logical and physical model comprising two users and two TTPs (see Figure 6-3);
2. TTP client/server model (see Figure 6-4);
3. Internal interface architecture of TTP server (see Figure 6-5);
4. Internal software architecture (see Figure 6-6).

6.3.2 General considerations

In general, the implemented TTP architecture meets the relevant parts of the specifications in deliverables [D07] and [D09]. This TTP architecture developed for the first TTP demonstrator is suitable for simple integration with other systems/applications.

The entities involved in this TTP architecture are two TTP servers and two mobile users. The Value-Added Service Providers (VASPs) and administrative roles specified in [D07] are not explicitly provided in this version. The logical and physical connections between the above entities in the demonstration are shown in Figure 6-2. All three physical configurations shown have been set up. (Other physical configurations - 3 PCs, or different distributions of roles in 2 PCs, say - are also feasible.)

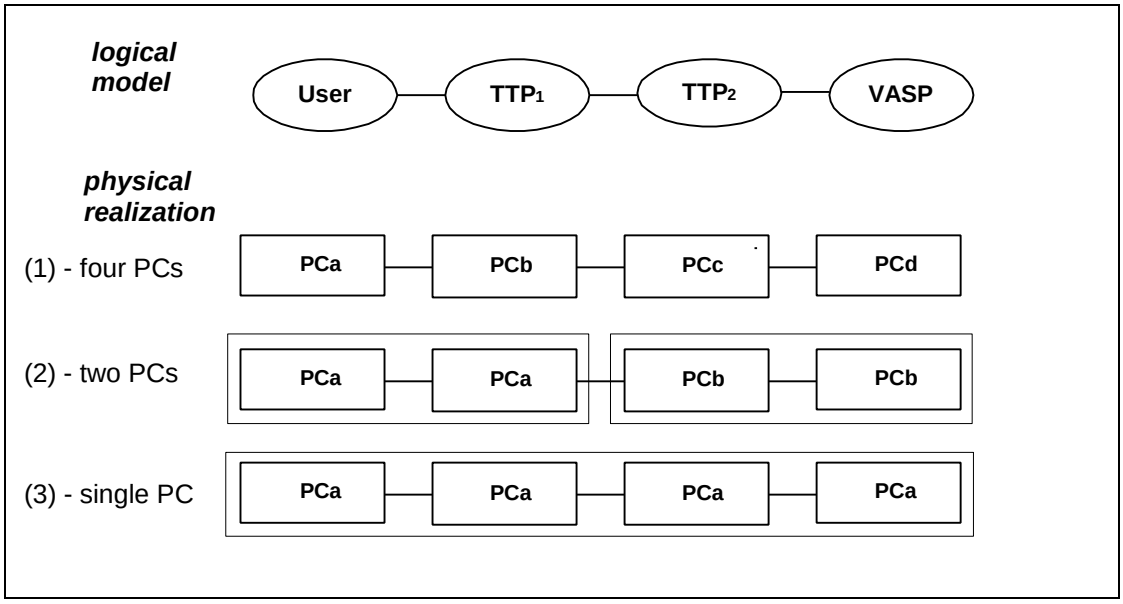


Figure 6-2 : Logical and physical connections between the entities involved in the demonstration.

Note: This first TTP demonstrator uses no external applications or services, but consists only of software produced by the project. The demonstration "application" comprises the execution of a protocol establishing an escrowed key supporting end-to-end confidential communication.

6.3.3 Service architecture considerations

The following gives an indication of TTP applicability in the three relevant cases.

	Applicability/Use
off-line	end-to-end communication
on-line	establishment of keys for end-to-end communication
in-line	none

6.3.4 Top-level architecture

Figure 6-3 shows the logical model of the first demonstrator. This model remains valid for the subsequent implementations. In particular, in the context of the Secure Billing demonstrator, for any User (or more correctly any *instance* of a User), there will be a single current TTP service provider, TA; TA may have bilateral relationships with many TB (and vice versa) secured by knowledge of each other's certified public keys; the target - (an instance of) a Value-Added Service, say - again depends on the support of a single current TTP service provider (TB).

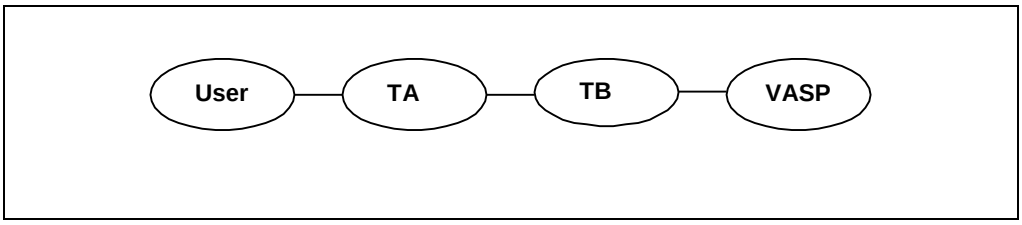


Figure 6-3 : Two-user/ two-TTP model

6.3.5 Functional architecture

Figure 6-4 illustrates the relationship between a TTP service user/client and the TTP server. The service components may be distributed over a number of servers (physical or virtual machines).
The feasibility of, and need for, future implementations to provide the layering and modularity of this view is for further study; in particular the need for some protection of inter-process communication (over and above that already provided by the protocol).

6.3.5.1 TTP service API

The user interface provided in the first demonstrator is the GUI (Graphical User Interface), which was entirely appropriate to that context. For the future implementations, the *normal* user may not act directly with the TTP service, but with applications which in turn need to use the TTP service. In this context an API (Applications Programming Interface) may be required.
In addition to supporting normal user applications, the API may be required to provide for the *key-recovery* facility, and possibly some simple management operations. This is for further study.

6.3.5.2 TTP services, functions and internal operations

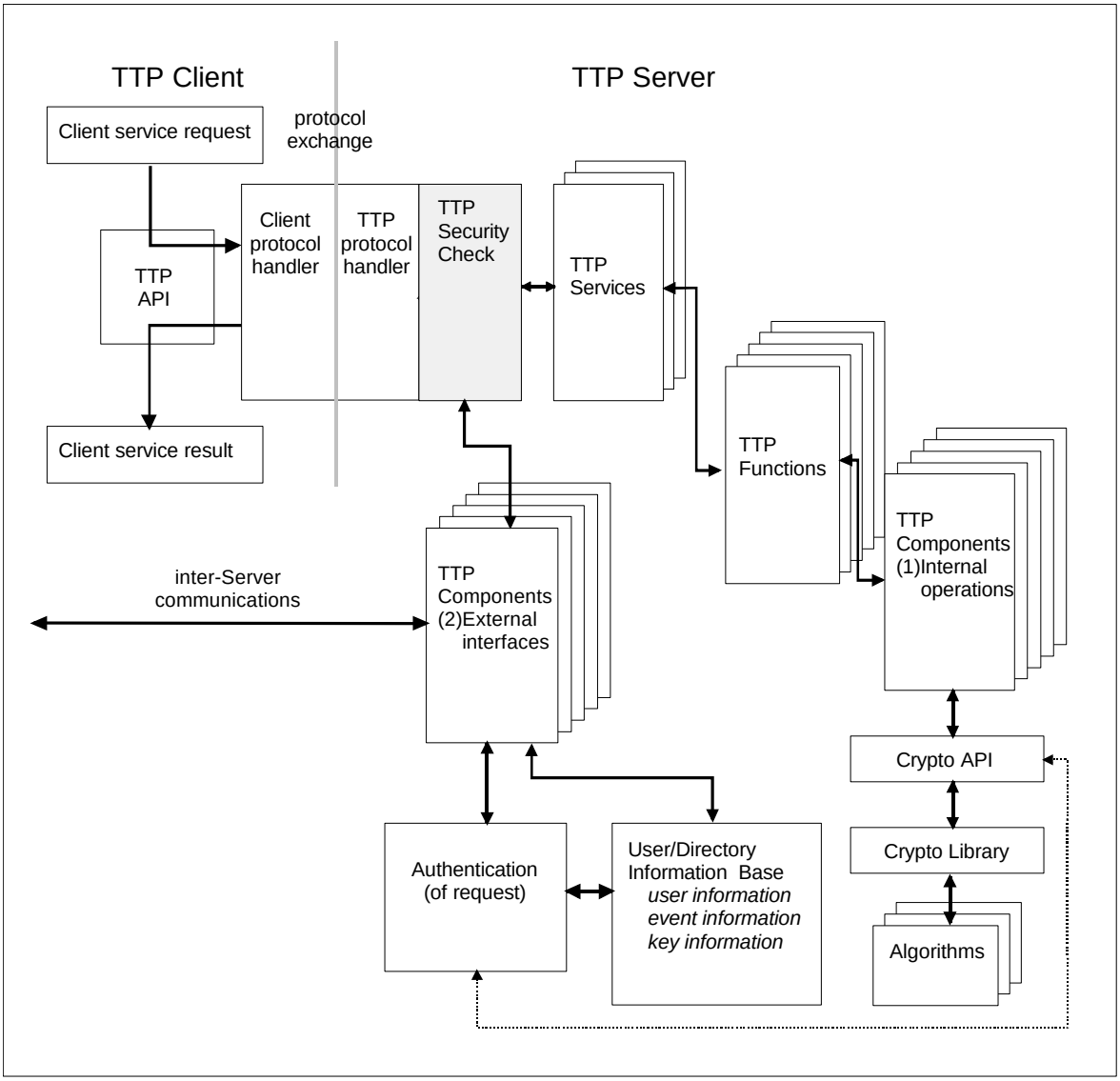


Figure 6-4 : Relationship of TTP functional units

6.3.6 Internal architecture

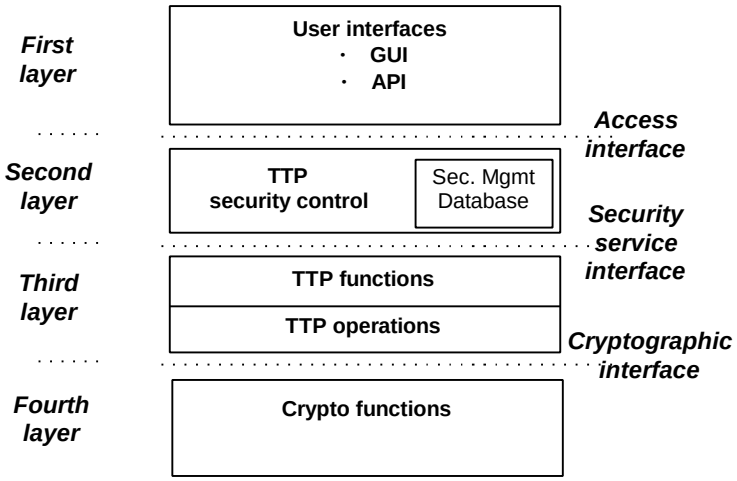


Figure 6-5 : Internal interface architecture of TTP server

Document [D07] defined a number of layers which relate to Figure 6-5.
This gives rise to identifiable interfaces between layers which provide useful external references.

- access interface;
- security service interface;
- internal operations interface;
- cryptographic interface.

6.3.7 TTP software architecture

The TTP uses the general software structure described in [D07] as shown in Figure 6-6.

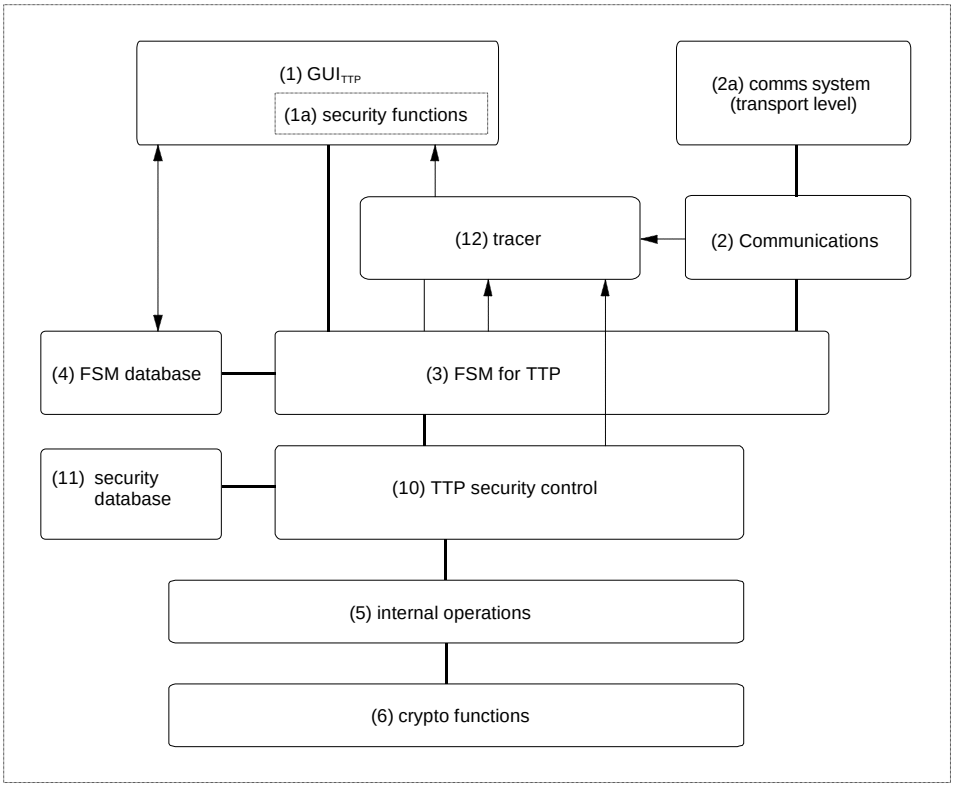


Figure 6-6 : Software architecture of first demonstrator

A description of the functional blocks in the software structure is as follows.

Block (1): GUI

The TTP Graphical User Interface has the following functionality.

1. Provides a TTP User Interface, including additional Block 1A: *Security Functions*.
2. Supports multiple concurrent windows displaying simultaneous "views" of activities - e.g. $USER_A$, $USER_B$, TA and TB.
3. Sets *trace-points* and levels of trace-points before and during the tests or demonstrations.
4. Allows a user to start execution of a protocol.
5. Shows the information related with the demonstration, such as Set-up. Run and History.
6. Offers additional procedures to administer databases (e.g. *Security Database*, Block (11)) and to open windows and display messages in them (e.g. for *Tracer* - Block (12)).
7. Processes administration of the communication database within Block (2): creating a new entity with its address; configuring the TCP/IP network; and activating the TCP/IP connections.
8. Provides access to the *Tracer* to set-up and to monitor *trace points*.

Block (2): Communications

This block is the communications sub-system which provides access to external TCP/IP communications and to local media. An additional sub-block (Block 2.6) is used to provide access to a diskette.

Block (3): Finite State Machines for TTP

This software block defines all the finite state machine classes. One instant of a finite state machine class represents one entity in the configuration. One application may represent one or more entities. One of the available finite state machines is chosen for each entity running on the PC. A single FSM is required for a TTP. In the first demonstrator only a single mode of operation supporting an end-user interface will be provided. Future versions will support additional modes. The role of the FSM is

- to recognise TTP requests and protocol messages,
- to analyse incoming protocol messages,
- to formulate protocol response messages,
- to identify security aspects, and
- to initiate the appropriate sequence of actions in security control.

Block (4): FSM database
This block holds parameters used in the TTP finite state machine. These parameters can be written via the GUI.

Block (5): Internal Operations
In the general model in [D07], this block is titled Security Functions/Procedures. Here, TTP *internal operations* are in this block. Details of the interface depend on the facilities that will be available from the ACRYL library, although it is currently anticipated that differences in, say, *certificate* will necessitate some specific interface and operations here.

Block (6): Cryptographic Functions
This block contains the cryptographic algorithms and related computational facilities. These are provided by the ACRYL library.

Block (7): ASN.1
Not shown, as it is not planned to use ASN.1 in the first TTP demonstrator.

Block (8): Winsocks
Not shown, as this block contains the standard Winsock libraries, and is accessed only via Block (2).

Block (9): Existing applications
Not shown, as no existing applications are used by TTP.

Block (10): Security Control
In the case of a TTP this could be integrated into the FSM block. It is currently a separate block which checks identified security issues

- initiates sequence of *internal operations*, and
- passes results of internal operations back to FSM.

Block (11): Security Database
The security database holds parameters specific to the security layer and is administered via the GUI.

Block (12): Tracer
The tracer can display messages on a screen. The finite state machines instruct the tracer to display or save messages. Trace-points are defined in the finite state machine.

This software architecture of the first TTP Demonstrator has the following interfaces.

FSM - TTP security control

This interface consists of the TTP *functions* implemented.
TTP security control - TTP functions and operations
This interface consists of the TTP *internal operations* implemented.
TTP functions and operations - cryptographic functions
This is the cryptographic interface. It is to be provided by the ACRYL API.

6.3.8 Criteria and Enhancements

1. *Critique of Demo 1 itself*

   The TTP architecture developed for the first TTP demonstrator meets the specifications in deliverables [D07] and [D09]. It works well.

2. *General applicability*

   This architecture was specific to the first implementation, and provides a free-standing demonstrator.

   For the future, the more general requirement is for an API to be called by a particular application requiring protected communication, or possibly a protection package called by the user which pre-requests keys for communication with address-book entries, say.

3. *Secure billing and trial requirements*

   It is required in the second demonstration and EXODUS based trial that the user and VASP will insert the TTP and secure billing trial smart card, and the TTP will have a card reader

4. *Enhancements*

   The following enhancements are suggested for the specification and implementation of the second demonstrator and the trial.

   - Check and improve the TTP architecture used in the first demonstration to match the architecture of ASPeCT entities in the secure billing trial.
   - Implement more complete and flexible TTP security information storage function.
   - Provide a TTP service API.

6.4 Appearance of demonstration

6.4.1 First demonstration's Graphical User Interface

The main objective of the first WP2.3 demonstration is to present a procedure for communication, supporting end-to-end encryption, through the use of TTP services. It comprises two clients, User A and User B, located in different domains and the relevant pair of TTPs, one for each domain. User A communicates securely with User B with the intervention of their respective TTPs which collaboratively perform the role of providing the users with key management services. Figure 6-7 illustrates the logical connections between the entities involved in the first demonstration.

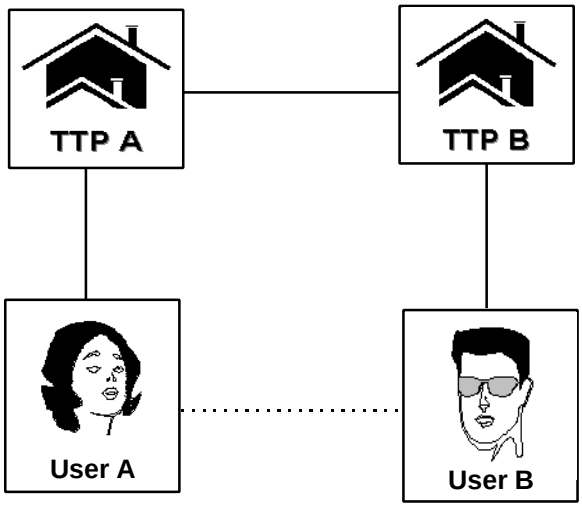


Figure 6-7 : The logical connections between entities

The GUI offers the possibility for the user to run one application, which simulates all the involved entities in a single PC, or to assign the entities over a number of applications, which may run on one or more PCs. In the latter case, the communication between the applications is provided via an Ethernet LAN, using the TCP/IP protocol.

The protocol involves the four participants, namely two users, one as a sender (user A) and the other as a receiver (user B) and their corresponding TTPs, exchanging seven messages, as shown in Figure 6-8, in order to establish secure communication. First the two TTPs establish a shared secret key among them (messages M1 and M2). Then, user A sends his TTP a request for communication with user B (message M3). In message M4, user A receives from TTP A the items he needs to prove his identity to the other party (a certificate) and compute the secret key that the two users will share. Now, user A is ready to send the certificate and his message, encrypted with the secret shared key, to User B, in M5. Finally, user B and TTP B exchange messages M6, M7, in order for user B to verify A's identity and compute the shared key.

This is the minimum amount of information that the demonstration user needs to know, in order to understand the interactions he will enable.

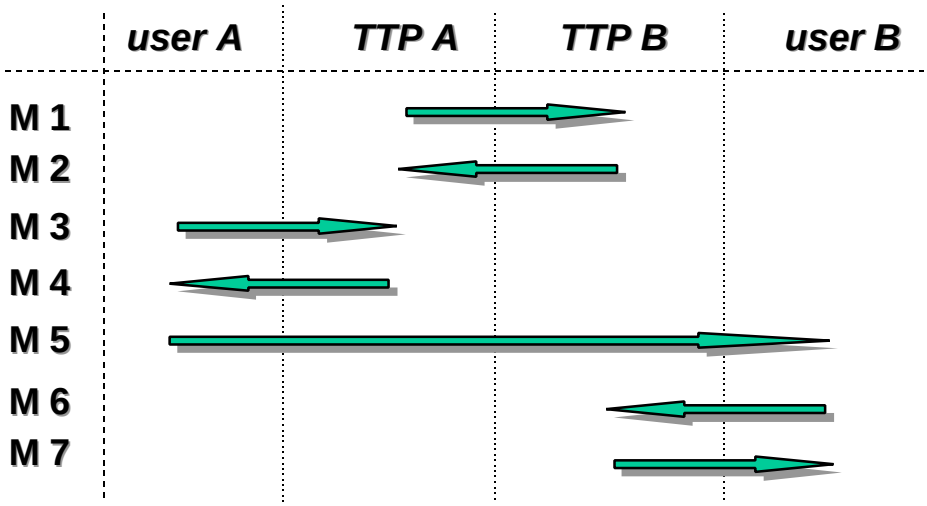


Figure 6-8 : Demonstrated message exchange

6.4.2 The observer's view

The role of the GUI is to provide the user with all the necessary means to initiate, observe and guide the end-to-end communication establishment process. From that point of view, the GUI must be unambiguous, detailed, informative and flexible against any improper user's selections.
In this first demonstration, the user is able to configure and view a series of important parameters about the communication database, such as:

- type mode (server or client),
- number of clients,
- server IP address, and
- client identity

and the involved entities, namely:

- state
- identity
- secret, public and shared keys, and
- certificate.

An overall graphical representation of the message flow is provided via the Monitor option, so that any viewer of the demonstration can immediately have an idea of what is going on.
The Tracer window, on the other hand, has the complementary role of providing the information needed for observing the internal process, including key generation, certificates, entities' state, etc., while the protocol is running. Thus, the information contained in the Tracer window is intended for users that are familiar with the protocol and the scope of the demonstration, while these data also enable all users to verify the successful execution of the secure communication establishment procedure.

The demonstration is entirely driven by the GUI. First, the GUI user has to configure the communication subsystem and then he must "create" and register the four entities involved. Then, through the entities' Action menu items, he can initiate and steer the process of the end-to-end confidential communication establishment. To achieve this, he must follow the correct sequence of steps, described in the user guide.

Error messages are displayed when an improper selection is made. There are also some indicative messages which appear in order to help the user follow the proper sequence of actions. Finally, there are informative messages that confirm the successful completion of some actions. Thus, the GUI provides information to the demonstration user on the impact of his actions.

6.4.3 Suggestions for enhancement

The current implementation requires that the demonstration user possesses either a minimum of related background or a small user guide. The addition of the on-line help will fulfil that requirement.
Some improvements can be made concerning the displayed messages in Monitor window, since now these are only indicative. This improvement would allow the message flow to be more intelligible for a project-unrelated user.

The Tracer window can also be enhanced to support options, for example to save or print the presented messages, so that they can be viewed off-line. An option initiating the automatic execution of the protocol can be added. This alternative will facilitate performance measurements and allow users to simply observe the procedure rather than conducting it.

6.5 User-friendliness

6.5.1 Quality of Service

Among many definitions, it can be generally assumed that Quality of Service (QoS) is determined by the user's perception of the degree to which the service meets, or surpasses, the need for which it is designed. It may also be defined by the level of overall user satisfaction regarding the provided service. QoS is defined in [ITU94] as follows: "The collective effect of service performance which determine the degree of satisfaction of a user of the service". This definition of QoS is a wide one, encompassing many areas. The QoS parameters, such as user satisfaction, are subjective in nature, depending on individual perceptions and expectations.

As derived from [ITU93], the user-oriented QoS parameters provide a valuable framework for design, but they are not necessarily practical for specifying performance requirements. Similarly, the performance parameters ultimately determine the user-observed QoS but they do not necessarily describe that quality in a way that is meaningful to users.

6.5.2 GUI features and suitability

The Graphical User Interface enables the user to configure the communication database and to observe and guide the end-to-end confidential communication establishment process. Thus, the GUI must be designed according to the user's needs. In order to estimate the GUI's friendliness for the user, several points are considered, such as:

The overall appearance: The GUI provides a simple visual representation of the demonstrated protocol through the Monitor bitmaps and an accurate record of all interactions through the Tracer window.
The flexibility: The demonstration is flexible regarding potential misuse by the users. There are messages that appear when the user makes inapplicable selections, in order to help them define the proper sequence of actions.
The sufficiency of the displayed features: The user has the options of configuring the communication database and the entities involved and watching the message flow in an abstract (Monitor option) or concrete way (Tracer option).
The degree of detail of the displayed information: The Tracer window enables the user to observe and understand the internal actions that are not presented in the main GUI window (key generation, certificates, etc.). Also, detailed information on the involved entities' features is included in the Status windows for the user's convenience.
The degree of guidance of the user on the steps he must follow: The menu bar is designed in a way that all the actions can be selected sequentially. The on-line Help option that will be available in the future is intended to include more extensive information about the GUI features and the recommended succession of user actions.
The security functions impact as perceived by the user: The duration of the security functions execution is not perceivable by the user.

6.6 Performance

6.6.1 Methodology

The performance of the first demonstrator was evaluated by making a number of timing measurements of the processing delays involved during execution of the protocol. Following the full protocol description in Appendix C (Section 8.3), the protocol was divided into four parts and the processing delays involved for each part were treated separately. Where timings were appropriate, estimates of the processing delays were made by measuring the delays between fixed measuring points. These measuring points are indicated in Figure 6-9 and Figure 6-10. Each timing was measured five times and an average measurement was computed and taken as a basic estimate of the processing delay. The results are shown in Tables 1 to 6.

All timings were made using an INTEL 486 33,4 Mhz.

6.6.2 Estimates of processing delays

Part 1: Share a secret between two TTP's

Part 1 can be conducted off-line, and thus no secure communication timings were taken for this part of the demonstrator operation.

Part 2 : Certificate generation in A's domain

The measurement points for Part 2 are indicated in Figure 6-9.

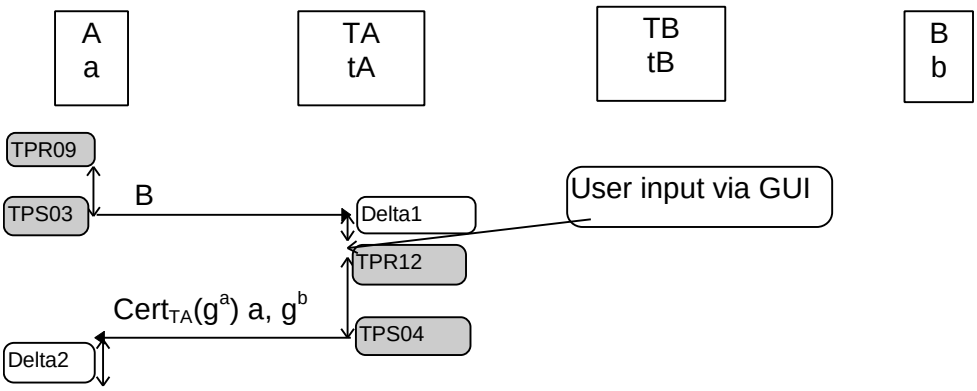


Figure 6-9 : Measurement points for Part 2

Two measurements are provided for Part 2 of the protocol. Firstly user A prepares a request to send to TA (TPR09 - TPS03) and then TA generates a certificate and key values for A. Due to a minor technical problem it was not possible to obtain estimates of the full processing time of the certificate generation process during Part 2. However most of this process was measured (TPR12 - TPS04) and the remaining time Delta1 represents the time taken to generate a user window in the demonstrator, and can be considered as negligible. A similar technical problem prevented a processing delay estimate being

calculated for the generation of the session key at user A. This measurement is represented in Figure 6-9 as Delta2 and corresponds to one exponentiation.

The measured delays during Part 2 are shown in Table 1 (seconds.milliseconds).

TPR09	TPS03	Time at A	TPR12	TPS04	Time at TA	Time at A and TA
19.570	19.630	0.060	26.990	37.640	10.650	10.710
21.380	21.380	0.000	50.760	01.470	10.710	10.710
04.310	04.310	0.000	15.510	26.220	10.710	10.710
18.620	18.620	0.000	41.360	52.070	10.710	10.710

Table 1 : Measured delays during Part 2

The average processing delay during Part 2 (excluding timings Delta1 and Delta2) is thus 10.710 seconds.

Parts 3 and 4 : Message transmission from A to B and certificate generation in B's domain

The measurement points for Parts 3 and 4 are indicated in Figure 6-10.

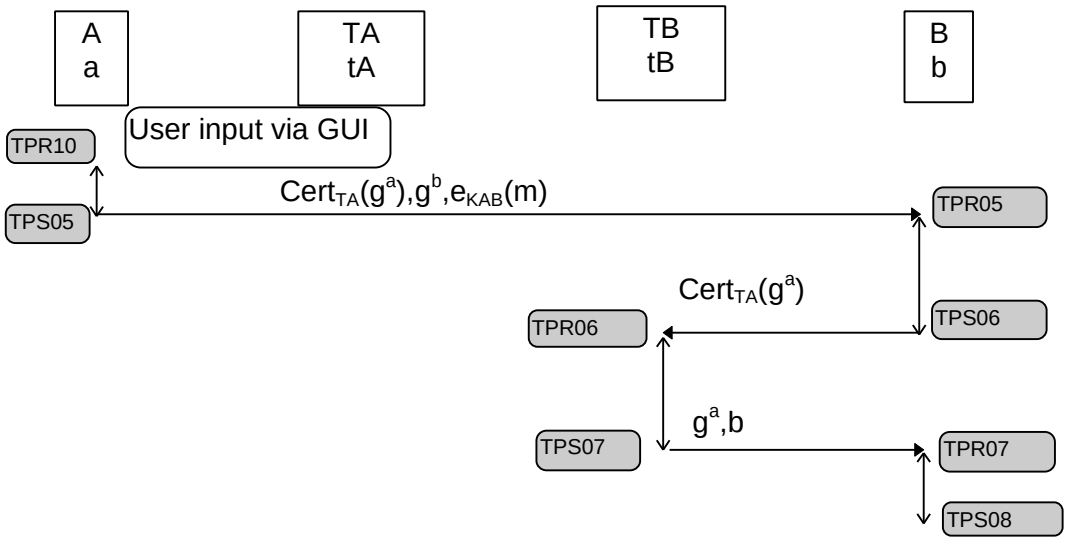


Figure 6-10 : Measurement points for Parts 3 and 4

The first measured delay was that of the sending user (user A) as they prepare the message to be sent to B (TPR10 - TPS05). This processing includes one encryption and the delays are shown in Table 2 (seconds.milliseconds).

TPR10	TPS05	Time at A
01.530	01.960	0.430
11.970	12.410	0.440
17.570	17.950	0.400
37.870	38.250	0.470
35.320	35.650	0.330

Table 2 : Delay at A during Parts 3 and 4

The average processing delay at A during Parts 3 and 4 is thus 0.414 seconds.

At the receiving end there are two sources of processing delay - user B and TB. The delay at TB (TPR06 - TPS07) is largely taken up by a certificate verification and the measured delays (seconds.milliseconds) are shown in Table 3.

TPR06	TPS07	Time at TB
03.610	14.930	11.320
13.950	25.320	11.370
19.490	30.860	11.370
39.950	51.380	11.430
37.300	48.670	11.370

Table 3 : Delay at TB during Parts 3 and 4

The processing delay at TB during Parts 3 and 4 is thus 11.372 seconds.

The processing delay at B consists of two stages. Firstly B extracts a certificate for the message received from A and forwards this to TB (TPR05 - TPS06). Then later B computes the session key from information received from TB and decrypts the message (TPR07 - TPS08). The measured processing delays at B are indicated in Table 4 (seconds.milliseconds).

TPR05	TPS06	Time at B	TPR07	TPS08	Time at B	Total time at B
02.020	03.500	1.480	15.200	19.930	4.730	6.210
12.410	13.900	1.490	25.540	30.320	4.780	6.270
17.950	19.380	1.430	31.080	35.800	4.720	6.150
38.360	39.850	1.490	51.490	56.320	4.830	6.320
35.710	37.190	1.480	48.830	53.560	4.730	6.210

Table 4 : Delay at B during Parts 3 and 4

The average processing delay at B during Parts 3 and 4 is thus 6.232 seconds.

Table 5 indicates the total processing delay involved at the receiving end (TPR05 - TPS08). This delay include processing at both user B and TB.

TPR05	TPS08	Time at B and TB
02.020	19.930	17.910
12.410	30.320	17.910
17.950	35.800	17.850
38.360	56.320	17.960
35.710	53.560	17.850

Table 5 : Total delay at B and TB during Parts 3 and 4

The average processing delay at B and TB during Parts 3 and 4 is thus 17,896 seconds.

6.6.3 Analysis

From Tables 1 to 5 in the previous section we can calculate the total measured processing delays at each of the entities involved in the first demonstrator. These are exhibited in Table 6.

A	TA	TB	B
0.414	10.700	11.372	6.232

Table 6 : Total processing delays at each entity during Parts 1 to 4

Remarks

1) There has been no attempt to provide best possible timings by running the demonstrator on suitably fast machines. These timings represent a first measurement of processing time and their main relevance is as an indication of the balance of the demonstrator performance. It is suggested that running the demonstrator on an INTEL pentium 75 Mhz would immediately reduce the measured timings by about 40%.

2) There is a marked difference between the total measured processing time at A and B. This difference may be explained by the combination of a couple of factors:
   - The total processing time at user A shown in Table 6 does not include the omitted time Delta2 during Part 2. Delta2 includes one exponentiation and is thus a significant measurement.
   - The processing delay at B of approximately 1.480 seconds between TPR05 and TPS06, shown in Table 4, is not caused by the execution of any cryptographic primitives. It is possible that further investigation will allow this processing delay to be significantly reduced.

The above remarks lead to two suggested enhancements for the second demonstrator. Firstly attempts could be made to obtain complete timings of the protocol using a faster machine and hence provide a more accurate estimation of the efficiency of the implementation. This would include the provision of a full set of appropriate measurement points for obtaining data concerning processing delays. Secondly, the full cause of the imbalance in the two user processing delay measurements will be investigated and the causes of the unnecessary delays will be attended to.

6.7 Applicability

The provision of end-to-end confidentiality services in the demonstrator relies on the establishment of a TTP-based key management infrastructure. It is possible that the requirement to support end-to-end confidentiality services alone will not sufficiently stimulate the development of a TTP infrastructure for UMTS. Instead, TTPs might need to offer a wider range of value-added security services.
The requirements for TTP-based security services in UMTS were investigated in ASPeCT Deliverable [D02]. In this section we discuss further the application of a TTP infrastructure to support UMTS security services in three areas:

- support for user-to-network mutual authentication;
- support for end-to-end security services;
- support for inter-network roaming.

6.7.1 Support for user-network mutual authentication

Many of the user-network mutual authentication schemes proposed for UMTS are based on public key cryptography. As such they require that the entities in the protocol can obtain authentic copies of the public keys they need. In this scenario a TTP could provide key management services to users, which would include the certification and provision of public keys. Typically, authentic copies of public keys could be made available as certificates generated and issued by a TTP acting as a certification authority.

In a simple scenario where the user just needs to authenticate with one network, authentic copies of public keys and associated private keys could be distributed by interacting with the TTP off-line. For instance, the keys could be distributed by the TTP as part of smart card pre-personalisation. However, in order to be able to enhance the level of security, the user or network may also require on-line access to a TTP in order to request updates on public keys, or to check or request appropriate revocation lists. In the more general case, the user will need to be able to register and authenticate with different networks, with which it has had no previous security association. In this case the relevant authentic public keys could be made available through a trusted infrastructure of TTPs acting as certification authorities and certificate servers.

6.7.2 Support for end-to-end security services

The TTP infrastructure can allow two users who have had no previous security association to establish secure communications by receiving trusted services from their respective TTPs. A TTP infrastructure could be used in support of a wide range of security services for end users. In the following we describe the role of TTPs in support of some of these services.

6.7.2.1 Confidentiality

A TTP infrastructure can provide the means to allow UMTS users to establish confidential communication channels with other users, possibly in different countries, whilst being able to satisfy law enforcement requirements at both the national and international level. This is the service provided in the demonstrator.

In the demonstrator a TTP-based key management scheme allows the users to establish a shared secret confidentiality key for use in a symmetric cryptosystem. An important feature of the scheme is that some information used to generate the shared secret confidentiality key is escrowed to the TTPs. Thus, an interception agent can, under certain prescribed conditions, obtain access to communications between the users by presenting a valid warrant to an appropriate TTP, who releases the escrowed information.

Note that it is important that only keys used for confidentiality can be recovered from the TTP. It should not be possible for an agent to obtain secret signature keys, or keys used for integrity protection, for example. Therefore, care must be taken in distinguishing TTP services that support confidentiality, from TTP services that support integrity, authentication, access control and non-repudiation.

6.7.2.2 Data integrity and origin authentication

A TTP infrastructure could also be used to help users establish a shared secret integrity key for use in generating a message authentication code (MAC). This type of service would allow users to protect their messages against unauthorised modification. It also allows receivers to be sure of the true identity of the sender.

Digital signatures could also be used to support message integrity and origin authentication. In this case the TTP infrastructure could be used to manage the asymmetric key pairs required to support the digital signature scheme. Typically, the TTPs would act as certification authorities and certificate servers for the public keys to be used in the scheme.

6.7.2.3 Entity authentication

It is likely that some users and services may require a mechanism whereby end users of a UMTS telecommunications service can mutually authenticate each other. For basic telephony services this is unlikely to be a requirement. However, for non-telephony services, where users interact with services provided by other users acting as value-added service providers, a cryptographic means of user-to-user authentication may be a strong requirement. Note that in implementing user-to-user authentication, it may be desirable to re-use the user-network mutual authentication mechanism.

In supporting entity authentication a TTP can be used as an authentication server in schemes that involve third parties. The services can be categorised according to whether the authentication server is on-line, off-line or in-line.

6.7.2.3.1 On-line authentication

In symmetric authentication schemes, there is a requirement for every verifier to maintain a secret symmetric key with every claimant. This may not be practical for end-to-end user authentication. Instead, a trusted on-line authentication server could be introduced. This would share a secret key with every claimant and verifier. Two general approaches to this scheme are outlined below:

- In the first approach, the claimant encrypts or seals a message with its secret key and sends it to the verifier. Since the verifier does not share the claimant's key, it must obtain it through a separate exchange with the server. This exchange should be secured using a shared key between the server and the verifier.
- In the second approach, the claimant first obtains a ticket from the server which contains a secret key to be used by the claimant to authenticate itself. The communication with the server is protected using a secret key shared between the server and the claimant. After the exchange, the claimant authenticates itself by encrypting or sealing a message with the secret key it received from the server and sending it to the verifier for checking.

The major disadvantage of this approach is that the users require on-line access to the TTP during the authentication process. This may not be practical for end-to-end user authentication in UMTS.

6.7.2.3.2 Off-line authentication

With asymmetric authentication the need for the authentication server to be on-line is removed. Instead, verifiers can obtain certified public keys for claimants and certificate revocation lists from an off-line server, prior to or during authentication. This information can be cached and reused to avoid having to communicate with the server each time authentication is initiated. However, if a user wants to be absolutely certain that a certified public key has not been revoked it should verify the status of the certificate or the certificate revocation list with the server. An off-line authentication server may act simply as a directory server for certificates generated by another TTP acting as a certification authority.

6.7.2.3.3 In-line authentication

In-line authentication involves an authentication server positioned in the communication path between claimant and verifier. The authentication is, in effect, split into two sets of interactions. Firstly, the claimant attempts to authenticate itself to the server, which vouches for the identity of the verifier. Secondly, the verifier attempts to authenticate the server, which vouches for the identity of the claimant.

The major disadvantage of this approach is that the authenticated channel is set up via a TTP, therefore subsequent messages passed between the users may have to be sent via the TTP. This may have a serious impact on performance in a UMTS environment.

6.7.2.4 Access control

Users may wish to obtain access privileges in order to be able to access services or data provided by other end users. In this scenario a TTP infrastructure may be used to certify and manage privileges for access control. Certified privileges may be obtained from a TTP in the form of a Privilege Attribute Certificate (PAC).

The PAC will contain a list of resources that the user may wish to access and the associated privilege level that the user has been assigned. A user will send a request for a given privilege attribute to a TTP, which will then identify and authenticate the user. If the current security policy states that the user is authorised to make the requested access, then the TTP will generate and certify the privilege attribute. The privilege attribute certificate is then distributed by the TTP by publishing it in a directory.

The veracity of privilege information can be checked on demand by anyone who obtains the TTP's public certification key. The PAC may need to be revoked if changes in access control privileges are required, or if a compromise of sensitive information is suspected.

6.7.2.5 Non repudiation

It is possible that end users will require protection against repudiation of origin or delivery of messages sent to other uses. This type of service may be particularly important where a user interacts with another user, acting as a value-added service provider, in order to receive a chargeable service. In this case, in order to ensure that charging is incontestable, the parties involved should have the assurance that justified claims relating to charges can be proved and that unjustified claims cannot be successfully made. A non repudiation service can be used to provide incontestable charging by generating evidence which can be used to prove that the user invoked certain chargeable actions.
Non repudiation involves the generation, verification and recording of evidence about a particular event or action, and the subsequent retrieval and re-verification of this evidence in order to resolve disputes. A TTP infrastructure may have an important role to play in each phase of a non-repudiation service.

Evidence may be generated by digitally signing information related to a specific event or action. Information may include the identities of the entities involved, the communicated data, and the date and time. Additional information that may be required could include the mode of transfer, the location of entities or the creator of data. The evidence may be generated by the evidence subject, perhaps in conjunction with a TTP, or by a TTP alone. In the case where digital signatures are used, a TTP infrastructure could be used to manage the asymmetric key pairs require to support the digital signature scheme.
A TTP may also have to record evidence in a non-repudiation role so that it can be retrieved by an evidence user or adjudicator. The evidence to be recorded will typically be received via an interface with the network over which the communication entities are sending messages. The recording process may involve the addition of a time-stamp and certain user-related information.

The purpose of evidence verification is to provide the evidence user with the confidence that the supplied evidence will be adequate in the event of a dispute. The evidence verifier may be the evidence user, or a TTP trusted by the evidence user. For evidence generated using digital signatures, evidence verification will involve verification of the digital signature. In dispute resolution, a TTP could act as an adjudicator making a decision, based on evidence collected, to resolve the dispute.

6.7.3 Support for inter-network roaming

In UMTS it is likely that there will be an extremely large number of network operators providing a wide range of mobile telecommunication services to users. The requirement to provide global coverage will necessitate that UMTS users can roam freely between networks. It is assumed that in order for a user to

be able to roam onto a given network, his service provider must have some kind of roaming agreement established with that network.

In GSM roaming is facilitated by a set of procedures laid out by the GSM MoU Association. In particular, the MoU specifies procedures for establishing written bi-lateral roaming agreements between MoU members. Procedures also exist to facilitate the establishment and testing of signalling links between networks. Currently, the management and administration of roaming agreements in GSM is manageable because each operator operates under the control of the MoU, which effectively acts as a regulatory body. Moreover, the number of operators involved is relatively low. Thus, there is a relatively high degree of trust among operators which allows roaming agreements to be established more easily.

However, it cannot be assumed that all UMTS operators world-wide would be willing to join such an MoU. Also, the cost and complexity of roaming trials and agreements between a large number of operators world-wide would be very difficult to support. Even within the GSM community, it is widely accepted that a complete set of bi-lateral roaming agreements between every GSM network will never be achieved due to the administration and costs involved. Currently, of the approximately 20,000 potential world-wide roaming agreements, only around 1,500 are currently in place, and of those there are a significant number whose volume of traffic do not justify their maintenance costs.
The establishment of roaming agreements will become even more cumbersome in UMTS as the number of network operators and service providers increases. Moreover, the trust relationships between the network operators and service providers will become more complex because of the potential lack of a regulatory body similar to the GSM MoU and because of the number of entities involved. Therefore, it is likely that alternative mechanisms will be required for UMTS. One approach may be to make use of a TTP infrastructure to help manage the trust relationships between network operators and service providers and thus support the establishment of roaming agreements.

6.7.3.1 Mechanisms to facilitate roaming in GSM

In GSM various mechanisms have been established in order to reduce the cost of setting up roaming agreements between network operators. For instance, clearing houses can be introduced to reduce the cost associated with exchanging billing information. This significantly reduces the cost in cases where small individual amounts of billing information need to be sent to a large number of operators. In this way an operator need only interact with one clearing house in order to exchange billing information with many operators.

The cost of establishing roaming between networks can also be reduced by introducing an intermediary which acts as an relay point for signalling information sent between networks. Thus, instead of setting up bi-lateral signalling links with many other networks, links would only need to be established with a relay point. This would significantly reduce the cost associated with the establishment and subsequent testing of signalling links.

In both scenarios the network operator would need to set up agreements with the clearing house or signalling relay point in order to negotiate and agree conditions. If the roles of clearing house and signalling relay point are combined then the combined agreement can effectively replace bi-lateral roaming agreements between operators.

Recently, within the GSM community, a scheme has been established, which facilitates the establishment of roaming agreements through the use of a combined International Roaming Platform (IRP) to handle signalling traffic and a clearing house facility to handle the exchange of billing information [GCS97]. The *Global Cellular Service* involves the establishment of written agreements between the network operator and the entity offering the service. This means that individual bi-lateral

roaming agreements between network operators are no longer required. This is a particularly attractive solution in cases where the amount of roaming traffic is low.

6.7.3.2 The application of a TTP infrastructure to facilitate roaming in UMTS

We now consider how a TTP infrastructure could be used to facilitate roaming in UMTS.

6.7.3.2.1 Support for electronic roaming agreements

In UMTS it would be desirable to avoid having to exchange written bi-lateral agreements as part of establishing roaming between a service provider and a network operator. For instance, it might be desirable for roaming agreements to be set up dynamically, as and when they are required (e.g. on a per call basis). Clearly, this cannot be implemented if the exchange of written documents is required.
In the first instance we consider how electronic roaming conditions could be negotiated and agreed off-line. We then consider how electronic agreements could facilitate the on-line establishment of roaming.

Before negotiating and establishing a roaming agreement electronically, a network operator and service provider will have to mutually authenticate each other. After authentication has been performed roaming conditions could be negotiated and agreed. When an agreement has been reached, the service provider could commit to the conditions by signing them using a digital signature. To support this procedure a network of TTPs could be used to provide a certification infrastructure which would enable network operators and service providers to mutually authenticate each other, and subsequently allow service providers to sign a roaming agreement.

This procedure could also facilitate the on-line establishment of roaming agreements. In this case we assume that a UMTS subscriber, wishing to gain access to a network, sends the identify of its service provider to the network. The network could then initiate the on-line establishment of a roaming agreement with the user's service provider using the electronic procedures described above. The roaming agreement may be established for this particular user access, or it may apply to subsequent accesses by any user of the particular service provider concerned.

If the roaming agreement only applies to particular user accesses, then the network operator could give the service provider a statement containing specific call related information (e.g. the identity of the user, the charging rate and a time stamp). If the service provider is satisfied with the terms of the statement he will sign it with his unique digital signature and this will be his commitment to pay for the services provided to the subscriber. However, if the roaming agreement applies to subsequent user accesses, then the agreement will specify the long term roaming conditions which apply.

Further study is required into an appropriate certification infrastructure based on TTPs for the support of electronic roaming agreements in UMTS. It is also necessary to study the feasibility of the on-line establishment of roaming agreements in UMTS.

6.7.3.2.2 Reducing the number of bi-lateral agreements

In UMTS it is likely that in many cases the amount of roaming traffic associated with a particular service provider / network operator pair is unlikely to justify the cost of managing and maintaining individual bi-lateral roaming agreements. Thus, it may be desirable to avoid having to establish individual bi-lateral roaming agreements between every network operator / service provider pair.
One approach to reduce the cost of establishing individual bi-lateral roaming agreements may be to establish roaming agreements only when they are required. In this case roaming agreements may apply to individual accesses to a network, or they may apply to all subsequent accesses by users of the particular service provider concerned. The choice here will depend on the likely amount of roaming

traffic associated with the particular service provider / network operator pair. This approach assumes that roaming agreements can be established on-line, as described in Section 6.7.3.2.1 above.
An alternative, but potentially complementary approach, for reducing the cost of managing and maintaining individual agreements, would be to group service providers and network operators in TTP domains. Instead of establishing individual bi-lateral agreements between each network operator / service provider pair, agreements would be established between a TTP and each of the service providers and network operators in its domain.

For each service provider, the agreement would specify the conditions to be applied by each network it wishes its users to be able to roam with. For each network operator the agreement would specify the conditions to be applied by each service provider whose users are permitted to roam on the operator's network. Thus, when a user attempts to access a network, the appropriate roaming agreement can be obtained and verified by the network operator. In the case where more than one TTP is involved, it may be necessary to verify a chain of certificates before being able to verify the appropriate roaming agreement.

TTPs offering this type of service could also act as clearing houses for billing information or relay points for signalling information in a similar way to that described in Section 6.7.3.1 above.

7. Summary of suggested enhancements

In summary, the first TTP demonstrator meets the specifications of [D07] and [D09] and runs well. There are however a number of suggested enhancements for the implementation of the second demonstrator. These have been described throughout Section 6 and we provide a brief summary here.

The most significant direction to future enhancements is provided by the need to integrate with the secure billing demonstrator and the trial with EXODUS. Further study is needed before it can be precisely stated what changes to the current demonstrator are needed to successfully navigate this integration process, however some likely enhancements are as follows:

- ***Key escrow protocol***. The protocol implemented in the first demonstrator is unlikely to be perfectly suitable for implementation in the second demonstrator. Clarification of likely escrow scenarios is necessary. Either the existing protocol will be subject to some suitable refinements or alternative protocols will be considered. If the existing protocol is adapted then some extra provision for time-bounded interceptions is a likely enhancement.
- ***Security Aspects***. More TTP services, functions and internal operations could be provided. Suggested services include key revocation and certification for access control and time-stamping. These services are a priority because they are services likely to be demanded on integration with the secure billing demonstrator.
- ***Architecture***. The architecture should be matched with that of the secure billing trial. A more complete and flexible TTP security information storage function should be provided as well as a TTP Service API.
- ***Demonstrator Appearance***. Suggested enhancements include the addition of an on-line help, improvements in Monitor message display, extra Tracer options to permit off-line viewing and an automatic protocol execution option. Note that other minor refinements to the first demonstrator could include an explicit demonstration of key recovery (rather than an implicit one) and an option

to slow down certain screen operations (such as message arrow display) to ease visual comprehension of the protocol message sequence.

- ***Implementation***. Implementation refinements arising from analysis of the first demonstrator should be incorporated into the second demonstrator.

8. Appendix

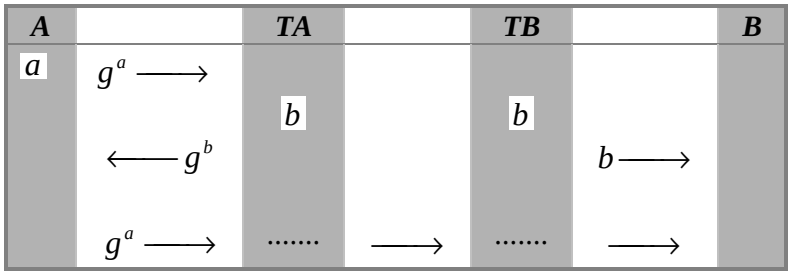
8.1 Appendix A: Two-way key escrow protocols

This appendix contains supplementary information relevant to Section 6.1.5.3.

We consider here some simplified protocols that have appeared in ASPeCT related documents. In the following protocols user A establishes a session key for communication with user B. Each user has a home TTP (TA for A, and TB for B) with which they can communicate (securely). Some of the protocols require a secure link between TA and TB. Note that we have simplified all the protocols for ease of comparison.

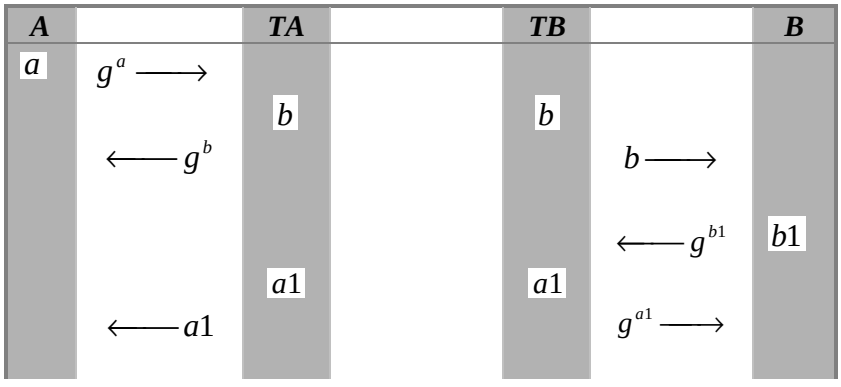
In the illustrations of the protocols a value appearing in the column of entity X means that entity X generates that value. An arrow and values between column X and column Y indicate transfer of values between entity X and entity Y (dotted lines in the column of entity X indicate that the value being transferred is not sent to that entity).

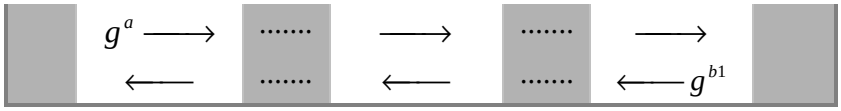
Protocol IA



Protocol 1A is an earlier version of the protocol in [JMW96b]. At the end of the exchange A and B can both compute the key g^{ab}. In Protocol IA, b is a function of the identities of TA, TB and B. This protocol is a one-way protocol and we simply adopt the one-way key as a two-way key.

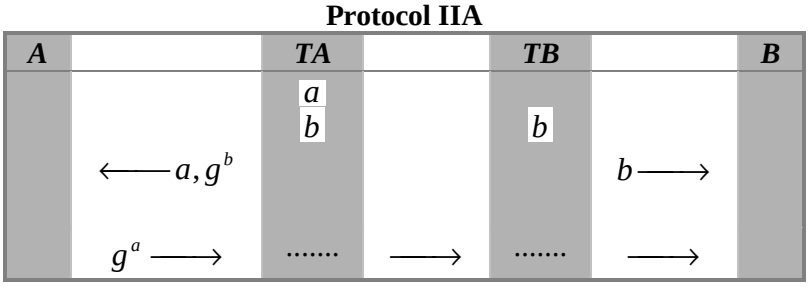
Protocol IB





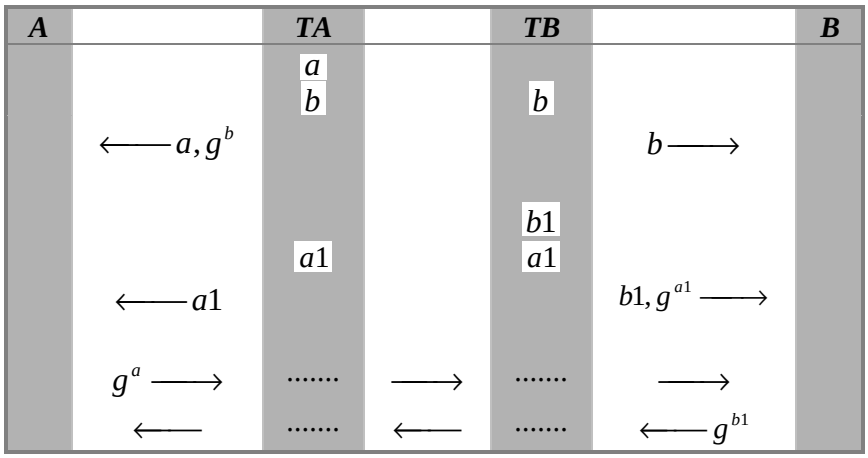
Protocol IB simply consists of two runs of Protocol IA, one in each direction, and a subsequent combination of the two one-way keys. At the end of the above exchange A and B can both compute the keys g^{ab} (for communication from A to B) and g^{a1b1} (for communication from B to A). Two-way communication could now be established by using some combination of these two keys, perhaps $g^{ab} + g^{a1b1}$. In Protocol IB, b is determined as a function of the identities of TA, TB and B, and $a1$ is determined as a function of TA, TB and A.

Protocol IIA



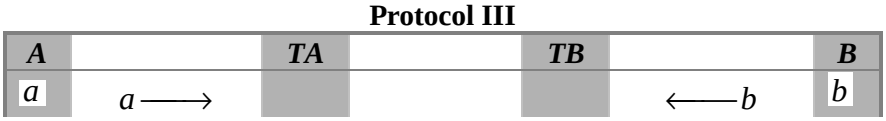
Protocol IIA is the original one-way protocol of [JMW96b]. At the end of the previous exchange A and B can both compute the keys g^{ab}. In Protocol IIA, b is a function of the identities of TA, TB and B. We simply adopt the one-way key as a two-way key.

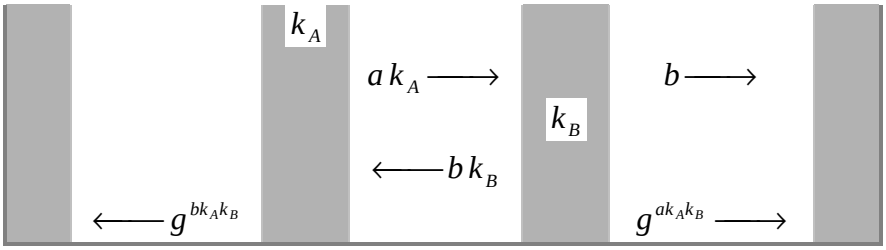
Protocol IIB



Protocol IIB simply consists of two runs of Protocol IIA, one in each direction, and a subsequent combination of the two one-way keys. At the end of the above exchange A and B can both compute the keys g^{ab} (for communication from A to B) and g^{a1b1} (for communication from B to A). Two-way communication could now be established by using some combination of these two keys, perhaps $g^{ab} + g^{a1b1}$. In Protocol IIB, b is determined as a function of the identities of TA, TB and B, and $a1$ is determined as a function of TA, TB and A.

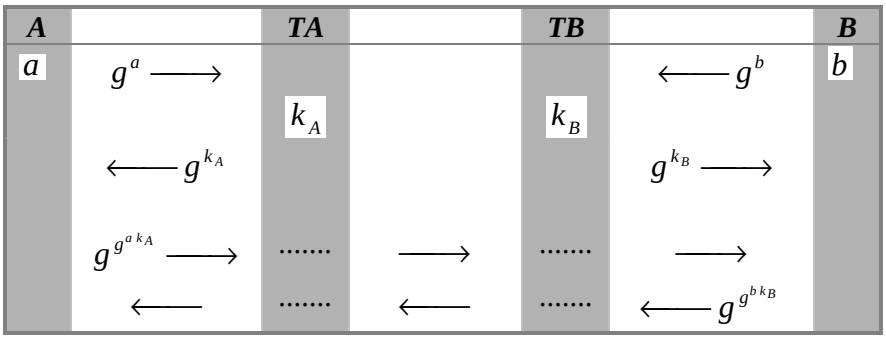
Protocol III





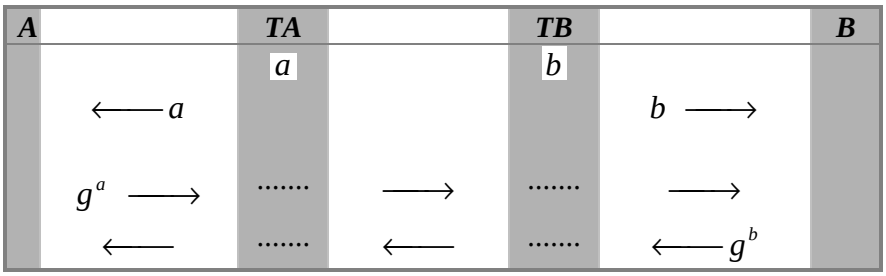
Protocol III is a reduction of the protocol of [CGM96]. At the end of the above exchange A and B can both compute the key $g^{abk_Ak_B}$. In Protocol III, k_A is determined as a function of the identities of TA, A and B, and k_B is determined as a function of TB, A and B.

Protocol IV



Protocol IV is a reduction of the protocol of [CM96]. At the end of the exchange A and B can both compute the key $g^{g^{ak_A+bk_B}}$. In Protocol IV, k_A is a function only of the identity of TA, and k_B is determined as a function only of TB.

Protocol V



Protocol V is a basic escrow version of the original Diffie-Hellman key exchange [DH76]. At the end of the above exchange A and B can both compute the key g^{ab}.

First we make some comments on parameter differences among the protocols (see Section 6.1.2.1).

Communication Structure

- All protocols need links between A and TA, B and TB and between A and B.
- All protocols except Protocol IA and Protocol IV require a secure link between A and TA.
- All protocols except Protocol IV require a secure link between B and TB.
- Protocols IA, IB, IIA and IIB require a secure (off-line) link between TA and TB.
- Protocol III requires an on-line secure link between TA and TB.

Cryptographic flexibility: Key Generation

- In Protocol IA the session key is generated jointly by A, TA and TB.
- In Protocols IB, III and IV the session key is generated jointly by A, B, TA and TB.
- In Protocols IIA, IIB and V the session key is generated jointly by TA and TB.

Interception safeguards

In each protocol an interception authority can obtain the session key by approaching either TTP. We consider possible releases of information for the various protocols.

Protocol IA :

- TA or TB release b . All messages between users with home TTP TA and B (initiated by the user with home TTP TA) can be read for the lifetime of key b ;
- TA or TB release g^{ab} . All messages from A to B (initiated by A) can be read for the joint lifetime of keys a and b .

Protocol IB :

- TA or TB release b and $a1$. All messages between A and B can be read for the joint lifetimes of keys b and $a1$;
- TA or TB release $g^{ab} + g^{a1b1}$. All message between A and B can be read for the joint lifetimes of keys $a,b,a1$ and $b1$.

Protocol IIA :

- TA releases a . All messages between A and any user (initiated by A) can be read for the lifetime of key a ;
- TA or TB release b . All messages between users with home TTP TA and B (initiated by the user with home TTP TA) can be read for the lifetime of key b ;
- TA or TB release g^{ab} . All messages from A to B (initiated by A) can be read for the joint lifetime of keys a and b .

Protocol IIB :

- TA release a and $a1$. All messages between A and B can be read for the joint lifetime of keys a and $a1$;
- TB release b and $b1$. All messages between A and B can be read for the joint lifetime of keys b and $b1$;
- TA or TB release b and $a1$. All messages between A and B can be read for the joint lifetime of keys b and $a1$;
- TA or TB release $g^{ab} + g^{a1b1}$. All messages between A and B can be read for the joint lifetimes of keys $a,b,a1$ and $b1$.

Protocol III :

- TA or TB release $g^{abk_Ak_B}$. All messages between A and B can be read for the joint lifetime of keys a,b,k_A and k_B .

Protocol IV :

- TA releases k_A. All messages between A and B can be read for the lifetime of key k_A;

- TA releases g^{ak_A} .All messages between A and B can be read for the lifetime of keys a and k_A;
- TB releases k_B. All messages between A and B can be read for the lifetime of key k_B;
- TB releases g^{bk_B} .All messages between A and B can be read for the lifetime of keys b and k_B.

Protocol V :
- TA releases a . All messages between A and B can be read for the lifetime of key a ;
- TB releases b . All messages between A and B can be read for the lifetime of key b

The most significant differences among the five protocols occur with respect to the communication structure. Protocol III requires on-line secure communication between TA and TB which for many situations, including within ASPeCT, is fairly impractical. Protocol IV in contrast requires no secure links between any of the entities. Note that our basic ASPeCT communication scenario does not require this lack of secure links.

There does not seem to be a great deal of difference between Protocols IB, IIB, III, IV and V with respect to types of interception possible. All only obviously permit interception of messages directly between users A and B (although in Protocols IV and V wider interceptions might be possible depending on the scope of use of key information generated by TA and TB). If a wider interception is required then extra information will have to be released. In contrast Protocols IA and IIA allow certain wider interceptions to take place. In this respect it can be claimed that Protocols IA and, particularly, IIA are more efficient from the point of view of interception authorities.

The extra complexity of Protocols IB and IIB does not seem to have any distinct advantages over the simpler Protocol V. Indeed Protocol IV has the additional attraction of reducing the use of secure links between the main entities. Neither do there seem to be any significant advantages of using Protocol IB over IA, and of using Protocol IIB over IIA. The notable difference between Protocols IB and IA is that both users can contribute to the generation of the session key in IB, but as B trusts TB (who contributes to the key generation in Protocol IA) this does not seem a significant gain. These remarks are summarised in Section 6.1.5.2.

8.2 Appendix B: Some alternative escrow protocols

This Appendix provides details relevant to Section 6.1.6.

8.2.1 LWY protocol

The following protocol is referred to as the *LWY protocol* and was proposed in [LWY95].
Let p and q be large primes, such that q divides $p-1$, and let g be an element of order q (modulo p). Let $S(x), P(x)$ be the secret and public keys of X, where $P(x) = g^{S(x)}$ (modulo p). Let E be a secure block cipher and H be a one-way hash function. Values p, q, g, E and H are all public.

In advance of communications

1. User A selects a TTP in each domain that requires access to the future communication. Let A select TTP TA, and let B select TTP TB.
2. User A generates $S(a), P(a)$, while user B generates $S(b), P(b)$.
3. $S(a)$ is escrowed to TA , while $S(b)$ is escrowed to TB.

Note that in step 2 we assume that the users generate their own secret and public keys. However, in [LWY95] it is not specified who generates these keys.

At time of communication

1. A computes the following:

$$K(a,b,d) = H(P(b)^{S(a)}, d)$$
$$S(a,d) = H(S(a), d)$$
$$S(a,b,d) = H(S(a,d), P(b))$$

2. B computes the following:

$$K(b,a,d) = H(P(a)^{S(b)}, d)$$
$$S(b,d) = H(S(b), d)$$
$$S(b,a,d) = H(S(b,d), P(a))$$

3. A sends $E_{S(a,b,d)}(K(a,b,d))$ to B.
4. B sends $E_{S(b,a,d)}(K(b,a,d))$ to A.

Users A and B can now exchange messages using session key $K(a,b,d) = K(b,a,d)$.

Key recovery

Several types of interception are possible using the LWY protocol:

- TA releases $S(a)$. All messages from or to A can be read for the lifetime of $S(a)$.
- TA releases $S(a,d)$. All messages from or to A can be read for the joint lifetime of $S(a)$ and date-stamp d.
- TA releases $S(a,b,d)$. All messages between A and B can be intercepted for the joint lifetime of $S(a), S(b)$ and date-stamp d.
- TB releases $S(b)$. All messages from or to B can be read for the lifetime of $S(b)$.
- TB releases $S(b,d)$. All messages from or to B can be read for the joint lifetime of $S(b)$ and date-stamp d.
- TB releases $S(b,a,d)$. All messages between A and B can be intercepted for the joint lifetime of $S(a), S(b)$ and date-stamp d.

8.2.2 IBM protocol (SecureWay)

The infrastructure of SecureWay [IBM97] relies on the existence of a number of *key recovery service providers (KRSPs)* in each domain. In the documentation an amount of stress is placed on suggesting that these service providers are not *trusted third parties (TTPs)*. We note however that the KRSPs do play a very similar role to TTPs in other protocols. In each case TTPs (KRSPs) hold some secret information that is on its own necessary but not sufficient to determine the session key.
Assume that user A wishes to send a message to user B. We briefly describe how to send a message using SecureWay.

In advance of communication

1. User A selects a subset of KRSPs in each domain that requires access to the future communication. For simplicity let A select two KRSPs in each of the sending and receiving domains; label these SPA1, SPA2 and SPB1, SPB2.
2. For each designated KRSP, users A, B and the KRSP mutually agree on a "random" number. This random number (one for each designated KRSP in each domain) is fixed over a number of different communication sessions.

Note that it is suggested that Step 2 could be achieved by means of a "three-way" Diffie-Hellman key exchange. Another option is for A and B to exchange a random seed that is then used to pseudorandomly derive the random numbers, and for them to pass them on to the KRSPs using a public key encryption method.

At time of communication

1. User A selects a symmetric encryption algorithm E.
2. User A selects a session key K.

3. For each random number $r_{A1}, r_{A2}, r_{B1}, r_{B2}$, user A computes a secondary parameter (by some publicly known procedure that takes as input the random number and, possibly, date, address details, encryption method etc.). Denote the results $k_{A1}, k_{A2}, k_{B1}, k_{B2}$.
4. A sends to B: $E_{k_{A2}}(E_{k_{A1}}(K)),\ E_{k_{B2}}(E_{k_{B1}}(K)),\ E_K(m),$ where m is the message.

On receiving

1. User B decrypts the message. To do this B must already have a copy of session key K. It is not precisely specified how B computes this key, but one option is by a key agreement exchange with A prior to communication.
2. User B checks the key recovery information (B can do this as B knows each of the random number seeds from which the secondary parameters are computed).

Key recovery

The process of key recovery is identical in each domain. For example, for the sending domain:

1. Interception authority approaches SPA1 and SPA2 with an interception warrant and details of the communication to be intercepted.
2. SPA1 computes and releases k_{A1}, SPA2 computes and releases.
3. Interception authority decrypts (in reverse order) the nested encryption of K.
4. Interception authority decrypts the message.

8.2.3 VKT protocol (Binding cryptography)

The concept of *binding cryptography* was discussed in [VKT97] and [VT97]. Binding cryptography can be thought of as a type of escrow protocol and it has some attractive properties that make it worth considering for ASPeCT.

Let p be a large prime and G be the multiplicative group Z_p^*. Let g be a generator of Z_p^*. Let $S(x), P(x)$ be the secret and public keys of user or TTP X, where $P(x) = g^{S(x)}$ (modulo p). The parameters p and g are public. The protocol is based on the ElGamal public key encryption system [ElG85].

In advance of communications

1. User A selects a TTP in each domain that requires access to the future communication. Let A select TTP TA, and let B select TTP TB.
2. User agrees ElGamal parameters with its TTP.

At time of communications

1. User A generates a random r
2. User A sends the following to user B:

$g^r, P(b)^r K$

$g^r, P(ta)^r K$

$g^r, P(tb)^r K$

binding data

Values 1, 2 and 3 represent ElGamal encryptions of K using the public keys of B, TA and TB respectively. The *binding data* is the text of a zero-knowledge proof that enables anyone to test that the three encryptions have all been performed using the same parameter r (and thus represent valid ElGamal encryptions for the three entities). Details of the construction of the binding data can be found in [VT97].

On receiving

1. User B decrypts the session key using his secret ElGamal key
2. User B checks the key recovery information (B can do this as B knows the public ElGamal keys for the TTPs TA and TB)

Key recovery

Only one type of interception is possible:

- TA or TB releases K. All messages from A to B can be intercepted for the lifetime of K.

8.2.4 Parameters of alternative protocols

8.2.4.1 Communications structure

Protocol	Description of parameter
LWY	An off-line link is required between A and TA and between B and TB in order to escrow secret key with TTP. In addition, an off-line link is required between TA and TB such that all entities can obtain (certified copies of) the public keys of A and B. During interception the agents needs to obtain relevant information from the TTPs in their domain in order to intercept the communications. Co-operation or agreements between TTPs are not required (although TTPs need to be able to obtain public keys of users belonging to other TTPs).
IBM	An off-line link is required between A and its KRSPs and between B and its KRSPs in order to agree on a random number, which is fixed over a number of sessions. During interception the agents must operate very closely with the relevant KRSPs to obtain the information necessary to decrypt each session key. Co-operation or agreements between TTPs are not required.
VKT	An off-line link is required between A and TA and between B and TB in order to let users obtain the relevant (certified) public ElGamal keys. During interception the agents needs to obtain relevant information from the TTPs in their domain in order to intercept the communications. Co-operation or agreements between TTPs are not required.

8.2.4.2 Trust relationships

Protocol	Description of parameter
LWY	The users and TTPs in this scheme have the same trust relationships as the users and TTPs in the JMW scheme.
IBM	The users and KRSPs in this scheme have similar trust relationships as the users and TTPs in the JMW scheme. The difference is that the basic IBM scheme allows for key splitting. Thus, the user need not trust its KRSPs individually, but must however trust them collectively.
VKT	The users and TTPs in this scheme have the same trust relationships as the users and TTPs in the JMW scheme.

8.2.4.3 Interception safeguards

Protocol	Description of parameter
LWY	TTPs hold information which can be used to generate the session key. Time-boundness is incorporated into the basic scheme. Many types of interceptions are possible. These include efficient mechanisms for performing node-based interception.
IBM	KRSPs hold information which on its own is not sufficient to generate the session key. Instead, the agent needs to recover information from the messages sent from/to the targeted user(s).

	The scheme naturally incorporates key splitting, thus the agent may have to approach more than one KRSP before being able to intercept the communication. Time-bounded interceptions are possible, but at the expense of a greater complexity of interception. Only one type of interception possible - KRSPs release information which can be used to help compute the session key. Thus, efficient node-based interceptions are not possible. The receiver can check the correctness of the key recovery information. However, this process is more complex than VKT.
VKT	TTPs hold information which on its own is not sufficient to generate the session key. Instead, the agent needs to recover information from the messages sent from/to the targeted user(s). Key splitting is not featured in the basic scheme. Time-bounded interceptions are possible, but at the expense of a greater complexity of interception. Only one type of interception possible - TTPs release information which can be used to help compute the session key. Thus, efficient node-based interceptions are not possible. The receiver can check the correctness of key recovery information. Before an interception, a third party can verify that key recovery information has been correctly computed.

8.2.4.4 Escrow type

Protocol	Description of parameter
LWY	Session key only
IBM	Session key only
VKT	Session key only

8.2.4.5 Cryptographic flexibility

Protocol	Description of parameter
LWY	Session key is jointly generated by A, B, TA and TB.
IBM	Session key generated by user. Flexible with regard to key recovery demands of different domains. Symmetric algorithm used to compute key recovery information can be different in each domain.
VKT	Session key generated by user. Flexible with regard to key recovery demands of different domains.

8.2.4.6 Communications type

Protocol	Description of parameter
LWY	Two-way communication (thus no directional targeting).
IBM	Two-way communication (thus no directional targeting).
VKT	Two-way communication (thus no directional targeting).

8.2.4.7 Implementation

Protocol	Description of parameter
LWY	No restrictions.
IBM	No restrictions.
VKT	ElGamal public key encryption used to compute key recovery information.

8.2.5 Evaluation of protocols against escrow requirements

8.2.5.1 LWY

1. **User completeness:** Yes
2. **Agent completeness:** Yes
3. **User soundness:** The general problems that apply to all key escrow protocols exist (see section 0).
4. **Agent soundness:** No. Although this is a difficult requirement to meet.
5. **User acceptability:** No. Users do not have good flexibility over key management (changing their secret and public keys is difficult). No built in mechanism for split escrow. However, communication between TTPs not required.
6. **Agent acceptability:** Needs clarification of user soundness. Many types of interception are possible, including efficient means of performing node-based interceptions. TTPs can also generate information required for interception in advance, therefore interceptions do not necessarily require computations to be performed by the TTP.

8.2.5.2 IBM

1. **User completeness:** Yes
2. **Agent completeness:** Yes
3. **User soundness:** The general problems that apply to all key escrow protocols exist (see section 0).
4. **Agent soundness:** No. Although this is a difficult requirement to meet.
5. **User acceptability:** Yes. Users have good flexibility over key management (they can change their session key), and cryptographic algorithms used. Split escrow can be catered for. Flexible with regard to key recovery demands of different domains. However, the initialisation stage is relatively complex.
6. **Agent acceptability:** Needs clarification of user soundness. Node based interceptions are inefficient. Agents may need to co-operate with many KRSPs. Computation required by KRSP in order to recover session key.

8.2.5.3 VKT

1. **User completeness:** Yes
2. **Agent completeness:** Yes
3. **User soundness:** The general problems that apply to all key escrow protocols exist (see section 0). However, a third party can detect that key recovery information has been correctly computed without actually performing key recovery. This may discourage abuse of the scheme.
4. **Agent soundness:** No. Although this is a difficult requirement to meet.
5. **User acceptability:** Yes. User can change session key at any time. Split escrow can be catered for. Flexible with regard to key recovery demands of different domains.
6. **Agent acceptability:** Needs clarification of user soundness. Node-based interceptions are inefficient. Computation required by TTP in order to recover session key. However, some abuse of the scheme can be detected without having to perform interception.

8.3 Appendix C: Protocol implemented in the first demonstrator

The following protocol, which is based on the protocol described in [JMW96a,JMW96b], is implemented in the first demonstrator. The protocol is in the context of a pair of users, where one user wishes to send the other a confidential message and needs to be provided with a session key to protect it. The start situation for the first demonstrator is as follows.

1. Four participants, namely two users (one as a sender, say A, and the other as a receiver, say B) and two TTPs (one for each user, say TA and TB), are involved.

2. Two TTPs have agreed between them values g and p. These values must have the usual properties required for operation of the Diffie-Hellman key exchange mechanism, namely that g must be a primitive element modulo p, where p is a large prime and p-1 can be divided by another large prime q. These values have been passed to the two users.
3. Each TTP has an asymmetric signature verification key pair. The private signature key is known only to himself, and an authenticated copy of the public verification key can be accessed by his own user and the other TTP.
4. Each user and his TTP have access to a protected channel between them, which provides origin authentication, data integrity and confidentiality.

Figure 8.3.11 shows the message exchanges of the protocol among the four participants. The protocol includes four parts, each of which can optionally be run separately.

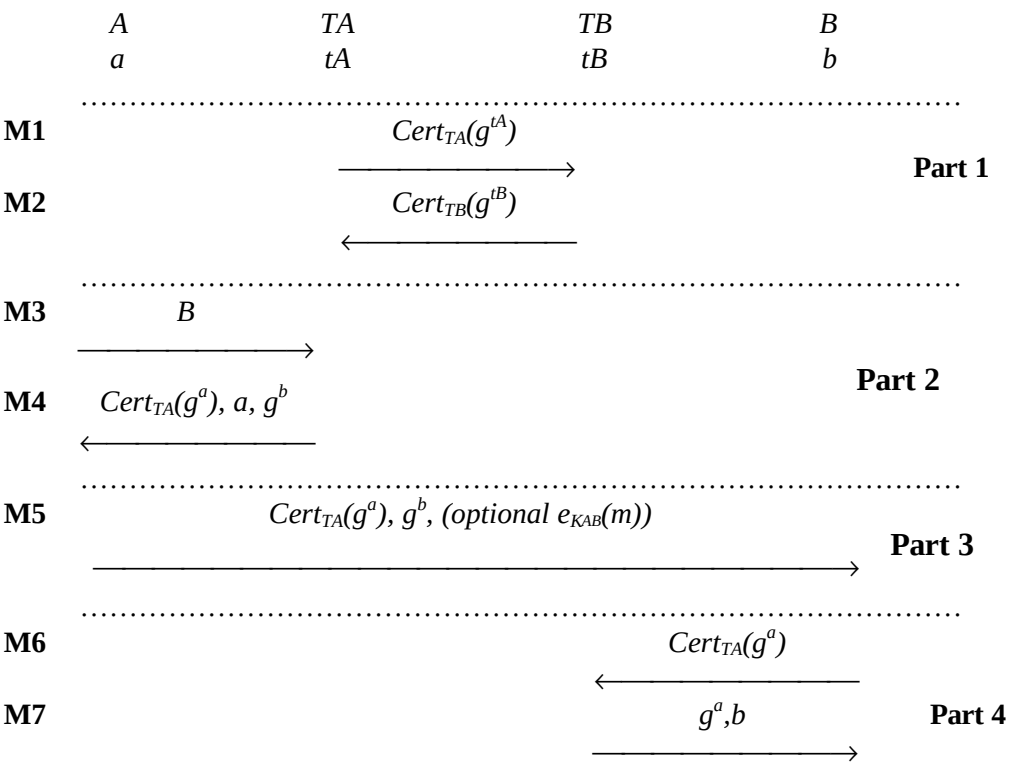


Figure 8.3.11 : The first demonstrator protocol

The following is the description of the protocol.

Part 1: share a secret between two TTPs.

1. Each TTP, *TA* or *TB*, generates a private and public key agreement key pair (tA, g^{tA}) or (tB, g^{tB}).
2. Each TTP sends his public key agreement key signed using his private signature key to the other TTP in M1: $Cert_{TA}(g^{tA})$ or in M2: $Cert_{TB}(g^{tB})$.
3. Each TTP verifies the received public key agreement key using an authenticated public signature verification key of one another.
4. Each TTP computes a shared secret, $K_{TATB} = g^{tAtB}$, using his own private key agreement key and the other's public key agreement key in Diffie-Hellman key establishment algorithm.

Part 2: Certificate generation in A's domain

1. *A* sends *TA* a request in M3 including *B*'s name.
2. *TA* generates a random number a as *A*'s private send key and a corresponding public send key, g^a.
3. *TA* makes a certificate for *A*'s public send key, $Cert_{TA}(g^a)$.

4. *TA* computes *B*'s private receive key, $b = h(K_{TATB}, B)$, and the corresponding public receive key, g^b.
5. *TA* sends *A*'s public send key certificate, *A*'s private send key and *B*'s public receive key to *A* in M4.
6. *A* computes a shared key, $K_{AB} = g^{ba}$, using his own private send key and *B*'s public receive key.

Part 3: message transmission from A to B

A sends the following information to *B* in M5:

- his public send key certificate issued by *TA*,
- *B*'s public receive key, and
- a optional message encrypted by the shared key, K_{AB}.

Part 4: Certificate generation in B's domain

1. After receiving M5, *B* sends *TB* a request including *A*'s public send key certificate issued by *TA*, and *B*'s public receive key sent by *A*.
2. *TB* computes *B*'s private receive key, $b = h(K_{TATB}, B)$, and verifies g^b.
3. *TB* verifies *A*'s public send key certificate using the public signature verification key of *TA*.
4. *TB* sends *A*'s public send key and *B*'s private receive key to *B*. Note that if *B* has already got his current private receive key, this key does not need to be sent in M7.
5. *B* computes the shared key, $K_{AB} = g^{ab}$, using his own private receive key and *A*'s public send key.

After running the protocol successfully, a shared key K_{AB} will be established between the sender and receiver, and this key will be escrowed by both TTPs.