On solutions to the key escrow problem

Mark P Hoyle* and Chris J. Mitchell**

Information Security Group
Royal Holloway, University of London
Egham
Surrey TW20 0EX

Abstract. The first part of this paper is devoted to explaining what key escrow is and why it exists, and attempts to put it into a historical context. The subsequent focus is primarily on key escrow schemes which will work in an international environment. The possibility of using conventional key distribution techniques to provide key escrow services in an international context is first considered, and the associated problems are explored. The 'Royal Holloway' (RH) key escrow scheme is then described in a way which is intended to clarify and motivate its design, and the properties of this scheme and some related schemes are considered.

1 Introduction

In the last two or three years there has been an explosion of interest in the 'key escrow' problem. Most recently this has given rise to an announcement from the US Government, and a simultaneous announcement from a group of major manufacturers, that export restrictions on encryption technology will be lifted for products incorporating key recovery facilities.

The purpose of this paper is to explain what key escrow is and why providing it is non-trivial (at least in an international context), and to introduce a family of technical solutions. One motive for producing this paper is to give a simple explanation of key escrow, and in so doing counter some of the widespread misconceptions about the nature of key escrow. To some extent these misconceptions arise from some of the wilder statements made by those parties opposing the principle of key escrow, often on extreme libertarian grounds.

Of course, what kinds of key escrow solutions should be implemented, if any, is a practical and political question, and one which is beyond the scope of this paper. The purpose of this paper is certainly not to argue for or against the principle of key escrow, but to attempt to provide a simple explanation of some of the technical issues. Only against such a background can a measured debate about the rights and wrongs of the use of the technology be played out.

* Work supported by EPSRC grant number 94007824.

** Chris Mitchell is currently at: Europay International, Chaussée de Tervuren 198A, B-1410 Waterloo, Belgium, whilst on leave from the University of London.

2 Key Escrow

2.1 Background

It is an undisputed fact that the rapidly growing use of public telecommunications and computer networks for sensitive and commercial applications argues strongly in favour of the widespread public use of cryptographic techniques. Of course, serious arguments do exist about how far we need to go in securing these networks, but these arguments tend to be about the level of security required, rather than about the need for any security at all.

The most fundamental services which can be provided by cryptographic methods are confidentiality and integrity protection for transferred and stored data. We apply the term integrity protection to mean not only protection against accidental or deliberate change, but also the provision of means to verify the claimed origin of data. Integrity and confidentiality can be provided independently of one another, and indeed are typically provided using different mechanisms.

Typically, integrity services will be provided by the use of a digital signature or a Message Authentication Code (MAC), and confidentiality will be provided by the use of encryption. For many commercial organisations, the primary security requirement is integrity, and confidentiality for transmitted data is at most of secondary importance. However, in recent years there has been an enormous growth in the use of public networks, in particular the Internet, for all kinds of applications, including many for which confidentiality is important. For example, encryption is probably essential for the security of electronic commerce over public networks; without confidentiality protection for users' bank details, a variety of frauds may become possible.

However, widespread encryption of user communications on an end-to-end basis, i.e. encryption at source and decryption at destination with no decryption between, presents a problem to police forces and other law enforcement agencies throughout the world. Currently, in many (probably nearly all) countries, certain official agencies may intercept telecommunications traffic, if so authorised by an appropriate legal process. Typically a police force wishing to intercept the communications of a suspected criminal can do so if granted a warrant from a judicial authority.

Note that, at least within the UK, such legal interception powers are non-trivial to obtain. Typically, interception warrants will only be granted in cases of serious crime, e.g. crimes for which the first offence penalty would be a prison sentence of a year or more. Once a warrant has been granted, the collection of the intercepted data would normally be performed by the network provider (e.g. a public network operator) acting on behalf of the agency wishing to read the traffic. The involvement of another third party also helps make abuse of interception powers more difficult.

Obviously, the widespread use of encryption will nullify this interception capability. Observe that integrity protection through MACs and digital signatures is not a problem in this respect, since it does not threaten the interception capability valued by law enforcement agencies world-wide. We are thus presented with the problem of reconciling the legitimate

requirement of users and business for confidentiality of communications, with the requirements for legal interception.

2.2 Key escrow — a solution?

One solution to this problem, which has recently received very wide attention, is to use a *key escrow* scheme. The idea of such a scheme is that copies of the keys used by parties to encrypt their messages are lodged with *escrow agents*, and that, when an agency is given authorisation to intercept a particular user's communications, they can apply to the appropriate escrow agent for a copy of that user's key. An escrow key mechanism can then simply be regarded as a method of key generation and distribution such that users are equipped with encryption and decryption keys when they need them, and where all relevant escrow agents also have access to a copy of decryption keys (in the event that they are required by legally authorised parties).
We therefore arrive at the following 'model' of a key escrow system, with three types of entity involved:

1. *Users* are entities who wish to exchange confidential messages,
2. *Trusted Third Parties* (*TTPs*), which can be further split up into two, not necessarily disjoint, groups:
   (a) *TTPs* offering trusted services (e.g. time-stamping and certification of public keys),
   (b) and *KEAs* (generally known as *Key Escrow Agencies* or *Key Recovery Agencies*) that provide a key management service to users and also provide a key escrow/recovery service on demand,
3. and an *Interception Agency* which will make use of an escrow agency to obtain keys when authorised to do so.

This is, of course, a simplified model. Corporate entities may also wish to have the means to intercept encrypted traffic going to or from their employees, at least while it involves use of their equipment or communications facilities. In such an event companies, or parts of companies, may fit any or all of the above three roles (since companies may provide the key management for their own private networks).
The notion of Trusted Third Parties acting on behalf of users, but within a regulatory regime which requires them to divulge user secrets when legally required to do so, is a familiar one. Most banks fill precisely this role; they look after money for users, and will provide information regarding financial transactions to tax authorities when legally required to do so.
In recent months a number of governments have suggested that they will encourage the development of encryption products incorporating a key escrow facility; in particular, the US Government has stated that existing export controls for encryption products will be relaxed if escrow is included. The current approach appears to be to encourage voluntary adoption of such schemes, without forcing all users of cryptography to adopt them. Thus the existing user of home-built encryption software can carry on using it, even though it may not incorporate an escrow facility. However, if international use is required, then the legal controls on the

export of encryption products will make use of such products much more difficult. This primarily voluntary approach would appear to reduce the threat perceived in some quarters regarding the loss of existing freedoms to use cryptographic techniques.

One question that immediately arises in the context of *voluntary* (rather than compulsory) use of escrow technology, is 'Why bother if all the criminals will not use escrowed technology'? There are a number of possible answers to this question, including, possibly, to admit that there is no point in bothering! However, to see why this might not be the case we need to examine in a little more detail how escrowed encryption might be offered and used.

The main likely area of application for key escrow services is in public data communications networks, such as the Internet and data transmission services using mobile telephony. We might expect to see end-to-end encryption, with built in key management (and key escrow) support, being offered by either the provider of the network, or by independent network service providers. Commercial organisations, who may already buy in certain network services, and who also wish to have a confidential communications facility, would then subscribe to one of these key management services for confidentiality support. These organisations would then enjoy the benefit of secure encryption technology being available on an international basis, whilst governments would be happy to allow free export of the necessary hardware and software, safe in the knowledge that interception using a key escrow service will be possible for their law enforcement agencies.

Indeed, if all public networks offered a practically unbreakable encryption service without any provision for key escrow, criminals would find such a service extremely useful, and it would remove a valuable weapon from the armoury of those agencies which we collectively appoint, authorise and pay to protect us against the actions of criminals. Faced with the possibility of freely available encryption to all, governments are much more likely to prohibit encryption altogether, and/or widen the use of other, possibly more intrusive, methods of monitoring traffic. Thus the real alternative is probably not whether we use some kind of key escrow or not, instead it is whether we have encryption with key escrow or no public encryption service at all!

We can now attempt to answer the question as to why criminals would use an escrowed network for their confidential messages, since they would surely be aware that law enforcement agencies could apply for a warrant to decrypt their encrypted messages. There are a number of reasons why we might expect this to happen.

- Firstly there is the issue of *convenience*. It is to be expected that, at some time in the not too distant future, all public communications networks will offer an encryption option with automatic key management support. Criminals are likely to use such an option, even if they know they may be intercepted, since they 'might as well' (it will cost them relatively little and will certainly not make life any easier for the police!).
- Secondly we have a point regarding *history*. Criminals have continued to use the public telephone and postal networks in full knowledge

of the possibility of interception. It seems rather unlikely that they would avoid use of public networks with an escrowed encryption facility. Moreover, they may wish to communicate with non-criminals, and in such a case they would be obliged to use standard network facilities. Of course the sophisticated criminals will use their own 'unescrowed' encryption, but this should not, in itself, prevent society from limiting the public availability of completely untouchable encryption. Most criminals are not sophisticated!

Having considered two reasons why key escrow might continue to be a valuable resource to society, we should also consider why anyone would worry about the idea of key escrow, given that, in most countries, everyone is already subject to telephonic and postal interception. The usual answer of opponents is to suggest that this is one more step to a 'big brother' society. Whilst the prospect of constant electronic monitoring of all aspects of our lives is certainly not a welcome one, the introduction of key escrow does not necessarily do anything to bring this closer. Indeed, its introduction will not enable the state to do anything it could not do before, since the automatic collection of intercepted data is already an established technology, and all escrow will do is enable the continued use of such data for law enforcement purposes; indeed, it is possible to argue that it is non-escrowed encryption which threatens the status quo.

Traditionally most European societies have managed to cope with the necessary compromises between individual privacy and the need to give the state certain limited powers to protect its population against serious crime. Ultimately, the decision about where lines should be drawn is a political one, and outside the scope of this paper.

It is interesting to note that arguments about the rights and wrongs of key escrow often seem to mirror arguments about gun control. That is the arguments revolve around the conflict between individual rights to use a technology, and the protection of society, particularly its weaker members, against misuse of technology by criminal elements. One argument against gun control, which precisely mirrors one of the arguments against key escrow, is that criminals will bypass such legislation. Of course they will, but there are still benefits from preventing the open sale of weapons, particularly in reducing armed crime. Of course, limiting the availability of weapons is a major loss of rights for an individual, and each society makes its own decision about the weight of the conflicting arguments.

Finally note that there are other uses for key escrow technology, e.g. for companies to recover their data which may have been encrypted by an ex-employee who took the keys with them, which even some of the most vocal opponents of key escrow recognise as being potentially valid. This application is often referred to as *key recovery*, although the difference between key recovery and key escrow is not always clear; indeed it would appear that in some circles the term key recovery is used instead of key escrow. Certainly the idea that work on this technology is inherently a bad idea, in the same way that one might argue that work on nuclear weapons is immoral, now seems to be rather out of fashion.

2.3 The international key escrow problem

As we see below, while it would appear to be relatively straightforward to design escrow systems which operate in a single domain (e.g. a large country), with a single legal framework, international use presents a number of problems. Some of the most significant problems are as follows.

- There will typically be more then one intercepting body, normally at least one per domain.
- A legal warrant issued in one domain will typically have no validity in another domain. Hence at least one escrow agent will need to exist in every domain, so that every intercepting agency has an escrow agent to whom they can pass a warrant.
- Different domains will typically have different legal rules about to whom and in what circumstances a warrant can be issued for legal access to encrypted communications.
- There may be a lack of trust between domains, making arrangements applying to cross-border encrypted communications difficult to define and operate.

In Section 5 we consider how solutions to the international problem may be devised.

2.4 Some historical background

For many years most western governments have sought to control cryptographic technology. There are two major reasons for these controls, which have mainly applied to ciphers rather than other types of cryptographic technique. Note that this concentration on ciphers is both for historical reasons (cryptography was the same as the study of ciphers until relatively recently), and because of the fact that confidentiality is the most politically sensitive of the services which cryptographic techniques can provide.

- The first reason relates to national security; in particular governments typically wish to intercept other countries' communications. By limiting the export of equipment incorporating cryptographic facilities to 'friendly' countries, the objective is to ensure that unfriendly countries use inferior ciphers, or perhaps none at all. This can then lead to a decisive military advantage. The importance of cryptography and cryptanalysis in war is well documented, particularly in the case of the second world war. Indeed, the needs of cryptanalysts during the second world war played a major role in the development of modern electronic computers, as testified by the development of the ground-breaking Colossus machine at Bletchley Park. There has also been a general desire to discourage public research in cryptography; although much research in cryptography has been conducted by academics and others since the 1970s, previously very little appeared in the public domain. Most developed countries maintain significant government-controlled interception, cryptographic and cryptanalytic capabilities, e.g. at GCHQ in the UK and at NSA in the US, whose research is most certainly not in

the public domain, although publication by members of these organisations is not unknown.

In recent years there have been many efforts by those inside and outside the cryptographic community to attempt to discredit the export controls that exist. It is certainly true that these controls have been incapable of stemming the flow of encryption software across the Internet, although laws are probably being broken every time these pieces of software cross international boundaries. However, computer experts who use such software are typically not the target of these controls, and the export of encryption hardware, which is often of key importance for military and paramilitary use, remains strictly controlled.

- The second reason for control stems from the need of government agencies to intercept internal communications to combat crime. For example, police routinely 'tap' the telephones of suspected serious criminals, when provided with appropriate warrants. Although it would seem unlikely that a criminal would use a public communications network to discuss criminal activity, particularly when the ability of the police to tap telephone traffic is well known, apparently they do, and the ability to tap telephone traffic is seemingly highly valued by law enforcement agencies.

In order to limit the flow of cryptographic technology around the world, all implementations of cryptographic algorithms (hardware and software) have been subject to COCOM-based export regulations (these regulations have recently been replaced by the Wassenaar arrangement, although the effect of these controls appears to remain much the same). Since World War II these regulations have been very effective in limiting access by certain states to sophisticated cryptographic equipment. Whilst documentary evidence is very difficult to obtain, for obvious reasons, there is strong anecdotal evidence that this policy has been enormously helpful to western countries in a variety of conflicts that have arisen since 1945.

In parallel with this control of export of cryptographic technology, some countries, e.g. France, have regulated the internal use of cryptography. The main objective of these controls has been to limit the use of ciphers, although the French law covers all cryptographic techniques. This has met the needs of law enforcement agencies who wish to retain the right to intercept traffic. Other countries, e.g. the UK and US, have not regulated the internal use of cryptography, possibly because cryptographic technology has not been widely available for use, and hence controls have been unnecessary. However, with the growth in personal computers and other cheap consumer electronics capable of performing sophisticated cryptographic calculations, one might anticipate pressure in some countries for the introduction of new legislative controls over the use of cryptography, and in particular of ciphers.

In opposition to this is the growing legitimate need for end-to-end privacy over public networks. If the Internet and other public networks, such as mobile telecommunications networks, are to be used for commerce, then in many cases privacy for confidential user information is required. This

in turn creates a growing requirement for public use of encipherment technology. This tension leads naturally to the topic of key escrow.

2.5 The development of key escrow

As far as the public domain is concerned, the history of key escrow started as recently as 1993, with the proposal by the U.S. government of the *Escrowed Encryption Standard* [19], EES, also known as the *CLIPPER* scheme. This scheme, which we now briefly outline, only attempted to solve the problem for the US, and did not address the needs of users wishing to have communications confidentiality for traffic entering or leaving the US.

At the heart of the CLIPPER scheme was an encryption algorithm (the SKIPJACK algorithm) devised by the NSA, and which was intended to remain secret, although certain interface details were made public. Details would obviously need to be released to selected integrated circuit manufacturers, but these manufacturers would be required to commit themselves to maintaining the secrecy of the algorithm. A range of products would then be developed incorporating the CLIPPER algorithm, for use by anyone in the US needing confidentiality for communicated data.

To enable key escrow, all implementations of the algorithm were to incorporate 'key checking' for all entered keys, i.e. checking that the key contains redundancy according to a particular (secret) cryptographic formula. How this was implemented is not clear, but it could have been arranged by adding a cryptographic check value to each key, computed using a secret key known only to the devisers of the scheme.

Whilst such a scheme is a reasonable candidate for use within a single domain or country, it is less suitable for international use since it depends completely on the secrecy of a single algorithm. Once the algorithm is known to more than one entity, control of keys is lost, and the scheme becomes unworkable. On the other hand, leaving the control of all keys within a single country could never be widely acceptable.

In the remainder of this paper we therefore focus on other types of solution to the key escrow problem.

2.6 Key escrow and the public key infrastructure

Before proceeding it is worth spending a few moments distinguishing between the Public Key Infrastructure (PKI) and TTPs providing Key Escrow services. Since they are both TTP-based and provide security services, it is easy to confuse the two; however the goals are typically very different.

The purpose of the PKI is to provide a means of distributing public signature verification keys on a wide, possibly global, scale. It is based on the idea of a network of Certification Authorities (CAs), signing public key certificates for individual users. These certificates consist of a copy of the public verification key for that user, concatenated with the user's

name, an expiry date, and certain other information, all signed using the CA's private signature key.

Anyone wishing to obtain the public verification key for another user, first obtains the public key certificate for that user, and then verifies the certificate using the public key of the appropriate CA. This yields a verified copy of the user's public verification key, which can be used to check digital signatures on messages originating from that user. The certificates are typically, although not necessarily, distributed by means of *directories*, which need not be trustworthy.

Thus the Public Key Infrastructure's primary role is to support the widespread use of digital signatures. As such, it has been supported by governments and business world-wide, since there is much to gain and very little to lose from implementing global, secure, digital signatures. It will make certain types of fraud much more difficult, yet it will do nothing to interfere with the ability of government agencies to intercept communications. In fact, this notion fits well with the US government backed signature algorithm DSA, which has the great advantage that, unlike RSA, it cannot be used for encryption.

Of course, in principle there is nothing to stop CAs signing certificates containing public encryption keys instead of public verification keys, although this is not the typical case. Such an idea is also likely to meet with government resistance because typically it will not allow key escrow. Key escrow, unlike the PKI, supports the widespread use of encryption, and simultaneously allows warranted interception to take place. There are many ways of providing key escrow, as we describe in the remainder of this paper, but typically it involves a TTP handing over a user's encryption key to an interception agency when warranted to do so.

It is important to stress that key escrow relates to encryption keys and not signature keys. Some opponents of key escrow have, accidentally or deliberately, muddied the waters somewhat by suggesting that governments wish to escrow signature keys. Not only is this not justified by the evidence, but the fact is that the US and other governments have actively promoted the development of the PKI, which is quite orthogonal to the notion of key escrow. The development of the DSA standard can be seen as a very deliberate effort by the US to promote a secure signature technique which avoids export restrictions by being a signature-only algorithm, and can therefore be used world-wide; this is hardly the act of a body wishing to restrict the use of signatures.

In fact governments and their law enforcement agencies have every reason not to escrow signature keys. The worst possible scenario would be for criminals to repudiate signatures on the basis that the government has access to their private signature key, and therefore a government agency must have forged their signature!

3 What do we want from a Key Escrow System?

There are two main objectives/requirements for a typical key escrow system.

- *Warranted interception*, i.e. the ability of an interception authority (typically a government agency) to obtain the means to decipher intercepted enciphered messages and/or stored enciphered data, typically for law enforcement and/or protection of national security. This is the main objective behind the CLIPPER proposal. Escrow schemes can be used elsewhere by having other parties play the role of the 'authority', e.g. companies, organisations, private persons.
- *Data recovery*, i.e. the ability to recover plaintexts from ciphertexts in case of lost, damaged, or sabotaged keys.

3.1 Requirements for key escrow

There have been many papers and groups suggesting various requirements/constraints for key escrow systems. It is impossible to formulate a single universally acceptable list, because the exact practical requirements will always be dependent upon the application of the system. In this paper we will confine ourself to giving just one list. This is a draft list of requirements produced by the UK Department of Trade and Industry (DTI), which, although not official UK policy, indicates the direction of DTI thinking on the role of TTPs in supporting key escrow services.

1 The framework should provide benefits to the legitimate user.
2 It should provide for both national and international working.
3 It should be public and unclassified.
4 It should use well known techniques.
5 It should support all forms of electronic communication.
6 It should be compatible with the different laws and regulations of participating countries concerning interception, use, sale and export.
7 It should not impede the due process of law and order. In particular, it should allow near-real-time access when a warrant is held.
8 It should provide access under warrant (or other legally-constituted form of authority) to both incoming and outgoing communications.
9 It should enable the sender to limit the length of time for which any key is used.
10 It should provide for the use of a variety of cryptographic algorithms whether in hardware or in software.
11 It should not enable those with a warrant to fabricate false evidence.
12 It should ensure that attempted abuse by the sender can be noticed by the receiver.
13 It should not require a user to deal with a Trusted Third Party in another country.
14 It should not require either regular or on-line communication between Trusted Third Parties.

This list is clearly intended to apply to international solutions (see requirements 2 and 13). The technical solutions we describe later in this paper are intended to fit, as much as possible, to this list, which we refer to as the 'DTI requirements' (although this is not meant to imply that they are official DTI policy).
Before proceeding note that, since we are concerned only with escrowing encryption keys, requirement 11 is not an issue here. Indeed, it is further

evidence to contradict any suggestion that there is a requirement to escrow signature keys.

3.2 Types of warrant

In the rapidly growing literature relating to key escrow, there has been a considerable amount of discussion regarding what the reasonable scope of an interception warrant might be; note in particular [17, 18]. Such an interception warrant would typically be issued by a judicial authority to an interception agency, and will describe what access to users' communications should be provided to this Interception Agency by a Key Escrow Agency. Of course, the exact nature of the key escrow system will determine what types of access can be readily provided, and hence understanding what types of warrant will be issued gives a very important measure of the usefulness of a key escrow scheme.

The most typical warrant would appear to be one which relates to a single communicating entity. Some authors suggest that it may be useful to be able to support separate warrants for all outgoing communications from this entity, and for all incoming communications to the entity. Others suggest that 'time-bounding' warrants is a very important requirement, i.e. so that the interception agency is only given access to an entity's communications for a specified period of time (e.g. if they are given a key to decrypt enciphered communications, then it should stop working when the warrant expires, and it should also not work with messages sent prior to the start date in the warrant).

The DTI requirements list in Section 3.1 only explicitly specifies that warranted access to all incoming and outgoing communications should be possible (see requirement 8), without separating the two. It does also refer to the need for a sender to be able to limit the time period for which any particular key is used, which implies a type of time-bounding (see requirement 9). We therefore restrict our attention mainly to the case where warrants cover communications both incoming to and outgoing from an entity, and where time-bounding may be a requirement. The requirement for separate send and receive warrants seems unlikely to be of much practical significance.

Note that it is an implicit assumption of most key escrow schemes that interception warrants only normally apply to communications either originating or terminating in the country (or domain) where the warrant is issued. Although other possibilities have been considered in the literature, see, for example, [5, 6], we restrict our attention to this case. Even in this case, there are two possibilities for the entity covered by a warrant:

- the entity concerned is sending or receiving communications from within the domain of the interception agency, in which case all that entity's communications are covered by the warrant, or
- the entity concerned is sending or receiving communications from a different domain to that of the interception agency, in which case only those communications which originate or terminate within the interception agency's domain are covered by the warrant.

Of course, these two possibilities are not mutually exclusive, given that users may migrate from one domain to another during the lifetime of

a warrant; however, to simplify the discussion, we treat the two cases separately here. Finally observe that the first case can be considered as the 'most typical'.

3.3 Types of solution

Almost all of the proposed solutions to the key escrow problem adhere to a model similar to that described in Section 2.2, i.e. where TTPs act on behalf of both users and interception agencies. Again it is generally the case that these TTPs provide key management and/or key generation services for users, whilst at the same time providing keys to interception agencies (to enable decryption of intercepted messages). There are then two main ways in which such a key management scheme can work:

- the encryption algorithm is fixed and secret (typically implemented in some kind of secure hardware), and only works with keys of a certain secret form (e.g. the Clipper scheme), or
- the encryption algorithm is not fixed, and the key management system simply arranges for the distribution of keys which may be used in a variety of algorithms.

The advantage of the first type of scheme is that it prevents abuse, in that users have no alternative but to use officially issued keys, i.e. keys which are known by the Escrow Agency. The main disadvantage is that it prevents use by a multiplicity of TTPs in different domains (since each TTP needs to know how to generate keys), and hence is not suitable for international use.

The main problem with the second type of scheme is that it opens up the possibility of abuse, i.e. of users making use of part or all of the key management scheme in order to establish a shared secret key for encipherment, but to 'distort' the scheme in such a way that the escrow agent does not have the correct key to decipher the encrypted communications. This is a possible problem we discuss further in Section 5.3 below.

Given that we are interested in solutions which can work in an international environment, we do not consider further escrow schemes of the first type (i.e. which are based around a specific secret algorithm), and we restrict our attention to solutions of the second type from this point on.

4 Possible solutions using existing key distribution methods

In this section we examine solutions to the key escrow/recovery problem based on existing techniques for key management. We look at solutions in the light of international use and identify some problems involved with the use of these methods.

4.1 Symmetric cryptography

Probably the most common solution to the problem of key distribution using symmetric cryptography is for the network(s) to provide a special

type of TTP known as a *Key Distribution Centre* (KDC), i.e. a trusted network resource that shares a key with each subscriber and uses these in a bootstrap process to provide additional keys to the subscribers as needed. When one subscriber wants to communicate securely with another, he/she first contacts the KDC to obtain a session-key for use in that particular conversation. The KDC then generates a session key and provides the means for both users to obtain a copy of this in a way which preserves its confidentiality.

Key distribution protocols vary widely depending on the cost of messages, the availability of multiple simultaneous communications, whether the subscribers have synchronised clocks, and whether the KDC has authority not only to facilitate, but also to allow or prohibit, communications. In the discussions below we sketch two key distribution protocols; these are not complete specifications in that we ignore the provision of entity authentication, an important aspect of key distribution. However, depending on what methods are used for timeliness, e.g. time-stamps or unpredictable nonces (i.e. 'challenges' used only once), data items can be added to the messages specified (and, if necessary, preliminary messages added), to make the protocol provide mutual entity authentication. For details of how this can be achieved see, for example, ISO/IEC 9798–2 and 11770–2, [12, 13]).

The protocols described are intended to be 'typical' examples of key distribution protocols based on symmetric cryptographic techniques. However, as we explain in the accompanying discussions, minor adaptations have been made in order that they can also support key escrow.

The idea of modifying conventional symmetric cryptography based key distribution protocols to support key escrow services is certainly not a new one. In Denning's on-line catalogue of key escrow systems, [7], which is a companion document to Denning and Branstad's taxonomy, [8], a number of mechanisms of this type are listed, including 'Diffie Time-Bounded Clipper' (1994), 'Leiberich Time-Bounded Clipper' (1994), and 'PC Security Stoplock' (1995).

Single domain with single TTP The following example applies to the case where a single domain with a single TTP is involved. We suppose A wishes to send B a confidential message, where both A and B share a secret key (denoted K_A and K_B respectively) with a KDC.

1. A calls the KDC and requests a key for communicating with B.

$$A \longrightarrow KDC: \quad A, B$$

2. The KDC responds by sending A a pair of tokens. Each token contains a copy of the required session key K_S, one encrypted so that only A can read it, and the other so that only B can read it.

$$KDC \longrightarrow A: \quad \{K_S, A, B\}_{K_A}, \{K_S, A, B\}_{K_B}$$

where $\{X\}_K$ denotes the symmetric encryption of data X using key K. A can immediately decrypt the first token and obtain the session key K_S, to be used between A and B.

3. When A wishes to send a secret message M to B, A encrypts it under the session key, and sends it with both tokens to B.

$$A \longrightarrow B: \quad \{M\}_{K_S}, \{K_S, A, B\}_{K_A}, \{K_S, A, B\}_{K_B}.$$

   B uses the second token to recover K_S, the session key needed to decrypt M.
4. Both tokens should also be sent with all subsequent messages (from A to B) which are encrypted using the same session key K_S.

First note that the above protocol differs from a 'typical' mechanism of this type (see, for example, [13]) in two respects. First it would normally only be necessary to send the second of the two tokens with the encrypted message in step 3. Second, in subsequent uses of the same session key, it would normally not be necessary to send any tokens. These two (minor) differences are present to allow for key escrow. Of course, even without the transfer of both tokens the interception agency could decrypt the messages with the assistance of the appropriate KDC (as long as the KDC retains a copy of the session key), although we wish to consider schemes here that involve the minimum interaction between Interception Agencies and KEAs (as in DTI requirement 7).

Now suppose that the KDC is also licensed to act as a Key Escrow Agency. To be able to decrypt the communications between A and B, the Interception Agency will first need to obtain a warrant giving it permission to read either all A's communications or all B's communications. Once the Interception Agency has obtained this warrant, it presents it to the KDC, who hands over either K_A or K_B, depending on who the warrant covers. Armed with this key, and given any intercepted message, the appropriate token accompanying the message can be used to obtain the session key and decrypt the message.

The scheme, as described, does not permit time-bounding of warrants. However, this is not difficult to add to the scheme. One way of achieving this is as follows. Instead of using K_A to encrypt the token for A, the KDC uses a 'daily' key S_A, which is computed as a (public function) of the date and the key K_A, e.g. using a one-way hash-function; session keys are also changed at least once per day. Given a time-bounded warrant valid for a set period of days, the appropriate set of 'daily' keys could then be computed in advance and passed to an Interception Agency, but these keys would not be of any use in decrypting messages sent at other times.

Multiple domains The previous scheme is unsuitable for use in more than one domain, since it uses a single TTP, but can be extended for use with multiple TTPs in multiple domains, and might even be suitable for international use. It operates as follows, where we suppose that A and B have separate TTPs, T_A and T_B, with whom they share secret keys, K_A and K_B, respectively. We also suppose that the two TTPs share a secret key $K_{T_A T_B}$.

As previously, we suppose A wishes to send B a confidential message.

1. A sends T_A a request for a key for communicating with B (and an indication of who B's TTP is):

$$A \longrightarrow T_A : \quad A, B, T_B$$

2. T_A responds by sending A a pair of tokens. Each contains a copy of the required session key, one encrypted so that only A can read it and the other so that only B can read it. The encryptions are performed using newly generated keys K_A^* and K_B^*. T_A also sends K_A^* to A, encrypted under K_A.

$$T_A \longrightarrow A : \quad \{K_A^*\}_{K_A}, \{K_S, A, B\}_{K_A^*}, \{K_S, A, B\}_{K_B^*}$$

   where $K_A^* = f(K_{T_A T_B}, \mathrm{ID}_A)$, $K_B^* = f(K_{T_A T_B}, \mathrm{ID}_B)$, ID_A and ID_B are values (identifiers) uniquely identifying A and B, and f is a one-way function agreed between T_A and T_B (f could be public and globally used). A first obtains the key K_A^*, and then decrypts the first token to obtain the session key K_S.
   Note that, instead of encrypting the tokens with K_A^* and K_B^*, a 'typical' protocol of this type would simply use K_A to encrypt the first token and $K_{T_A T_B}$ to encrypt the second token; we make these minor changes both to permit key escrow and to reduce the amount of communication between B and T_B.
3. When A sends a secret message M to B, A encrypts it using the session key, and sends it with both tokens to B:

$$A \longrightarrow B : \quad \{M\}_{K_S}, \{K_S, A, B\}_{K_A^*}, \{K_S, A, B\}_{K_B^*}.$$

4. When B receives the encrypted message and tokens, there are two possible cases to consider (we assume that B knows which TTP A is using). If B already knows K_B^* (which is a function only of the identity of T_A), then the protocol is complete. If B does not know K_B^*, then B sends a message to T_B requesting a copy:

$$B \longrightarrow T_B : \quad T_A.$$

5. T_A now computes K_B^*, using the function f and the shared secret key $K_{T_A T_B}$, and sends it back to B:

$$T_B \longrightarrow B : \quad \{K_B^*\}_{K_B}.$$

   B can now recover K_S and use it to decrypt the message from A.
6. Both tokens should also be sent with all subsequent messages (from A to B) which are encrypted using the same session key K_S.

First note that the above protocol differs from the 'typical' mechanism of this type in several minor respects. First, as we have already observed, the keys that would normally be used to encrypt the first and second tokens are K_A and $K_{T_A T_B}$ respectively. Second, as for the previous mechanism, it would normally only be necessary to send the second of the two tokens with the encrypted message in step 3, and, in subsequent uses of the same session key, it would normally not be necessary to send any tokens. These differences are present to allow for efficient key escrow, i.e., as previously, we wish to avoid the Interception Agency having to enlist

the assistance of the KEA to decrypt every intercepted message. In fact these changes seem to improve the efficiency of the protocol at minimal cost even if key escrow is not required, and might usefully be considered for more general adoption.

Now suppose that the KDCs are licensed to act as Key Escrow Agencies in their respective domains. To be able to decrypt the communications between A and B, the Interception Agency will first need to obtain a warrant giving it permission to read either all A's communications or all B's communications. Once the Interception Agency has obtained this warrant, there are four cases to consider.

- If the Interception Agency is in A's domain and has a warrant to intercept all A's messages, then the warrant is presented to T_A, who hands over K_A^* (this enables reading of all traffic from A to any user making use of the KDC T_B).
- If the Interception Agency is in A's domain and has a warrant to intercept all B's messages, then the warrant is presented to T_A, who hands over K_B^* (this enables reading of all traffic from any client of T_A to user B).
- If the Interception Agency is in B's domain and has a warrant to intercept all A's messages, then the warrant is presented to T_B, who hands over K_A^* (this enables reading of all traffic from A to any client of T_B).
- If the Interception Agency is in B's domain and has a warrant to intercept all B's messages, then the warrant is presented to T_B, who hands over K_B^* (this enables reading of all traffic to B from any user making use of the KDC T_A).

Armed with the appropriate key, and given any intercepted message, the appropriate token accompanying the message can be used to obtain the session key and decrypt the message.

The scheme, as described, does not permit time-bounding of warrants. However, as with the previous scheme, it is not difficult to add time-bounding to the scheme, e.g. by including a date stamp in the calculation of K_A^* and K_B^*.

Evaluating the mechanisms We now briefly consider how well these two schemes meet the DTI criteria, and how efficient they are. If we note that A can ask for a new session key whenever he/she wants (see DTI requirement 9), the protocol deals purely with encryption (see requirement 11), and that no real-time inter-TTP communications are required in either case (see requirement 14), the second protocol appears to have the potential to meet all the DTI criteria, with the possible exception of requirement 12, since the recipient has no means to check that the first of the tokens is sent correctly. The first protocol meets all the criteria except numbers 12 and 13 (this latter requirement fails since the protocol only makes use of a single TTP).

As far as we are concerned here, the issue of 'efficiency' relates to the number of messages that need to be exchanged to support use of the protocol, particularly messages between a user and his/her TTP. In both mechanisms, A needs to exchange messages with his/her TTP to set up

communications with B. In the first mechanism B has all that he/she needs to read the message without further exchanges with the KDC; however, in the second protocol an extra exchange between B and B's TTP may be required to read the message.
The system could become rather unmanageable for very large global networks, because the TTPs are directly involved in setting up all communications, and hence are likely to become a significant bottle-neck. Of course these problems apply just as much to unescrowed secure networks where key management is based on the use of symmetric cryptography.
Also note that every pair of TTPs will need to have access to a shared encryption algorithm. If the scheme were to be used on a global scale, this in itself could present significant implementation difficulties.
Despite these reservations, it should be clear that symmetric cryptography based mechanisms have the potential to meet all but one of the main requirements for an escrow scheme. The main limitations are the usual limitations of symmetric cryptography based key management schemes, i.e. the need for *on-line* access to TTPs by both sender and recipient.
In the remainder of this paper we consider ways in which the use of asymmetric cryptographic techniques can be used both to reduce the number of message exchanges, and, perhaps most importantly, to reduce the need for on-line communications between a user and a *trusted* third party (and hence reduce the 'bottle-neck' problem). Typically, the use of asymmetric cryptography allows the substitution of an untrusted distributor of certificates for the on-line TTP required when symmetric cryptography is used for key distribution. As we shall see, this result extends, at least partially, to the case where the key distribution mechanism also needs to support key escrow.

4.2 Asymmetric cryptography

We start our discussion of public key based schemes by considering the usefulness of conventional asymmetric cryptography based key distribution techniques for supporting key escrow.
We assume the reader is familiar with the notion of asymmetric or public key cryptography, as introduced in 1976 by Diffie and Hellman [9]. Public-key encryption is usually based on difficult number theoretic problems (e.g. factoring, discrete logarithms), which involve computationally complex calculations with large numbers. Thus public key encryption of entire messages is typically too time-consuming, and so it is common to employ a combination of symmetric and asymmetric cryptography. For confidentiality purposes, the message is (symmetrically) enciphered with a randomly generated *session key*, and this 'short' session key is (asymmetrically) enciphered with the public encryption key of the receiver. The asymmetrically enciphered session-key is sent along with the enciphered message. On delivery, the receiver can use his private decryption key to find the session key, and hence decrypt the message.
By setting up a *directory* of certificates of users' public encryption keys it is possible to set up a communication infrastructure allowing any two

users to communicate securely. This would be precisely analogous to the Public Key Infrastructure referred to in Section 2.6, with certificates containing public encryption keys instead of public verification keys.
In order to allow warranted interception we could require all Certification Authorities to obtain and store a copy of every user's private decryption key before they sign the user's public encryption key. This private key could then be handed over to an interception agency in possession of an appropriate warrant. The agency could then use it to obtain all the session keys used to encrypt messages sent to a specified entity.
There are two problems with such an approach. The first is that obtaining a specific private key will involve approaching the appropriate CA, which might reside in a different domain from that where the interception warrant is issued. This breaches DTI requirement 13, i.e. the use of this type of scheme does not permit international operation.
The second, even more fundamental, problem with this approach is that, while it is fine for providing warranted access to all the confidential messages *received* by a specified entity, providing warranted access to all messages *sent* by a specified entity is much more difficult. This is because each message to a different recipient will be protected using a different public key (i.e. the public key of the recipient). This leaves the interception agency with no choice but to approach the escrow agent with a request for the session key for each intercepted message, an approach which will become hopelessly inefficient and will, in most cases, breach DTI requirement 7.

US Public Key Infrastructure (Clipper III) A recent, unofficial draft of the US Interagency Working Group on Cryptographic Policy [20] envisions a form of key escrow incorporated in a government-sponsored, voluntary PKI, usable for both confidentiality and integrity. The PKI itself would be supported by private-sector key-management organisations (certifying authorities). These would also hold in deposit private encryption keys, and will operate within performance standards set by law. Thus, they serve as a Key Escrow Agency (KEA).
This type of scheme is very much of the category we have just discussed, and hence one problem with this scheme, which will affect its usage on an international basis, is that the receiver's private key is needed for the decryption, not the sender's. Thus, if someone suspected of criminal activities sends a message to a 'good user', law enforcement require co-operation of the receiver's KEA to decrypt the message with the good user's private key. This is exactly the type of problem we have just considered. Indeed, this type of solution would appear to fail to meet DTI requirement 7.
To use this scheme internationally would require co-operation between the receiver's KEA and the law-enforcement agency in the sender's country, which, in general, would be impractical. As we have already seen, this would appear to rule out this type of solution in an international context.
Thus what we require are new types of public key solution allowing key escrow in an international context, as well as permitting simple warranted

access to both sent and received messages. We can achieve this through the notion of separate 'send' and 'receive' key pairs, as we describe in the next section.

5 New types of solution

We now describe a series of key distribution mechanisms allowing key escrow, all based on the use of public key cryptography. The schemes have been designed to work in different operational environments. We start with the simplest scheme, which applies to a single domain environment. Before proceeding observe that all the mechanisms we describe here are variants of the Diffie-Hellman key agreement mechanism, [9]. We therefore assume that there is a globally agreed (large) prime p, and also a globally agreed primitive root g in the multiplicative group of non-zero elements in $\mathbb{Z}_p$. All calculations are performed modulo p.
We assume that the parameters are chosen so that the discrete logarithm problem is intractable, i.e. so that given an arbitrary h in $\mathbb{Z}_p^*$, it is computationally infeasible to find an s such that $g^s = h \bmod p$.

5.1 Single domain solutions

The concept of a domain can be likened to (and generally is considered the same as) a country. This domain can contain just one or multiple TTPs. We assume that there exists some level of trust between all TTPs, and that a single legal framework covers the whole of the domain.
Our model for single domain solutions is a simple one. We assume that there are a number of TTPs. Each interception agency is able to communicate directly with every TTP, and there exists an agreement between each interception agency and each TTP that the TTP will give up the appropriate data when presented with a warrant by the interception agency. This could, for example, be achieved by requiring TTPs to operate within a legally backed regulatory framework.
We suppose that each user X has two private/public key pairs: a send key pair ($S_s(X)$, $P_s(X) = g^{S_s(X)} \bmod p$) and a receive key pair ($S_r(X)$, $P_r(X) = g^{S_r(X)} \bmod p$).
All the private keys are registered with the user's TTP (T_X) prior to certification, and hence prior to use of the scheme. Certificates of the send and receive public keys are generated and marked as valid for encryption (certificates marked as valid for signature will be treated separately, and the private keys should definitely *not* be lodged with any TTP). We suppose that the certificates for all receive public keys are put into a universally available directory, and that all TTPs cross-certify each other and put all their cross-certificates in this directory; by this means any user can obtain the public receive key for every other user. In addition we suppose that every user is given a copy of the certificate for their own public send key; label the certificate for X's public send key $P_s(X)$ (which is signed by T_X): $\mathrm{Cert}_{T_X}(P_s(X))$. Note that this notation is not meant to imply that certificates only contain a copy of a user's public key;

at minimum they must also contain an identifier for the key's owner, and typically they will also contain an expiry date, an algorithm identifier, and other relevant information.

We now consider how the scheme will be used to provide key management for message encryption. We subsequently describe how warranted interception will operate. We suppose that user A (with TTP T_A) wishes to send a confidential message M to user B (with TTP T_B).

The sender A needs the public receive key ($g^{S_r(B)} \bmod p$) of the recipient B, and can get it either from a directory, or directly from B's TTP (T_B). A then uses its own private send key ($S_s(A)$) to compute the value

$$(g^{S_r(B)})^{S_s(A)} \bmod p = g^{S_r(B)S_s(A)} \bmod p.$$

A then selects a session key K_S, encrypts K_S using $g^{S_r(B)S_s(A)} \bmod p$, encrypts the message using K_S, and sends the following message to B:

$$A \longrightarrow B : \{M\}_{K_S}, \{K_S\}_{g^{S_r(B)S_s(A)} \bmod p}, P_r(B), \mathrm{Cert}_{T_A}(P_s(A)).$$

When B receives this message, B first verifies the certificate to obtain a copy of A's public send key: $P_s(A) = g^{S_s(A)} \bmod p$. B then combines this with B's private receive key $S_r(B)$ to obtain the value $g^{S_r(B)S_s(A)} \bmod p$. B can then use this to decrypt the session key K_S, and thence decrypt the message M.

B uses the supplied value $P_r(B)$ to determine which of its private receive keys should be used to compute the shared secret key with A. Note that, unlike the symmetric mechanisms described in Section 4.1, this mechanism *does* provide DTI requirement 12, because (implicitly or explicitly) B checks all the key data sent with the encrypted message.

To see how warranted interception operates we simply observe that knowledge of any user's private send and receive keys will immediately enable all messages sent or received by that user to be decrypted (since all the necessary public keys are always sent with a message). This is the reason for introducing separate send and receive keys. Thus, if an Interception Agency has a warrant to intercept A's communications, then this warrant is passed to A's TTP T_A. The TTP then hands over A's private send and receive keys, which can then be used to provide access to all A's in- and out-going messages.

As described, the mechanism does not provide time-bounding of warrants. Of course A and B can request new send or receive key pairs whenever they wish, giving a certain measure of time-bounding. However, to introduce a more sophisticated method of time-limiting warrants, we need to first describe the multi-domain version of the mechanism (see below).

In general, the mechanism offers a significant advantage over the corresponding symmetric cryptography based mechanism in reducing the possibility of TTP bottle-necks, since none of the TTPs need to be on-line and all public keys are distributed by means of certificates.

Finally we observe that this mechanism is not appropriate for a multi-domain environment. This is because we assumed that the interception agency had access to all the TTPs, which will not be the case in an international environment (see DTI requirement 13). As we see below,

to solve this we introduce a key-derivation technique similar to that used in the symmetric crypto based multi-domain mechanism introduced in Section 4.1.
Note that the single domain case is where most of the work in the public domain lies, but unfortunately much of these results are not very useful in the international case. The harder problems would appear to lie in devising multiple domain solutions that are both efficient and secure, and this problem is what we next consider.

5.2 Multiple domain solutions

The boundary line between solutions designed to operate in multiple domains within a single country, and international solutions is very fine. It can be argued that multiple domain and international key escrow schemes both have the same requirements. Fundamentally, international solutions do not rely on there being trust between domains, and need to be flexible enough to allow cooperating domains (countries) to implement different cryptography polices on the domestic and international use of encryption in a coherent way.
The need for key escrow systems which support international use is by now well documented. In addition to the original papers describing the RH scheme, [14, 16], the 1995 paper of Frankel and Yung, [10], discusses such a need. We now describe the RH scheme in the context of our previous discussions.

The Royal Holloway scheme The scheme which has become known as the *Royal Holloway (RH) Scheme* was first described in a pair of papers presented at conferences in 1995, [14, 16]; an elaborated description of the scheme was presented in [15]. We specify the scheme here as a natural evolution of the schemes we have already described, in an attempt to clarify the motivation behind its design.
If we consider the Diffie-Hellman based scheme described in Section 5.1, it is straightforward to see that one obstacle to its use in an international context is that the sender does not have an obvious means to obtain the recipient's receive public key. However, if the receive public key is a function of a key shared between TTPs (as in the second mechanism described in Section 4.1), then the sender's own TTP can provide the necessary information. This is the basis of the RH scheme, which we now describe.
We suppose that the TTPs acting as KEAs are trusted by their users and by the interception authority in their domain. We let T_X denote the trusted third party of user X. Just as in Section 5.1, we suppose that each user has two public/private key pairs, a send key pair and a receive key pair. Again, just as before, every user's *send* key pair is registered with their own TTP, and the TTP retains a copy of the private send key and generates a certificate for the public send key. The only difference is that the *receive* key pair for any user is a deterministic function of a secret key shared by a pair of TTPs, as we now describe.

Suppose A (with TTP T_A) wishes to send a secret message M to B (with TTP T_B). Analogously to the second mechanism in Section 4.1, we suppose that T_A and T_B share a secret key $K_{T_AT_B}$, and have also agreed on a Diffie-Hellman key generation function f, which on input of a key K and an identity value ID_X returns a secret receive key for user X. Then, the secret receive key for user B (for use when receiving messages from clients of TTP T_A) will be $S_r(B) = f(K_{T_AT_B}, \mathrm{ID}_B)$, where ID_B is a value uniquely identifying user B.

Observe that this means that a different receive key pair will be needed for each TTP from whose clients a user wishes to receive mail.

We can now describe the protocol. We use identical notation to that established above. We suppose that, when A chooses his/her TTP, a send key pair for A is chosen $(S_s(A), P_s(A) = g^{S_s(A)} \bmod p)$, and that T_A generates a certificate for $P_s(A)$ and passes it to A (T_A also retains a copy of $S_s(A)$. Label this certificate $\mathrm{Cert}_{T_A}(P_s(A))$.

1. A sends a message to T_A that he wants to communicate with B (who has TTP T_B).
2. T_A generates the private receive key for B (for receiving messages from clients of T_A) as $S_r(B) = f(K_{T_AT_B}, \mathrm{ID}_B)$ and computes the corresponding public receive key for B, i.e. $P_r(B) = g^{S_r(B)} \bmod p$. T_A now sends $P_r(B)$ to A via an authenticated channel. Note that this channel does not need to be confidentiality preserving, and hence the key could be sent in a certificate via a 'regular' communications channel. In fact the key could even be generated in advance and lodged in a public (not necessarily trustworthy) directory. Of course, there does need to exist a secrecy-preserving channel between A and T_A for the transfer of A's private key whenever it is changed, but this will typically occur relatively infrequently (e.g. once a week).
3. A computes a shared secret key as
$$K(A,B) = P_r(B)^{S_s(A)} \bmod p,$$
generates a random session key K_S (for encrypting M), and sends the following to B:
$$\{M\}_{K_S}, \{K_S\}_{K(A,B)}, \mathrm{Cert}_{T_A}(P_s(A)), P_r(B).$$
Note that $P_r(B)$ is present both to enable key escrow and to enable B to determine which of its private receive keys should be used to construct the shared secret with A.
4. B verifies the certificate containing A's public send key and computes
$$K(A,B) = P_s(A)^{S_r(B)} \bmod p.$$
This provides the means to decrypt K_S and thereby recover M.

First note that B only really needs to verify the Certificate for A's public key if the 'shared secret key' $K(A,B)$ is to be used to support integrity protection as well as confidentiality protection. Note also that if B is going to verify the certificate for A's public key, then B needs T_A's public verification key. Thus, should certificate verification be needed, on the first occasion that B receives a message from a client of T_A, B will need to ask his TTP to provide an integrity protected copy of T_A's public

verification key. This could be achieved by sending a certificate via a 'regular' communications channel, or via a public directory.

It is worth remarking at this point that the scheme is very similar to the single domain public key solution. As we have already mentioned, the only significant difference is that every entity's receive key pair is a deterministic function of a key shared by two TTPs.

To see how escrow works we need to consider four cases.

- If the Interception Agency is in A's domain and has a warrant to intercept all A's messages (or at least all messages sent by A), then the warrant is passed to T_A, who hands over $S_s(A)$. This enables all messages sent by A (to any domain) to be read, by combining the secret key with the public key sent with the encrypted message. This explains why the TTP T_A needs to be equipped with A's private send key $S_s(A)$; if this key was not in the possession of T_A then escrow would be much more difficult to achieve (and would have to be handled on a recipient by recipient basis).
- If the Interception Agency is in B's domain and has a warrant to intercept all B's messages (or at least all messages received by B), then the warrant is passed to T_B, who hands over $S_r(B)$. This enables all messages received by B (from clients of T_A) to be read, and thus T_B may need to hand over a selection of such keys, one for each TTP from whose clients B may receive messages.
- If the Interception Agency is in A's domain and has a warrant to intercept all B's messages (or at least all messages received by B), then the warrant is passed to T_A, who hands over $S_r(B)$. This enables all messages received by B from clients of T_A to be read.
- If the Interception Agency is in B's domain and has a warrant to intercept all A's messages (or at least all messages sent by A), then the warrant is passed to T_B, who can only help by providing individual shared secret keys.

Thus escrow is relatively simple in all but the last case. Given that the first two cases are much more likely to occur than the last case, we can say that DTI requirement 7 is 'mostly' met.

The scheme, as described, does not permit time-bounding of warrants, except that all entities can change their send key pairs as often as they wish. To enable receive key pairs to be changed regularly, which will provide the desired time-bounding, is simple to arrange. When computing receive private keys as a function of a TTP shared secret and an identity, a date stamp can also be added into the scope of the function, which means that every day a different receive key will be generated automatically. This idea is already described in [14, 15]. Note that there is a performance penalty associated with time-bounding, which increases as the time-bounding granularity becomes finer, since every new secret receive key needs to be transferred to its owner.

The above scheme also appears to meet all the DTI requirements. Requirement 9 is met by allowing the sender to change their key whenever they wish. Requirement 12 is met because, as in the previous mechanism, B checks all the fields in the received message (either implicitly or explicitly).

We conclude by comparing the mechanism with the symmetric mechanisms of Section 4.1. Although the general communications structure is closely analogous to the second mechanism in Section 4.1, there are the expected efficiency gains we would expect from using a public key solution. Notably, the communications between the sender and his/her TTP could be 'off-line', i.e. the required public key could be obtained via a directory, and also the message recipient has a minimal need for communication with his/her TTP (the recipient needs only contact his/her TTP for new receive key pairs, which could be done on a daily or weekly basis). There is also only a need for a shared key between TTPs, and not a shared encryption algorithm. Finally, the RH mechanism meets all the DTI requirements, which is not quite the case for the symmetric mechanisms.

A variation on the RH scheme We now consider a variation on the RH scheme; note that other variations, including support for 'split escrow', have been described in [15]. We suppose now that both the send and receive key pairs for every entity are deterministically generated as a function of keys shared between pairs of TTPs. In fact this means that the send and receive key pairs can actually be the same, and so from now on we just refer to the key pair for user X, written $(S(X), P(X) = g^{S(X)} \bmod p)$. This also means that all key pairs are specific to the TTP of the user who is to be communicated with, i.e. every user will have a set of key pairs, one for each TTP with whose clients the user wishes to exchange secret messages.

This variation of the RH scheme thus has the advantage that the keys produced can be used for two-directional communications (this would not be so simple for the 'standard' RH scheme because a receiver's TTP has no direct way of finding the sender's public send key). Moreover the TTPs do not have to archive the secret keys of their users. It does have the disadvantage (by comparison with the RH scheme) that the users initially have no method of controlling exactly when the values of their secret send keys are changed. However, this is not an abnormal situation: mobile phone users and users of many current encryption/security products, do not have any control over the values of their keys.

The definitions used in the revised scheme are almost the same as before. We suppose that A (having TTP T_A) wishes to send a secret message M to B (with TTP T_B). We also suppose that, in advance of the use of the protocol, A is equipped by T_A with key pairs (and public key certificates) for use with clients of a variety of TTPs (including T_B). To simplify the description below we suppose that $(S(A), P(A))$ is A's Private/Public Key pair for use in exchanging messages with clients of T_B, and $(S(B), P(B))$ is B's Private/Public Key pair for use in exchanging messages with clients of T_A.

The protocol operates as follows.

1. A sends a message to T_A that he wants to communicate with B (who has associated TTP T_B).
2. T_A computes the private key for B as $S(B) = f(K_{T_A T_B}, \mathrm{ID}_B)$ and computes the corresponding public key for B as $P(B) = g^{S(B)} \bmod$

p. T_A now sends $P(B)$ to A via an authenticated channel (as previously this would typically be by means of a certificate, or even conceivably via an untrusted third party).

3. A computes a shared secret key as

$$K(A,B) = P(B)^{S(A)} \bmod p,$$

generates a random session key K_S, and sends the following to B:

$$\{M\}_{K_S}, \{K_S\}_{K(A,B)}, \mathrm{Cert}_{T_A}(P(A)), P(B)$$

4. B verifies the certificate containing A's public key and computes $K(A,B) = P(A)^{S(B)} \bmod p$. This provides the means to decrypt K_S and thereby recover M.

As with the 'standard' RH scheme, B only needs to verify the certificate for A's public key if $K(A,B)$ is to be used to support authentication as well as encryption, and, in this case, B will need to be provided with T_A's public verification key.

Escrow works even more simply than for the 'standard' scheme. In each of the four cases described above, the TTP concerned can hand over the appropriate private key for the entity named in the warrant.

One possible shortcoming in the scheme is that a sender now has no control over their key pair, and cannot change it 'at will' (see DTI requirement 9). However, by including a date stamp within the scope of the function used to compute private keys, all key pairs are automatically of time-limited validity, and hence the scheme can be made to support full time-bounding (and, to all intents and purposes, DTI requirement 9).

Another possible shortcoming is that every user must store a number of key pairs, one for each TTP with whose clients they wish to exchange secret messages. Whilst this is not ideal, this is already largely true for the 'standard' RH mechanism, since every user of that scheme must store a receive key pair for each TTP from whose clients they wish to receive secret messages.

In terms of efficiency, the protocol is very similar to the previous scheme. Like the previous scheme, the sender of a message will need to obtain the recipient's public key from an on-line entity (at least the first time a message is sent), although the recipient will normally have all that he/she needs to process a received message.

5.3 Cheating

One method of modelling a general class of key escrow schemes based on a public key infrastructure, derived from [4], is to divide an encrypted communication into four components.

C1 The actual message encrypted with a symmetric encryption algorithm, using a random session key K_S.

C2 The session key K_S encrypted using the public key of the receiver.

C3 The session key K_S encrypted using the public keys of one or more TTPs.

C4 Other data.

Generally a user will encrypt the session-key under the public keys of the receiver, the sender's TTPs and the receiver's TTPs. These TTPs, who are able to decrypt the message but are not actually sent the message, are known as *virtual addressees*. It is also possible to add more virtual addressees to the message, for instance TTPs in any domain through which the encrypted message travels (although in this case the sender must know the route that messages take), making this a fairly flexible concept.

Not all these components may be present in each of the current key escrow schemes. For instance those schemes that escrow the user's secret keys, e.g. the RH scheme (and its variations), tend to have no **C3** component since the TTPs already have access to the appropriate secret key(s). In fact the RH scheme does not fit this model very well, since the RH scheme is a combined key escrow and key distribution scheme, and has meeting DTI requirement 1 as one of its objectives.

This model is the basis of several key escrow systems: TIS-CKE [1, 24], IBM SecureWay [11], AT&T CryptoBackup and many others, and has the advantage that the users do not have to deposit secret key information with KEAs beforehand. The flexibility of choice of TTPs also makes it suitable for use in the international environment.

Although this approach is very flexible, unfortunately it is not fraud-resistant. Its main drawback is that only the TTP and receiver can check whether the third component actually contains the correct session key; and the TTP can only detect this fraud after a lawful wiretap. Hence, by sending random data in place of **C3**, unilateral abuse is possible. This can be prevented by the addressee's client software recalculating and validating **C3** before decryption. However abuse by colluding of sender and receiver, through a one-time manipulation of the software, is still possible.

In [22] and [23], the authors address this problem using a novel concept called *Binding cryptography*. They add a fifth component (**C5**), to the communication, which contains 'binding data', with the idea that any (third party) monitor who has access to components **C2**, **C3** and **C5** (but not any additional secret information) can determine whether the session keys encrypted in **C2** and **C3** are the same, but not actually obtain any information on the actual session key. How effective this scheme will be in preventing cheating remains to be seen.

Finally we note that attempting to prevent 'cheating', while useful, is always likely to be of limited value, since once there is an infrastructure supporting the provision of authenticated channels between users, any pair of users can establish a shared secret key without any support from TTPs simply by using Diffie-Hellman key agreement.

5.4 Variations to fit different environments

We now briefly consider two other possible requirements on key escrow schemes. Firstly note that, within the context of international communications, it is possible that some countries may not want encrypted information crossing their jurisdiction without them having access to decrypt it. Obviously, this is a matter of policy agreement between the

domains concerned, although we note that it is unlikely to be a very common requirement.

Indeed, such a requirement is specifically ruled out by the ITU rules. To quote from ITU (1992) CV/Article 40 on Secret Language (para. 506):

> Members which do not admit private telegrams in secret language originating in or destined for their own territory must let them pass in transit, except in the case of suspension of service provided for in Article 35 of the constitution.

Nevertheless there may in certain special circumstances be a need for key escrow schemes which support it.

Very few of the schemes which have so far been published address this problem, with the exception of TIS-CKE [1, 24], and the scheme in [6]. The TIS-CKE scheme operates using the notion of virtual addressees.

More generally, if we consider the components **C1**–**C4** of encrypted messages, as introduced in the previous section, it is fairly easy to add extra encryptions within component **C3** to enable Law Enforcement authorities in any domain that the encryption may pass through to decrypt it. Obviously the sender is going to have to know the route that the message takes beforehand.

Secondly observe that it has been suggested that, to make key escrow more costly and hence restrict its use, the communicating parties should retain (and not escrow) a variable amount of the key material needed for message recovery. This retained key material must then be recovered by the Interception Agency by exhaustive search techniques. Ideas of this type have been explored in [2, 3, 21], and in [11] this is called the *residual work factor*. This option could be used to increase the overall workload involved in key recovery so as to discourage 'casual' recovery requests and/or to inhibit 'mass decryption' of communications. This would typically be implemented by giving the KEAs all but, say, 40 bits of the session-key, thus requiring the interception authority to do an exhaustive search over the remaining bits.

One major problem with such an idea is how to decide the amount of key material retained. If it is too small then there is little point. If it is large enough to make a difference, then it will impede the legal process. It would seem that this requirement is also unlikely to become a common one, since the only parties to gain would be the manufacturers of the computing equipment necessary to search through the large numbers of keys involved. Instead of putting artificial hurdles in the way of the legal interceptor, a better solution might be to ensure that the legal and auditing processes involved in providing access to both intercepted messages (typically involving a network provider) and escrowed keys (via the KEA) prevent simple abuse. In fact, whilst the bulk of network traffic and stored data remains unencrypted, as will probably be the case for some years to come, the main threat is not large scale key escrow but large scale interception of unencrypted traffic!

6 Conclusions

We have described the purpose of key escrow schemes and certain fundamental properties which such schemes are likely to need to satisfy. We have considered how we might adapt conventional key management schemes to provide key escrow and key recovery services. We have observed that, in an international escrow context, whilst adapting symmetric cryptography based key management schemes results in potentially usable mechanisms, using a conventional certificate-based key distribution infrastructure (based on asymmetric cryptography) is not really appropriate. This latter observation, combined with a desire to avoid some of the network bottle-neck problems associated with symmetric key management methods, led us to a description of a family of technical solutions based on use of the Diffie-Hellman key agreement scheme. These solutions, which are all variants of the RH scheme, have certain advantages over the symmetric cryptography schemes, although in some cases these advantages may not be terribly significant.

We have not attempted to provide an exhaustive list of all proposed mechanisms; there are by now a large number of such mechanisms, designed to meet a range of possible user requirements. For a list of mechanisms the interested reader is referred to the excellent survey by Denning and Branstad, [8]. It is likely that new mechanisms will continue to be devised, particularly given that the operational use of these schemes is likely to grow rapidly in the next few years.

Finally, if key escrow is to become a useful part of the secure networks of the future, then one major challenge will be to integrate key escrow techniques into secure network protocols (e.g. SSL and Secure IP) and secure distributed applications (e.g. MOSS and Internet Payment Protocols).

Acknowledgements

The authors would like to acknowledge the many people who have offered helpful comments and suggestions, without which the paper would have been much the poorer. We would particularly like to thank Mark King, Keith Martin, Fred Piper and Peter Wild. The authors, however, are solely responsible for all remaining errors.

References

1. D.M. Balenson, C.M. Ellison, S.B. Lipner, and S.T. Walker. A new approach to software key escrow encryption. Draft, August 1994. Trusted Information Systems, 3060 Washington Rd., Glenwood, MD.
2. M. Bellare and S. Goldwasser. Encapsulated key escrow. Technical Report 688, MIT Laboratory for Computer Science, April 1996.
3. M. Bellare and S. Goldwasser. Verifiable partial key escrow. In *Proceedings of the 4th Annual ACM Conference on Computer and Communications Security*, 1997.

4. M. Bellare and R.L. Rivest. Translucent cryptography — An alternative to key escrow, and its implementation via fractional oblivious transfer. Technical Report 683, MIT Laboratory for Computer Science, February 1996.
5. L. Chen, D. Gollmann, and C.J. Mitchell. Key escrow in mutually mistrusting domains. In M. Lomas, editor, *Security Protocols — International Workship, Cambridge, UK, April 1996*, pages 139–153. Springer-Verlag LNCS 1189, 1997.
6. L. Chen and C.J. Mitchell. Key escrow in multiple domains. In *Proceedings of INFOSEC'COM 97, Paris*, June 1997. To appear.
7. D.E. Denning. Descriptions of key escrow systems, February 1997. See: http://www.cs.georgetown.edu/~denning/crypto.
8. D.E. Denning and D.K. Branstad. A taxonomy for key escrow encryption systems. *Communications of the ACM*, 39(3):34–40, March 1996.
9. W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Trans. Inform. Theory*, IT–22(6): 644–654, 1976.
10. Y. Frankel and Moti Yung. Escrow encryption systems visited: Attacks, analysis and designs. In D. Coppersmith, editor, *Advances in Cryptology: CRYPTO '95*, pages 222–235. Springer-Verlag LNCS 963, 1995.
11. IBM. IBM SecureWay key recovery technology, 1997. See ftp://service2.boulder.ibm.com/software/icserver/doc/keyrec.pdf.
12. International Organization for Standardization, Genève, Switzerland. *ISO/IEC 9798–2, Information technology — Security techniques — Entity authentication — Part 2: Mechanisms using symmetric encipherment algorithms*, December 1994.
13. International Organization for Standardization, Genève, Switzerland. *ISO/IEC 11770–2, Information technology — Security techniques — Key management — Part 2: Mechanisms using symmetric techniques*, 1996.
14. N. Jefferies, C.J. Mitchell, and M. Walker. Trusted Third Party based key management allowing warranted interception. In *Proceedings: Public Key Infrastructure Invitational Workshop, MITRE, McLean, Virginia, USA, September 1995*. National Institute of Standards and Technology, 1995.
15. N. Jefferies, C.J. Mitchell, and M. Walker. Practical solutions to key escrow and regulatory aspects. In *Proceedings of Public Key Solutions '96, Zurich*, September 1996.
16. N. Jefferies, C.J. Mitchell, and M. Walker. A proposed architecture for trusted third party services. In E. Dawson and J. Golic, editors, *Cryptography: Policy and Algorithms — Proceedings: International Conference, Brisbane, Australia, July 1995*, pages 98–104. Springer-Verlag LNCS 1029, 1996.
17. L.R. Knudsen. Key escrow schemes — properties and options: New schemes for warranted interception, 1996. Preprint.
18. A.K. Lenstra, P. Winkler, and Y. Yacobi. A key escrow system with warrant bounds. In D. Coppersmith, editor, *Advances in Cryptology — CRYPTO '95*, pages 197–207. Springer-Verlag LNCS 963, 1995.

19. NIST. Escrowed Encryption Standard (EES). Federal Information Processing Standards Publication (FIPS PUB) 185, 1994.
20. US Interagency Working Group on Cryptographic Policy. Enabling privacy, commerce, security and public safety in the global information infrastructure. See http://www.cdt.org/crypto/clipper_III, 17th May 1996.
21. A. Shamir. Partial key escrow: A new approach to software key escrow. The Weizmann Institute, presentation at NIST Key Escrow meeting, Sept. 15 1995.
22. E.R. Verheul, B.-J. Koops, and H.C.A. van Tilborg. Binding cryptography. A fraud-detectible alternative to key-escrow proposals. *Computer Law and Security Report*, pages 3–14, January–February 1997.
23. E.R. Verheul and H.C.A. van Tilborg. Binding Elgamal: a fraud-detectable alternative to key-escrow proposals, 1997. To be presented at Eurocrypt '97.
24. S.T. Walker, S.B. Lipner, C.M. Ellison, and D.M. Balenson. Commercial key recovery. *Communications of the ACM*, 39(3):41–47, March 1996.

This article was processed using the LaTeX macro package with LLNCS style