A Modular Public Key Infrastructure for Security Management

A Modular Public Key Infrastructure for Security Management

Abstract

A modular public key infrastructure is outlined which supports key escrow for law enforcement, a separate data recovery component for restoration of archived information, secure message encryption, and strong authentication. The design is flexible in concept and is based on publicly established cryptographic algorithms. One strength of the system is that certifying authorities and data recovery centers never have access to users' secret keys, nor are they revealed in the warrant process. Moreover, warranted law enforcement access is limited both in time and direction by the public key mechanism.

1. Introduction

Any design of a public key infrastructure (PKI) must delicately balance the needs of industry, government, law enforcement, and the individual user community. The evolution of fast, reliable networks and the increasing reliance of society on them to conduct business present unique challenges to industry and government to guarantee authenticated, secure communications while providing legitimate access for warranted law enforcement. Moreover, this must be accomplished on a global scale.

As the electronic community evolves, so must the infrastructure that supports it. As such, the PKI described herein is *modular* in form allowing for components to be mixed and matched as appropriate. Thus, Escrow Agents serve only to process law enforcement warrants (and perhaps to restore lost secret encryption keys to users), Certificate Authorities serve only to authenticate users, and Data Recovery Centers serve only to restore archived information. Any subset of these components may be deleted from the PKI while still preserving its functionality (albeit with fewer services). Moreover, it is *scalable* both on the protocol level (key sizes, bit fields, etc.) and the user level (multiple Escrow Agents, Certificate Authorities, etc.).

In the following we describe each of the PKI components as they would interact in the context of a larger security management infrastructure. We do not prescribe any particular encryption algorithms, signature schemes, or hash functions in this proposal as these will no doubt need to evolve through consultation between industry, government, and the user community. However, it is important that a standard set of protocols and algorithms be defined—we only require that, upon their definition, they be implemented amongst the suite of options included in any PKI compliant product.

The design of this PKI was undertaken assuming a number of criteria had to be met. Nevertheless, this still provided for many choices and, when presented with such, a decision was generally made in favor of decreased network and computational overhead, user friendliness, and interoperability. Flexibility was a design requirement whenever possible so that, for example, the needs of law enforcement, the courts, and users are robustly drawn in the warrant process: only those communications precisely specified by the warrant can be deciphered by any parties other than the communicants.

Criteria

The following is a list of general design criteria:

- All components of the PKI will be made public.
- A key escrow mechanism, in which participants agree to escrow their secret encryption key(s) for warranted law enforcement, must be accommodated.
- The required default protocols and algorithms have been thoroughly vetted in the public domain.
- Encryption algorithms are essentially unregulated, but must be registered. Encryption packages must always include the default algorithms.

- Data recovery of encrypted files is accommodated by a mechanism distinct from the Escrow Authority (i.e., the Escrow Agents are not a part of the data recovery process).
- The infrastructure must be scalable to accommodate a large number of users.
- Secure communication must not *require* the recipient to play an active part in the key exchange (i.e., a common session key can be computed by the sender alone).
- Interoperability, at the cost of increased overhead, can be achieved with *key encryption key* (KEK) systems, such as *Royal Holloway.*

Features

Some of the design features of the PKI include:

- Signature keys are not escrowed in any way: private signature keys are never shared.
- None of the Data Recovery Centers, Certificate Authorities, warranted law enforcement entities, Escrow Agents, nor legitimate users of the PKI can digitally sign as another user.
- Recovery of a user's *encryption* secrets within this design requires the (illegal) collusion of the users Escrow Agents together with his Certificate Authority.
- Isolation of Certificate Authorities from knowledge of users' encryption *and* signing secrets reduces their liabilities and overhead, promotes more public trust, and allows them to focus on authentication.
- Data recovery of encrypted files is accommodated by a mechanism distinct from the Escrow Authority.
- Law enforcement is symmetric, i.e., the communications between two users can be legally monitored via access to *either* user's Escrow Agents.
- The key escrow mechanism requires very little software overhead.
- Users have the option of generating their own private encryption secrets or having these secrets generated for them.
- Infrastructure overhead per message has been minimized (e.g., no LEAF).

(b)(1)
OGA

FBI

[redacted]

- A data recovery center (DRC) can (and most likely will) be distinct from an Escrow Agent. As a DRC, it need only protect its own secret encryption key. Thus, it avoids many of the legal responsibilities (and costs) associated with an Escrow Agent.
- Default protocols are specified throughout guaranteeing any two users a secure communication path.
- Certificate Authorities can supply signed public key encryption certificates only after escrow requirements have been satisfied.
- Dishonest users cannot bypass escrow in communications with legitimate users (unlike LEAF systems).

In order for a user level software package to be certified PKI compliant (and eligible for export) this proposal requires

- Software must be registered with Department of Commerce, together with documented source code and encryption executables.
- All default algorithms (for hashing, signing, and encryption) must be implemented.
- User level software must check that incoming protocols are as expected (date stamp is accurate, signatures and certificates are valid) and notify the user of anomalies.
- Software must enforce fully qualified user identities (user@xyz.com.us).

2. Authentication

As with other specific algorithms implemented under this PKI, we leave the choice of a secure protocol for signing messages (e.g., the federal *Digital Signature Algorithm*) to the collaboration of industry, government, and users' groups. The Certificate Authority (CA), whose roles and responsibilities are detailed below, must have a mechanism for identifying a particular user to the network infrastructure by binding a public signature key to the user's identity. Once this is accomplished, the user can send and receive authenticated cleartext messages across the network.

Authentication is truly the bedrock on which the PKI is founded. Thus, each user's secret signing key is his most important property, for without it he cannot prove who he is, and in the possession of an untrusted third party it allows that third party to masquerade as the individual. In this proposal, *users are entrusted with securing their own signing keys.* (This is not to preclude that a user might want to enlist the services of an Escrow Agent for this purpose — but it is not a requirement of the PKI).

A user follows the following procedure to enroll in the PKI:

- The user presents identification and public signing parameters to a certified CA (This might have to be done physically, with a floppy disk or smart card, or by a notarized document exchange through the mail. A Policy Authority will have to set these guidelines.).
- The CA forms a public signature certificate binding the user's identity to his public key:

$$\langle ID_A, \text{public signature key, expiration date, ...} \rangle_{CA}$$

  where $\langle\ \rangle_{CA}$ means the contents are signed by the CA, and ID_A is user A's identity. (The precise format of the certificate may, for example, conform to the X.509 protocol.)
- The CA sends the certificate to a Network Directory Server (or several such servers) which are designated *authoritative* for that user, or CA.
- Upon compromise, revocation, change, or expiration of a signature certificate, the *CA* is responsible for immediately notifying the user's authoritative servers of a change in status.

The Network Directory Servers (NDS) may be configured to operate like DNS (*Domain Name Service*) so that together they form a distributed data base of user signature certificates. In this scenario an individual NDS services requests for which it is authoritative and may also choose to cache frequently requested certificates for which it is not authoritative.

In order to handle the case that a cached certificate may have been revoked (analogous to the situation in which a purchase is attempted with a credit card that has recently been stolen or has expired) a request for an authoritative response (i.e., directly to the user's authoritative NDS) should be supported. Moreover, the Policy Authority should establish *time to live* guidelines to prevent the long term caching of certificates. To prevent a malicious replay of an authoritative response (within the time to live period) a CA may choose to offer a notary service that provides a sender or receiver with a time stamped, signed, public key certificate for any user in its domain.

The implicit tree structure from the root Policy Authority down through the Certification Authorities to users enables a chain of trust to be established between any two users. For example, if User A has trusted Certification Authority CA_A while User B has trusted Certification Authority CA_B (all under the Policy Authority PA), then there exists a common parent CA (which may be PA itself) to both CA_A and CA_B. Trust of a parent of a trusted Certification Authority is implicit since the binding of an identity to a public signature key is based on the signature of the parent.

This chain of trust can be extended between different domains (i.e., between users under different Policy Authorities) if PAs cross-certify each other. This requires mutual agreement between domains (e.g., between different countries) that in particular includes agreement on signing parameters. Of course, cross-certification may take place at the Certification Authority level as well, say between CAs of a given company either within the same domain or between subsidiaries in different geopolitical domains.

3. Key Escrow

Encryption in this PKI allows for any registered user to communicate securely with any other registered user provided they share a common cryptographic engine. When key escrow is mandatory, each user will be required to provide a copy of his secret encryption key(s) to one or more certified Escrow Agents (EA). When not mandatory, a user may choose to escrow his keys with a trusted third party or even act as his own Escrow Agent (in such a case split escrow may not offer any advantage). The following escrow mechanism is independent of this choice.

The following description assumes *split key escrow*, i.e., a user constructs two secrets and shares one secret with each of two Escrow Agents. There is no design requirement for split escrow. Indeed, keys can be split between one, two, or many Escrow Agents and users with differing numbers of escrow agents can be easily accommodated. However, there are clear benefits to be had with two and that is the system advocated here. In addition, it is understood that all arithmetic is computed modulo a fixed, universal prime p together with a universal base g. The sizes and structure of these universal parameters will have to obey well established, fully vetted mathematical guidelines and be constructed in a manner that is acceptable to the PKI community. For example, the use of an elliptic curve group for exponentiation (in this case g is a base point for a universally known elliptic curve and exponentiation is usually written as multiplicaton) may offer valuable overhead savings with no attendant loss of security when compared to exponentiation in the ring of integers modulo p.

We follow the standard convention that when a quantity is framed by <> it is presumed to be digitally signed.

To register for encryption services:

- User A generates two secret numbers u_A and v_A which form the basis for all his encryption services.
- User A escrows his secrets with the split Escrow Authorities EA_1 and EA_2 by generating two additional secret numbers u_1 and u_2 (which he must securely store for possible data recovery). Each Escrow Agent will have published a universal parameter g^{r_1} and g^{r_2} respectively, which allows User A to form the values $g^{r_1 u}$ and $g^{r_2 u_2}$. These values will be used to form keys to encrypt his secret information for sending to the Escrow Agents.
- User A sends the following package to his Escrow Agents:

$$A \rightarrow EA_1 \colon \langle g^{u_1}, E^{g^{u_1 r_1}}(u_A), ID_{CA}, \ldots \rangle_A$$

$$A \rightarrow EA_2 \colon \langle g^{u_2}, E^{g^{u_2 r_2}}(v_A), ID_{CA}, \ldots \rangle_A$$

  where E is a known, and fixed, encryption algorithm used by the Escrow Agents. The session key for the encryption is $g^{u_i r_i}$.
- Upon receipt, each Escrow Agent decrypts its part of the split secret, and computes g^{u_A} and g^{v_A}, respectively. Each archives the information sent by user A and sends back to User A (or his CA) the package

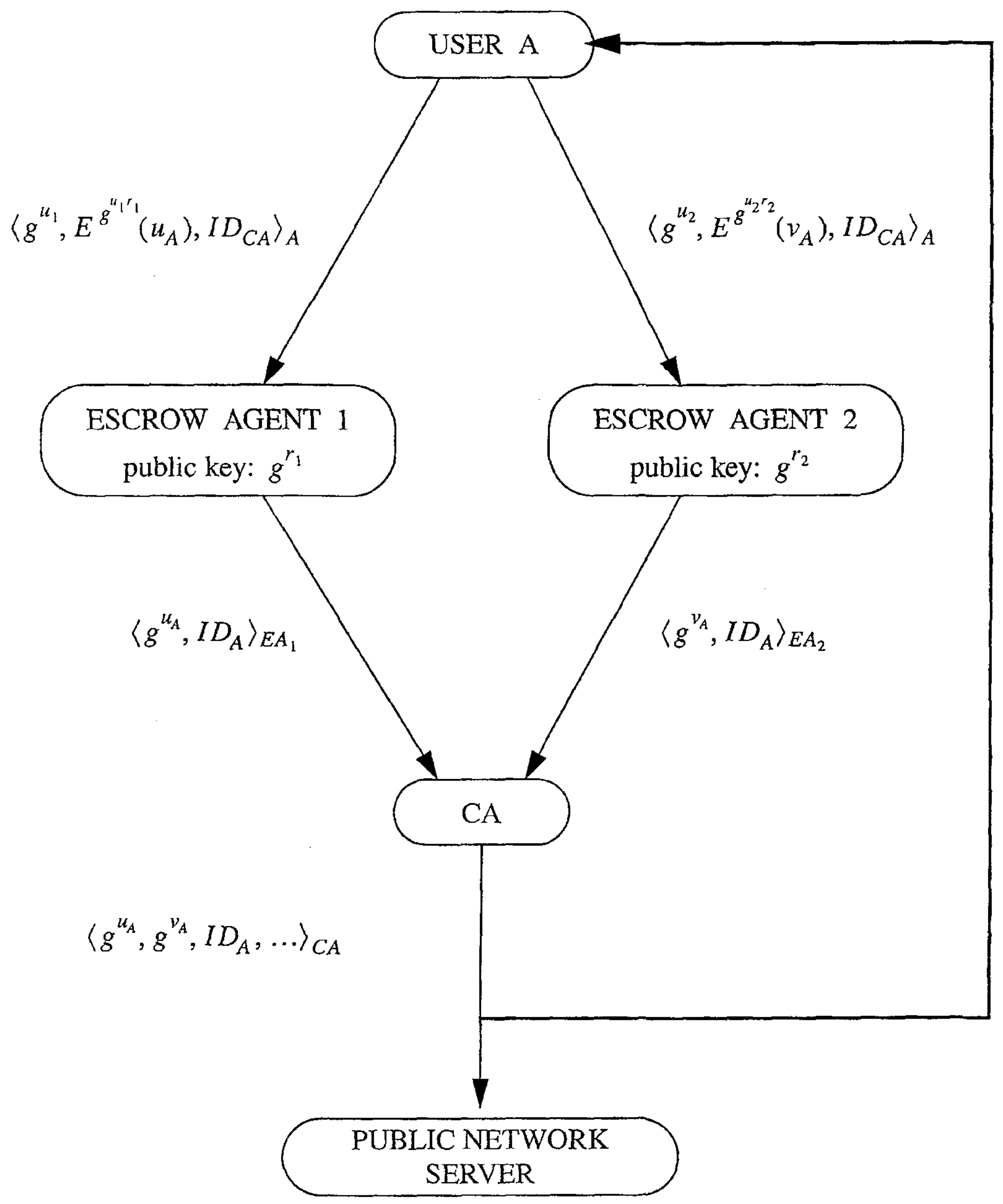
$$\langle g^{u_A}, ID_A \rangle_{EA_1}$$

$$\langle g^{v_A}, ID_A \rangle_{EA_2}$$

  respectively.
- Upon receipt of this information, either by User A or by User A's Escrow Agents, the CA computes and forms the signed public key certificate

$$\langle g^{u_A}, g^{v_A}, ID_A, \ldots \rangle_{CA}$$

  which it transmits to both User A (for verification) and to the Network Directory Server for placement in the public directory. The CA also records the identities of User A's Escrow Agents.


USER A
$\langle g^{u_1}, E^{g^{u_1 r_1}}(u_A), ID_{CA}\rangle_A$
$\langle g^{u_2}, E^{g^{u_2 r_2}}(v_A), ID_{CA}\rangle_A$
ESCROW AGENT 1
public key: g^{r_1}
ESCROW AGENT 2
public key: g^{r_2}
$\langle g^{u_A}, ID_A\rangle_{EA_1}$
$\langle g^{v_A}, ID_A\rangle_{EA_2}$
CA
$\langle g^{u_A}, g^{v_A}, ID_A, \ldots\rangle_{CA}$
PUBLIC NETWORK SERVER

Key Escrow Mechanism

4. Encryption

The encryption module is intended to be independent of whether or not there is mandatory key escrow. If escrow is not mandatory, user A will generate his secrets as described above but will bypass the Escrow Agent mechanism and directly send his (authenticated) public encryption keys to his CA. In either case, the CA registers user A's signed public encryption key certificate with an authoritative Network Directory Server(s).

The PKI is designed to allow any two users to construct a common *session key* (SK) for encryption. The SK can be used directly to key their common encryption algorithm or it can be used as a *key encryption key* (KEK). In the latter scenario, the sender generates a random encryption key and uses it to encrypt his message. He then sends $E^{SK}(\text{encryption key})$ along with the message. Using the common SK, the recipient decrypts this last packet, giving him the encryption key and therefore the ability to decrypt the message. On the surface, this adds unnecessary overhead to both the users and the network. However, there are advantages to using a KEK, including interoperability with other PKI proposals and reduced overhead for the user who wants to send the same message to many individuals.

Once User A and User B have been issued public key certificates (and have a common encryption package which may be the default standard, e.g., DES) they are ready to communicate securely. Let us assume User B wishes to send User A a secure message. The following sequence is then followed:

- User B queries a Network Directory Server for User A's public key certificate.
- Upon receipt of User A's certificate User B computes a session key by forming

$$SK = H\left\{H[H(g^{u_A u_B}, ID_B, ID_A, \text{month}),\text{day}] \oplus H[H(g^{v_A v_B}, ID_B, ID_A, \text{month}),\text{day}],\text{random}\right\}$$

  where H is a commonly held hash function, the month field is eleven bits (allowing for 2048 possibilities in this proposal, but may be any length), the day is five bits, and the random field is at least forty bits (to avoid the same session key if many messages are sent by User B to User A on the same day).
- User B encrypts the signed message under the session key SK and sends to User A

$$\text{control bytes,month,day,random,}E^{SK}(\langle m\rangle_B)$$

  where the *control bytes* indicate what encryption was used, the key lengths, and how to process the succeeding bytes. (User B might also send his public key certificate to avoid a network lookup by User A.) Alternatively, he may choose to use SK as a *key encryption key* (KEK) instead. In this case the message is encrypted with a random key, and an encrypted version of this random key is sent with the message. This latter encryption is performed under the session key.
- Upon receipt, User A computes the session key by forming $(g^{u_B})^{u_A}$ and $(g^{v_B})^{v_A}$. User A next decrypts the message and then authenticates it by checking that the signature is valid.

DOCID: 3219127

5. Law Enforcement

This PKI is designed to provide maximum flexibility for the needs of both law enforcement and the courts provided key escrow is enforced. In particular, it offers many layers of granularity in providing law enforcement warranted access to encrypted communications and/or encrypted archives, [redacted]

$$SK = H\left\{H[H(g^{u_A u_B}, ID_B, ID_A, \text{month}), \text{day}] \oplus H[H(g^{v_A v_B}, ID_B, ID_A, \text{month}), \text{day}], \text{random}\right\}$$

(b)(1)
OGA

where

FBI

- SK is the session key (or key encryption key) used to encrypt the particular message,
- H is a one-way hash function (e.g., SHA, the Federal standard Secure Hash Algorithm)

[redacted]

- The month field is comprised of eleven bits which identifies up to 2048 months both past and present, the day field is five bits, and the random field is at least forty bits.

The session key is *asymmetric* in that the order of ID_A and ID_B indicates the direction of the transmission. Thus, bidirectional traffic between two users on the same day that uses the same forty bit random pattern will use different session keys. However, since g^{u_A}, g^{v_A} and g^{u_B}, g^{v_B} are publicly known, knowledge of either pair u_A, v_A or u_B, v_B is sufficient to produce a session key given the month, day, and random fields. Therefore, a warrant served on either User A's or User B's Escrow Agent(s) will suffice to read their traffic.

The specific form of the session key allows for law enforcement to read traffic without explicitly being given either pair u_A, v_A or u_B, v_B. In particular, suppose a warrant is issued to read all the traffic sent by User B to User A during the month of March 1996. In terms of User A's Escrow Agents, the following sequence is then followed:

- User A's CA is queried for the identity of his Escrow Agents (the identity of the CA is manifest in User A's public key certificate).
- User A's split Escrow Agents EA_1 and EA_2 are served the warrant.
- EA_1 computes its half of the split key escrow u_A and sends to law enforcement (signed and encrypted):

$$EA_1 \rightarrow \text{Law Enforcement: } \langle H(g^{u_A u_B}, ID_B, ID_A, \text{month}), ID_B, ID_A \rangle_{EA_1}$$

- Similarly EA_2 computes its half of the split key escrow v_A sends to law enforcement

$$EA_2 \rightarrow \text{Law Enforcement: } \langle H(g^{v_A v_B}, ID_B, ID_A, \text{month}), ID_B, ID_A \rangle_{EA_2}$$

- Each day law enforcement hashes the day field into each of these packets, adds them together, and hashes in the per message random field to generate a session key.

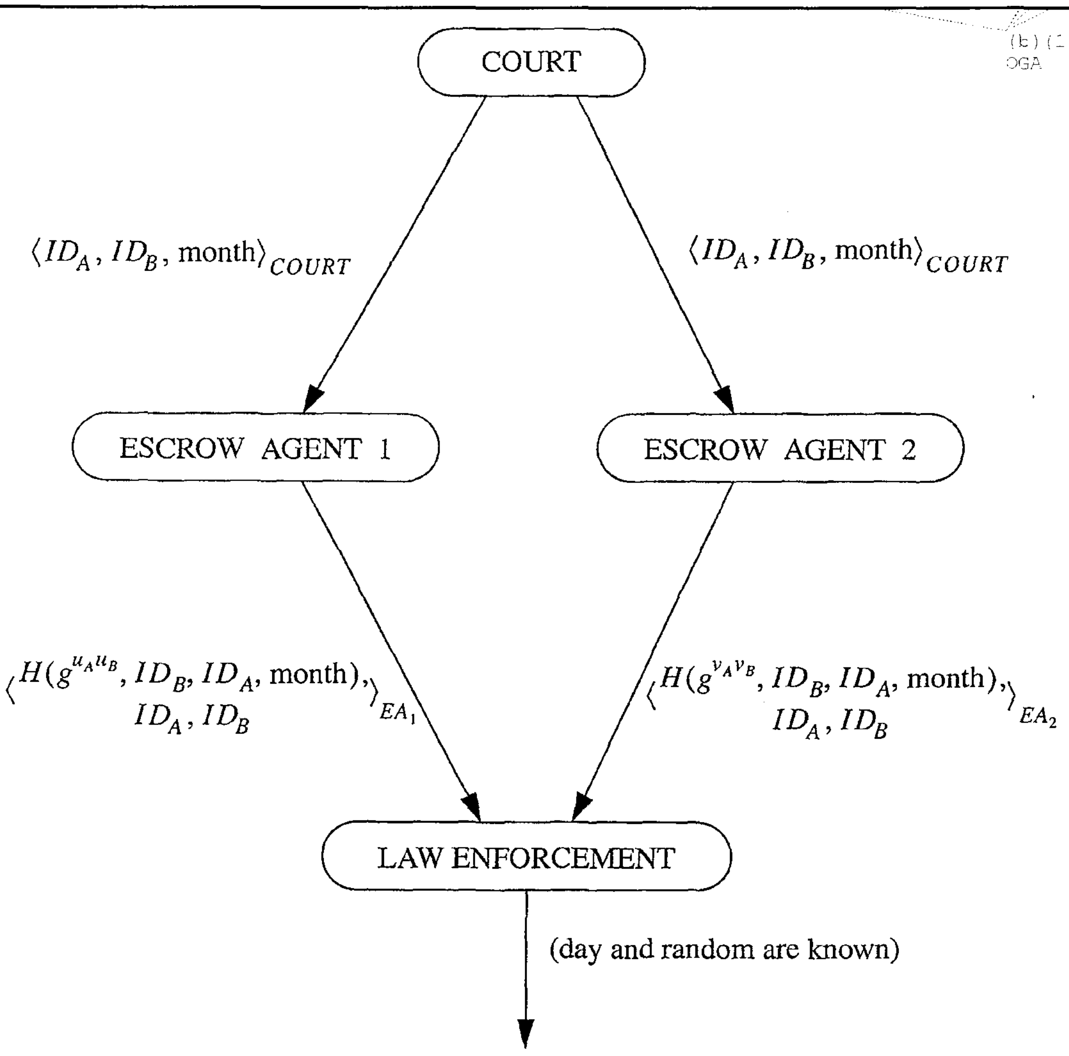
DOCID: 3219127

At this point law enforcement can read only that traffic specifically covered by the warrant. *No secret keys are revealed.* If the warrant covers a specific set of days, or even an extended period of time, the Escrow Agent generates several key packets that cover the specified time period. Similarly, key packets will have to be generated for each user and for each direction of transmission covered by the warrant. [redacted] Such circumstances should be extremely rare and governed by very stringent requirements.

Once the warrant expires, the user has no need to generate new [redacted]

(b)(1) OGA

FBI



COURT
$\langle ID_A, ID_B, \text{month}\rangle_{COURT}$
$\langle ID_A, ID_B, \text{month}\rangle_{COURT}$
ESCROW AGENT 1
ESCROW AGENT 2
$\left\langle H(g^{u_A u_B}, ID_B, ID_A, \text{month}), ID_A, ID_B \right\rangle_{EA_1}$
$\left\langle H(g^{v_A v_B}, ID_B, ID_A, \text{month}), ID_A, ID_B \right\rangle_{EA_2}$
LAW ENFORCEMENT
(day and random are known)

$$SK = \begin{array}{l} H(\, H(H(g^{u_A u_B}, ID_B, ID_A, \text{month}), \text{day}) \oplus \\ \quad H(H(g^{v_A v_B}, ID_B, ID_A, \text{month}), \text{day}), \text{random}\,) \end{array}$$

Law Enforcement Access

DOCID: 3219127

6. Data Recovery

In this PKI a user may contract with a data recovery center (DRC) to provide data recovery services for files encrypted on the user's disk. Outwardly, a DRC acts exactly like any user in the network: the DRC registers with a Certificate Authority, has its secret key(s) escrowed, and is a fully qualified user on the system. When a user wishes to store a file on his disk he encrypts it just as if he were sending it to his DRC, but writes it to his disk. Since the DRC is the legitimate target of the message, it can compute the session key that was used to encrypt it. Thus, if a user is unable to decrypt a disk file, he can send the file's header information (month, day, random) to his DRC who, in turn, can provide the user with the session key that decrypts the file.

From a law enforcement perspective, reading a user's disk files (given access to them) amounts to obtaining a warrant to read the traffic between the user and his DRC. Either the user's or the DRC's Escrow Agents can service this request. In this fashion, the DRC has no responsibility to provide services to law enforcement and must only protect its own [redacted]

FBI

According to this scenario, then, a user who receives a file encrypted with one session key will reencrypt the message with a new session key and write the result to disk. This latter session key can be reconstructed only by the user, his DRC, and the combined efforts of either's Escrow Agents when served with a warrant.

If a user does not elect to employ a DRC, he may simply archive received encrypted files, or in the case of encrypting a file for the first time do so as if mailing it to himself.

7. Roles and Requirements

Each of the component elements of the PKI has its own role and set of requirements in order for the infrastructure to interoperate efficiently. In this section we outline, in general terms, the major functions each component must provide.

Policy Authority (PA)

The Policy Authority establishes requirements for PKI components and certifies them compliant. Its role in the PKI is a continually evolving one that addresses and arbitrates the requirements of users, industry, government, and law enforcement. It is also responsible for:

- Fostering international agreements that promote interoperability.
- Developing and enforcing safeguarding standards for Escrow Agents.
- Determining default algorithms for digital signatures, encryption, and public key exchange.

- Establishing standards for Certificate Authorities and acting as their authentication agent.
- Developing new standards as the PKI infrastructure evolves.
- Issuing and responding to security alerts that affect the PKI.

Escrow Authorities (EA)

This PKI employs dual Escrow Agents within a given domain (domains will probably exist at the national level but may be distributed to lower echelons). The concept of split escrow immediately extends to any number of Escrow Agents servicing a particular user. Indeed, one can conceive of a single Escrow Agent (who now knows all of a user's secrets) to multiple Escrow Agents each archiving a proportional part of a user's secret encryption keys. In any scenario, the EA must provide a number of services:

- Escrow (and archive) users' (split) keys and protect them from unauthorized access.
- Process warrants for law enforcement.
- Provide Certificate Authorities with users' public key parameters which can then be signed and placed on the public network directory.
- Establish a secure channel to a user for exchanging data (the registration process requires that a user send his secrets to the EAs securely—the EA may have to provide publicly available software to accomplish this).
- Upon request, create secret encryption keys for a user.
- Be equipped to provide secret key recovery for each user in its domain (failure of a smart card, disk crash, etc. — this service may be at the Policy Authority's discretion).

Certificate Authorities (CA)

The role of a Certificate Authority in the PKI is made distinct from data recovery (although it may be a Data Recovery Center, as well) and encryption key escrow. It serves to authenticate users to the network by signing their public key certificates and forwarding their certificates to authoritative Network Directory Servers. It is conceivable that there may be multiple levels of authentication available corresponding to the level of rigor employed in identifying the user to the CA. Since only a CA is authorized to sign public key certificates, it must protect its signing secret(s) and assume some liability for their unauthorized disclosure. However, since it need protect no keys other than its own, the security requirements levied on a CA will be much less stringent than those of an Escrow Agent. Indeed, a CA might be little more than a secured workstation with appropriate network firewall protection. In practice, there will likely be a hierarchy of CAs that can establish a chain of trust between any two users. CAs will authenticate other CAs within their own domain with the top level CA being authenticated by a Policy Authority.

More specifically, a CA must provide the following services:

- Bind users' identities to their public signature keys (this may require physical identification — or more — for full authentication).

- Sign public key certificates (as received from the EAs in the case of encryption key escrow) and forward them to the user's authoritative Network Directory Server(s).
- Record the identities of each user's Escrow Agents, i.e., store $\langle g^{u_A}, ID_A \rangle_{EA_1}$ and $\langle g^{v_A}, ID_A \rangle_{EA_2}$. The availablity of this information outside of Law Enforcement will be determined by the Policy Authority.
- Escrow users' public key signature certificates for a time specified by law.
- Issue revocation certificates to the public network directory for users in its domain according to policies set forth by the PA.
- In accordance with PA policy, provide signed, time-stamped certificates of a user's signature key for revocation inquiries.

Public Network Service (PNS)

A Public Network Service maintains an up-to-date distributed directory of valid users together with their signed public signature and encryption certificates. This information is stored on Network Directory Servers in established formats and is accessed according to agreed to protocols (e.g., X.509). The PNS must also maintain key revocation lists for invalidated users and provide backup redundancy for network outages. The requirements for the PNS will be determined and enforced by the PA.