Key Recovery Server

Installation and Usage Guide

001.001.004

Table of Contents

List of Figures

List of Tables

Chapter 1. Introduction

The IBM Key Recovery product suite consists of the IBM KeyWorks Toolkit, the IBM Key Recovery Service Provider (KRSP), and the IBM Key Recovery Server (KRS). Components in the IBM KeyWorks Toolkit and the IBM KRSP encapsulate keys such that only authorized parties using the IBM KRS can recover the keys. This document describes the installation and usage of the IBM KRS.

1.1 Intended Audience

This document is intended to provide users and security administrators with an understanding of key recovery concepts, guidance in setting up a key recovery solution for an organization, and procedures for installing, configuring, and operating the IBM KRS.

Security administrators are assumed to have a working knowledge of public key infrastructures and network computing environments.

This document is intended for use by:

- Enterprise employees and security personnel
- Policy enforcement personnel
- Key recovery facilities security personnel

1.2 Documentation Set

The IBM KeyWorks Toolkit documentation set consists of the following manuals. These manuals are provided in electronic format and can be viewed using the Adobe Acrobat Reader distributed with the IBM KeyWorks Toolkit. Both the electronic manuals and the Adobe Acrobat Reader are located in the IBM KeyWorks Toolkit `doc` subdirectory.

- *IBM KeyWorks Toolkit Developer's Guide*
  Document filename: kw_dev.pdf
  This document presents an overview of the IBM KeyWorks Toolkit.
  It explains how to integrate IBM KeyWorks into applications and contains a sample IBM KeyWorks application.

- *IBM KeyWorks Toolkit Application Programming Interface Specification*
  Document filename: kw_api.pdf
  This document defines the interface that applications developers employ to access security services provided by the IBM KeyWorks Framework and service provider modules.

- *IBM KeyWorks Toolkit Service Provider Module Structure & Administration*
  Document filename: kw_mod.pdf
  This document describes the features common to all IBM KeyWorks service provider modules. It should be used in conjunction with the individual IBM KeyWorks Service Provider Interface Specifications in order to build a service provider module.

- *IBM KeyWorks Toolkit Cryptographic Service Provider Interface Specification*
  Document filename: kw_spi.pdf
  This document defines the interface to which cryptography service provider modules must conform in order to be accessible through IBM KeyWorks.

- *Key Recovery Service Provider Interface Specification*
  Document filename: kr_spi.pdf
  This document defines the interface to which key recovery service provider modules must conform in order to be accessible through IBM KeyWorks.

- *Key Recovery Server Installation and Usage Guide*
  Document filename: krs_gd.pdf
  This document describes how to install and use key recovery solutions using the components in the IBM Key Recovery Server.

- *IBM KeyWorks Toolkit Trust Policy Interface Specification*
  Document filename: kw_tp_spi.pdf
  This document defines the interface to which policy makers, such as Certificate Authorities (CAs), certificate issuers, and policy-making application developers, must conform in order to extend IBM KeyWorks with model or application-specific policies.

- *IBM KeyWorks Toolkit Certificate Library Interface Specification*
  Document filename: kw_cl_spi.pdf
  This document defines the interface to which certificate library developers must conform to provide format-specific certificate manipulation services to numerous IBM KeyWorks applications and Trust Policy (TP) modules.

- *IBM KeyWorks Toolkit Data Storage Library Interface Specification*
  Document filename: kw_dl_spi.pdf.
  This document defines the interface to which library developers must conform to provide format-specific or format-independent persistent storage of certificates.

1.3 References

Cryptography	*Applied Cryptography, Second Edition Protocols, Algorithms, and Source Code in C*, Bruce Schneier: John Wiley & Sons, Inc., 1996.
PKCS	*The Public-Key Cryptography Standards*, RSA Laboratories, Redwood City, CA: RSA Data Security, Inc.
X.509	*CCITT. Recommendation X.509: The Directory – Authentication Framework*, 1988. CCITT stands for Comite Consultatif Internationale Telegraphique et Telephonique (International Telegraph and Telephone Consultative Committee)

Chapter 2. Technology Overview

Because of the rise in popularity of the Internet and its huge potential for communication and commerce, the topic of security has percolated to the top of the list of major concerns of businesses, individuals, and governments. Fortunately, modern cryptographic techniques provide the foundation for security solutions. One the most useful of these techniques, data encryption, involves encrypting information using *cryptographic keys* to keep the data confidential to parties who know the keys.

In environments where data encryption is used, these keys become critical items that are needed to access encrypted information. For example, an enterprise may heavily utilize data encryption in its mission-critical applications for confidentiality and the enterprise's employees may possess knowledge of keys relevant to their daily work.

The ability to recover the keys used for data encryption is a pivotal issue to the enterprise and potentially to policy (law) enforcement officials. Reasons for recovering encryption keys may include:

- An individual has encrypted important information that needs to be reused in clear form
- A business needs access to employee-encrypted non-personal information and the employee is not available
- Law enforcement personnel obtain a court order giving them the right to access information (e.g., a search warrant)

The ability to recover keys is provided by key recovery technologies that are in existence today. They fall under the following two categories:

- Techniques involving some form of escrow of the key or key parts with a trusted party (e.g., Royal Holloway), and
- Encapsulation techniques that involve creating Key Recovery Fields (KRFs) that are mathematically related to, but not actually the key or parts of the key. These fields are associated with the encrypted data. Later, the key recovery fields can be used to recover the key.

IBM key recovery technology encapsulates keys. The IBM key encapsulation technique is based on the paradigm that a cryptographically encapsulated form of the key, a KRF, is made available to parties that require key recovery. The technique ensures that only trusted principals, possibly third parties called *Key Recovery Agents (KRAs)*, can perform unwrap operations on the encapsulated key block to retrieve the key material concealed inside.

When encapsulation schemes are used to conceal ephemeral versus long-term keys there is much greater privacy for the individuals; they can generate and keep their own long-term keys. Ephemeral keys can be recovered independently, so there is maximal granularity with respect to the recoverable data.

Advantages of ephemeral key encapsulation and recovery can therefore be summarized in the following:

- More privacy for individuals
- Fine granularity for recoverable keys
- No latency to obtain and use public keys
- No need for individuals to belong to any government-approved public key infrastructure (PKI)

For additional information about key encapsulation, refer to Chapter 5 in the *IBM KeyWorks Toolkit Developer's Guide.*

Chapter 3. Key Recovery Scenarios

Three operational scenarios are supported by IBM key recovery technology:

- Law Enforcement key recovery
- Enterprise key recovery
- Individual key recovery

In the law enforcement scenario, key recovery is mandated by the jurisdictional law enforcement authorities in the interest of national security and law enforcement. For a specific cryptographic product, the key recovery policies for multiple jurisdictions may apply simultaneously. The policies (if any) of the jurisdictions of manufacture of the product, as well as the jurisdiction of installation and use, need to be applied to the product such that the most restrictive combination of the multiple policies is used. Thus, law enforcement key recovery is based on mandatory key recovery policies; these policies are logically bound to the cryptographic product at the time the product is installed or shipped. Mechanisms for vendor-controlled updates of such law enforcement key recovery policies will be provided; however, organizations and end users of the product cannot modify this policy at their discretion. The Key Recovery Agents (KRAs) used for this scenario of key recovery need to be selected carefully to ensure that these agents meet the eligibility criteria for the relevant jurisdiction where the product is being used.

Enterprise key recovery allows enterprises to enforce stricter monitoring of the use of cryptography, and the recovery of enciphered data when the need arises. Enterprise key recovery is also based on a mandatory key recovery policy; however, this policy is set (through administrative means perhaps) by the organization or enterprise at the time of installation of a recovery enabled cryptographic product. The enterprise key recovery policy should not be modifiable or circumventable by the individual using the cryptographic product.

Individual key recovery is user-discretionary in nature and is performed for the purpose of recovery of enciphered data by the owner of the data, if the cryptographic keys are lost or corrupted. Since this is a nonmandatory key recovery scenario, it is not based on any policy that is enforced by the cryptographic product; rather, the product may allow the user to specify when individual key recovery enablement is to be performed. There are few restrictions on the use of specific KRAs, which allows the individual the flexibility to choose the KRAs. In each of the above scenarios key recovery may be desired. However, the detailed aspects or characteristics of these three scenarios are somewhat varied. Table 1 summarizes the specific characteristics of the different operational scenarios.

Table 1. Comparison of Key Recovery Scenarios

Properties	Law Enforcement	Enterprise	Individual
Mandatory key recovery	Yes	Yes	No
Escrow or recovery agents are approved	Yes	Yes	No
Recovery enablement needs to be noncircumventable	Yes	Yes	No
Dual-sided key recovery enablement	Maybe	Maybe	No
Use of workfactor when generating KRF	Maybe	No	No
KRF contains agent identification	Yes	Maybe	Yes
User registration needed at escrow or recovery agents	Maybe	Maybe	Maybe
User authentication information needed within KRF	Maybe	Yes	Yes
End-user knowledge/cooperation required	No	No	Yes

IBM's key recovery enabled cryptographic products are designed so that the key recovery enablement operation is mandatory, noncircumventable in the law enforcement and enterprise scenarios, and discretionary for the individual scenario. The KRAs that are used for law enforcement and enterprise scenarios must be tightly controlled so that the agents are validated to belong to a set of authorized or approved agents. In the law enforcement and enterprise scenarios, the key recovery process typically needs to be performed without the knowledge and cooperation of the parties involved in the cryptographic association. The law enforcement and enterprise officials communicating with approved KRAs will enable such key recovery.

The components of the Key Recovery Fields (KRFs) also vary somewhat between the three scenarios. In the law enforcement scenario, the KRFs must contain identification information for the escrow or recovery agents, whereas for the enterprise and individual scenarios, the agent identification information is not so critical, since this information may be available from the context of the recovery enablement operation. For the individual scenario, there needs to be a strong user authentication component in the KRFs to allow the owner of the KRFs to authenticate themselves to the agents. For the law enforcement and enterprise scenarios, the KRFs may contain recovery information for both the generator and receiver of the enciphered data.

IBM key recovery products support all three of the above scenarios for key recovery. The law enforcement and enterprise scenarios for key recovery are mandatory in nature and are enforced. On the other hand, the individual scenario for key recovery is discretionary and no policy checks on the KRA credentials are performed. The application/user requests key recovery operations at their discretion.

Chapter 4. Key Recovery Phases

Using IBM key recovery technology, the process of cryptographic key recovery involves three major phases. First, there is a *key recovery capability definition phase*, where the parties that desire key recovery performs some initialization operations with a party authorized to issue key recovery requests; these operations include obtaining a public key certificate for relevant Key Recovery Agents (KRAs). Next, parties that are involved in cryptographic associations have to perform operations to enable key recovery (such as the generation of Key Recovery Fields (KRFs))—this is typically called the *key recovery enablement* phase. Finally, authorized parties that desire to recover the data keys do so with the help of a Key Recovery Coordinator (KRC) and one or more KRAs—this is the *key recovery request* phase.

The entities involved in key recovery include the following:

- Key Recovery Officer (KRO) - The security officer representing an enterprise which controls the enterprise domain or a subdomain. The KRO is responsible for configuring the systems that use data encryption such that the appropriate KRFs are generated.
- Key Recovery Coordinator (KRC) - A software component that runs as a server listening for key recovery requests from authorized entities. Typically, this software is run within a secure facility.
- Key Recovery Agent (KRA) - A software component that runs as a server listening for requests to compute secondary keys for the KRC. IBM key recovery technology allows an arbitrary number of KRAs. This software also is assumed to run within a secure and separate facility.

While it is strongly recommended that the KRAs involved in the recovery phase be as isolated as possible to prevent collusion attacks, the IBM Key Recovery Server (KRS) product is flexible enough to allow these entities to be configured on one or several machines connected via a TCP/IP network.

4.1 Key Recovery Capability Definition Phase

In this phase, the KRA, the KRC, and the KRO must be issued RSA key pairs in the form of public key certificates and private keys. Each of the parties must safely store their private keys and not divulge them. The KRO must choose the KRAs that will be involved in key recovery and get their public key certificates. These public key certificates contain critical information that will be needed to generate the KRFs. The KRFs will be generated on key recovery enabled systems that have the IBM Key Recovery Service Provider (KRSP) installed and configured. These public key certificates must be sent to IBM for conversion into a form that the IBM KRSP can use. The resulting *sealed* configuration files must then be installed on all key recovery enabled systems. The certificates and private keys issued to the KRO and the KRC are used to digitally secure communication between them.

4.2 Key Recovery Enablement Phase

During the key recovery enablement phase, the IBM KRSP generates KRFs so that a given party that is authorized can recover the key associated within the KRFs. Since multiple parties may be authorized to recover a given key, a set of KRFs may be generated for a given key. These KRFs are contained in a single data block called a Key Recovery Block (KRB). Applications that have been written to the application programming interface (API), which are provided by the IBM IBM KeyWorks Toolkit, will be able to access the services of the IBM KRSP.

The purpose of a KRSP is to generate a correct KRB given an ephemeral symmetric key and to make sure that KRBs have not been tampered with subsequent to generation.

For additional information about key recovery enabled application development, refer to the *IBM KeyWorks Toolkit Application Programming Interface Specification.*

For additional information about KRSPs, refer to the *Key Recovery Service Provider Interface Specification*, which defines the interface to which KRSPs must conform in order to be accessible through IBM KeyWorks.

4.3 Key Recovery Request Phase

When a key is unavailable for any of the reasons previously outlined in Chapter 2, encrypted information is effectively "locked" until the key can be recovered. The KRB, which is associated with the encrypted data, is retrieved by the KRO. In the case of an encrypted file, the KRB is typically stored in a filename derived from the original encrypted file. In the case of an encrypted network communication, the encrypted data and the KRB are both sent on the network.

The KRO, using the IBM provided software, sends the KRB as part of a key recovery request to the KRC. The KRC acts as a front-end for coordinating key recovery requests. It distinguishes between law enforcement, enterprise, or individual originated key recovery requests, authenticating the requesting party when needed, and processing the KRB by sending requests to the appropriate KRAs that are involved in the key recovery phase. The identities of the KRAs are stored in the KRB.

Each KRA receives a request to compute a secondary key and sends it back to the KRC. When the KRC receives positive responses from each KRA, it can perform a number of decryptions to compute the actual original encryption key. This original encryption key is then sent back to the KRO, completing the recovery phase.

The recovered key can then be used to decrypt the encrypted data. Note that it is not necessary for the party recovering the key to be the party decrypting the data; however, this is often the case.

Chapter 5. Installation and Configuration

5.1 Key Recovery Server Components

The following software components are included in the IBM Key Recovery Server (KRS):

- Key Recovery Officer Client - This client software allows the KRO to submit key recovery requests to the KRC.
- Key Recovery Coordinator Server - This server software listens for requests from the KRO, includes authentication of KRO requests, communicates with a set of KRAs to obtain required secondary keys, and recovers the requested key information.
- Key Recovery Agent Server - This server software listens for requests from the KRC and computes secondary keys using its configured private key.
- Key Recovery Utility - This utility software allows the user to encrypt and decrypt an arbitrary file given a key. It can also generate the corresponding KRB if the system has the IBM KRSP installed and configured.
- Authentication Information Utility - This utility software allows the KRO to create an Authentication Information (AI) file that needs to be stored on each enterprise key recovery enabled system.

5.2 Choosing a Configuration

Every key recovery configuration involves a single KRC and an arbitrary number of KRAs. The first step in the installation and configuration process is to choose the number of KRAs that will be involved in the key recovery scenario. As previously stated in this chapter, increasing the number of KRAs provides added levels of protection against collusion between independent KRAs.

TCP/IP must be configured on each system in the configuration. Additionally, the KRC Server and KRA Servers listen for requests using TCP protocol. The TCP port numbers need to be chosen when they listen for requests. These TCP/IP address and ports will be needed in later configuration steps to help all parties to communicate.

Each KRA Server must be given a name by which to name its configuration data. This name must begin with the prefix “KRA_”.

Enterprise KROs must also choose a password that will authorize them to receive recovered keys. Each key recovery enabled system in the enterprise needs to have an AI file that is derived from this password so that each KRB is associated with this AI. Each KRB generated by KRSPs on key recovery enabled systems contains this AI. When a KRB is submitted to the KRC as part of a key recovery request, the KRO’s password is checked against the KRB for validation of the request. To create the AI file, the KRO must use the AI Utility (SETAUTHINFO.EXE) on a system that has the KRSP installed and configured.

Figure 1 shows an example configuration of a KRO, KRC, and KRA.

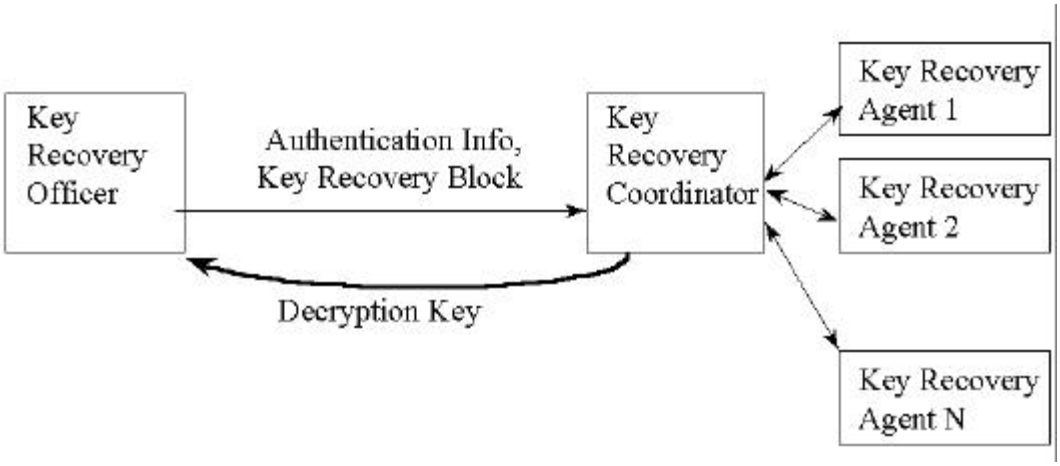

Figure 1. Example Configuration

5.3 Setting Up Secure Facilities

The KRC Server and each of the KRA Servers should run in a Key Recovery Server Facility (KRSF) as specified in Appendix A. Setting up a KRSF involves securing the site that houses the systems running the software as well as configuring the Windows NT 4.0 operating system properly. It is strongly recommended that these steps be followed before installation of the KRS. The KRO Client system should be physically secured because of the sensitive information that will be processed.

5.4 Installing the Software

For each system in the configuration, the relevant software components should be installed by following the instructions on the IBM KRS CD-ROM. The following are the minimum requirements for each system:

- Enterprise KRO - This system requires the Key Recovery Officer Client.
- Key Recovery Coordinator - This system requires the Key Recovery Coordinator Server and the Key Recovery Officer Client (for law enforcement requests).
- Key Recovery Agent - This system requires the Key Recovery Agent Server.

5.5 Obtaining Certificates and Private Keys

The enablement phase and the recovery phase of key recovery work together through the matching certificate and private key pairs that are configured on the systems involved.

A certificate and private key pair is needed for the following:

- Key Recovery Agent Servers
- Key Recovery Coordinator Server
- Key Recovery Officer Client

Only the certificates are needed for Trusted Anchors, which are used to verify the other certificates. The IBM KRS uses ASN/DER-encoded X.509 Version 3 certificates. The public/private key pair is comprised of 1024-bit RSA keys. The public key is stored in the certificate. The private key is DER-encoded, encrypted (optional), and stored in a file. **Note:** only private keys that are stored in the clear, or those encrypted using the SecureWay Encryption Utility *krutil.ex*e (distributed with the CD-ROM) are currently supported.

The public key certificates must be sent to IBM for conversion into a form that the IBM KRSP can use. The resulting *sealed* configuration files must then be installed on all *key recovery enabled systems* where KRBs will be generated.

A sample set of certificates, private keys, and KRSP configuration files has been generated for a configuration with two KRA Servers. Refer to the instructions distributed with the CD-ROM for more details.

5.6 Creating a Master Configuration File

All of the KRS components rely on a master configuration file called KRS.CFG. The master configuration file is stored in the CFG subdirectory below the installation directory, which is chosen at install time. For example, if the installation directory were chosen to be C:\KRS, the absolute path of the master configuration file would be C:\KRS\CFG\KRS.CFG. An example of a master configuration file follows:

```
[KRO]
certprivkpair = c:\krs\samples\kro.cert, c:\krs\samples\kro.prvkey

[KRC]
certprivkpair = c:\krs\samples\krc.cert
netinfo = krc.company.com, 29003
```

Each section corresponds to a role within the key recovery phase. Each attribute for a given role can have one or more values separated by commas.

The valid section names are:

- [KRO] - For attributes related to the KRO Client
- [KRC name] - For attributes related to the named KRC Server. It's customary for KRC names to begin with the prefix "KRC_".
- [KRA name] - For attributes related to the named KRA. It's customary for KRA names to begin with the prefix "KRA_".

The format of valid attribute/value pairs is:

- certfile = FileContainingCertificate
- certprivkpair = FileContainingCertificate, FileContainingPrivateKey
- netinfo = IPAddress or Name, IPPort

The certfile value is the name of a file that contains an ASN/DER-encoded X.509 Version 3 public key certificate. The certprivkpair value is the name of two files: the file containing the certificate and the file containing the private key (optionally encrypted using *krutil.exe*). The netinfo value contains the TCP/IP address of a server followed by the port number that the server would use to listen for incoming requests.

5.7 Configuring a Key Recovery Officer Client

The KRO Client requires access to certificates and private keys specified in the master configuration file. The KRO Client needs this to contact the KRC and communicate with it securely. Set the following attribute/value pairs after copying the named files onto the system:

```
[KRO]
    certprivkpair = FileContainingCertificate, FileContainingPrivateKey
[KRC]
    certfile = FileContainingCertificate
```

In the enterprise and law enforcement scenarios, the KRO needs to contact the KRC using TCP/IP. Therefore, the following attribute/value pair needs to be added to the master configuration file:

```
[KRC]
    netinfo = NetworkAddress, NetworkPort
```

5.8 Configuring a Key Recovery Coordinator Server

The KRC Server requires the following attribute/value pairs in order for it to communicate securely and receive sealed information. The KRA information must be separated in sections that have the same name as the KRA. This name also will be needed at KRA startup time. All private keys that either be stored in the clear, or encrypted using the encryption utility *krutil.exe* distributed with the CD-ROM.

```
[KRC]
    certprivkpair = FileContainingCertificate, FileContainingPrivateKey

[KRA_1]
    certfile = FileContainingCertificate
    netinfo = NetworkAddress, NetworkPort

[KRA_2]
    certfile = FileContainingCertificate
    netinfo = NetworkAddress, NetworkPort
```

5.9 Configuring a Key Recovery Agent Server

The KRA Server requires the following attribute/value pairs in order for it to communicate securely and for it to compute secondary keys in the recovery sequence. The name of the section should be the name of the KRA itself.

```
[KRA_1]
    certprivkpair = FileContainingCertificate, FileContainingPrivateKey
```

Chapter 6. Operation

The operational aspects of each of the components of the Key Recovery Server (KRS) are discussed in detail in the following subsections.

6.1 Key Recovery Officer

The Key Recovery Officer (KRO) consists of one main application, the KRO application, which may be used to initiate a key recovery request operation. Three types of scenarios are used to perform a key recovery request operation: the Enterprise scenario, the Individual scenario and the Law Enforcement scenario. In the Enterprise and the Individual scenarios, the KRO has both the Key Recovery Block (KRB) and the credentials that authenticate it to receive the recovered key. This information will be transmitted over the network to the Key Recovery Coordinator (KRC). In the Law Enforcement scenario, all recovery requests are brought to the KRC administrator using a physical medium such as a floppy diskette.

The Key Recovery Officer application can run either interactively or in batch mode. The following sections describe in detail:

- The Graphical User Interfaces (GUIs) that enable the interactive mode of KRO operation, and
- The command line options which facilitate the batch mode operation.

6.1.1 Interactive Mode Operation of the Key Recovery Officer

This section describes the GUIs that facilitate the interactive operations of the KRO for the three scenarios.

6.1.1.1 Sending an Enterprise Key Recovery Request

The actual key recovery phase begins when the KRO administrator uses the KRO application to initiate a key recovery request to the appropriate KRC. If the KRO's private key is protected, the administrator must first enter the password that was used to in the encryption of the private key. Next , the administrator selects a KRB to be sent for recovery, enters the Authentication Information (AI) that will be used to authenticate the request to the KRC, and submits the request. The KRO administrator manually starts up the KRO application based on the input the administrator receives from an enterprise employee who needs to recover a key embedded within a KRB. It is assumed that the KRO administrator has the means to look up the AI value for the owner of the KRB. This AI value is then used, along with the KRB, to send a recovery request to the KRC.

To send an on-line key recovery request:

1. The KRO administrator receives a key recovery request from the enterprise in the form of a Key Recovery Block (where the KRB could be an e-mail attachment).

2. To start the KRO application, the KRO administrator clicks on the **Enterprise Request** icon from the KRO program group.

3. The Key Recovery Officer Private Key Password dialog appears (Figure 2). The KRO administrator enters the password used to wrap the KRO private key and clicks on the **Continue** button. If the private key is stored in the clear, the administrator clicks on **Continue**, leaving the edit box blank. If no password is entered, another dialog box appears (Figure 3), prompting as whether to assume that the private key is not encrypted. Click on the **Yes** button to continue, or **No** to go back to the KRO Private Key Password dialog. **Note:** This step is common to all scenarios.

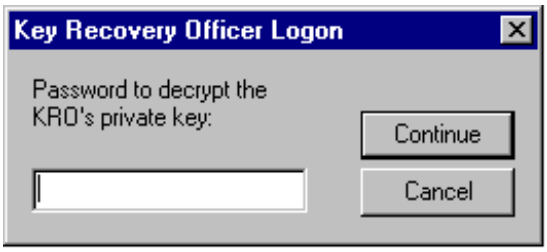

Figure 2. Key Recovery Officer Private Key Password

Figure 3. KRO Private Key Not Encrypted

4. The Key Recovery Officer Request Monitor screen then appears (Figure 4).

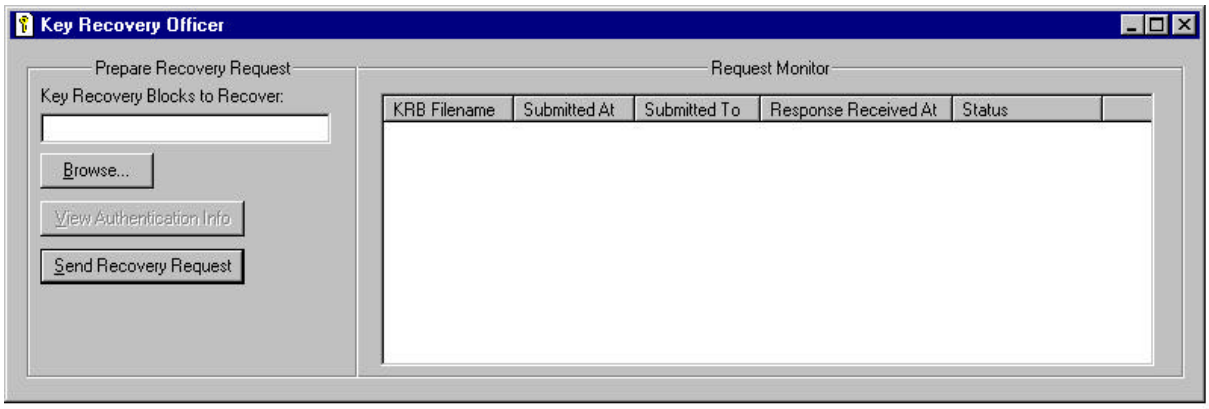

Figure 4. Key Recovery Officer Request Monitor

5. The KRO administrator clicks on the **Browse** button in the Key Recovery Officer Request Monitor screen to select the KRB file to be sent for recovery. The KRB Browse Directory appears (Figure 5).

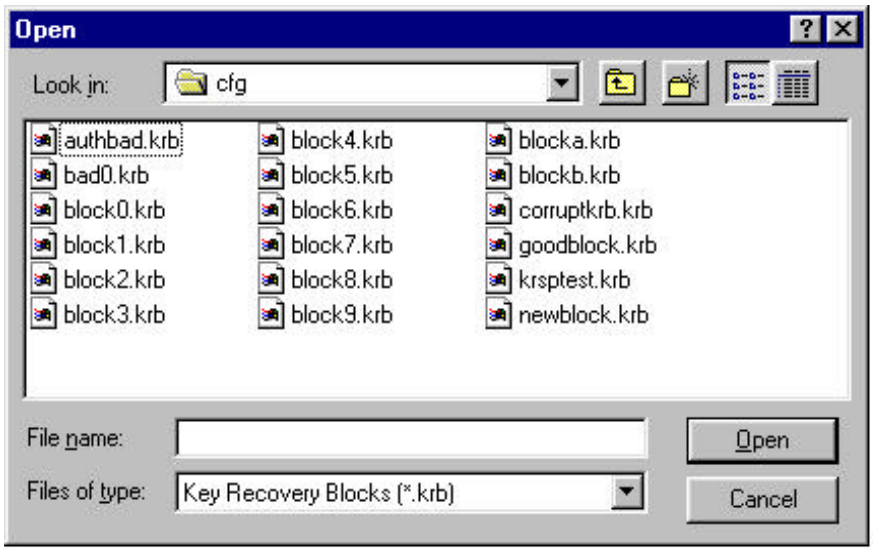

Figure 5. KRB Browse Directory

6. The KRO administrator selects the desired KRB files from the KRB Browse Directory, clicks on the **Open** button, and returns immediately to the Key Recovery Officer Request Monitor screen. The selected files will appear in the **Key Recovery Blocks to Recover** field.

7. The KRO administrator clicks on the **Send Recovery Request** button from the Key Recovery Officer Request Monitor screen. The KRB Authentication Password screen appears (Figure 6). The KRO administrator will be prompted to enter the Authentication Password associated with the selected KRB files. The KRO administrator clicks on the **Continue** button to send the key recovery requests to the KRC.

Figure 6. KRB Authentication Password

6.1.1.2 Sending an Individual Key Recovery Request

The detailed sequence of interactive operations for the Individual scenario is identical to that for the Enterprise scenario, except for the prompting for the appropriate type of the authentication password. The KRB Authentication Password dialog box in the case of the Individual Key Recovery Request will prompt for the Individual Key Recovery password, as required.

6.1.1.3 Sending a Law Enforcement Key Recovery Request

Sending a Law Enforcement key recovery request involves bypassing the KRO stage and moving directly to the KRC.

To send a manual law enforcement key recovery request:

1. The law enforcement agent must present the KRB and an RSA certificate on a 3.5-inch diskette to the KRC operator and submit the necessary credentials that are in the physical form of a legal warrant. This warrant will specify any information available to the law enforcement agencies which can be used to tie the warrant to the identity of the user for whom KRBs were generated (i.e., username, hostname, IP address).

2. The KRC operator checks the credentials of the law enforcement personnel and begins the recovery of the original encryption key from the KRB.

3. The KRC operator starts the key recovery operation by clicking on either the **Usage Jurisdiction or Manufacturing Jurisdiction Request** icons from the KRO program group. The Load Certificate screen appears (Figure 7), prompting the KRC operator to enter the certificate filename. The KRC operator can also click on the **Browse** button to select the desired certificate filename.

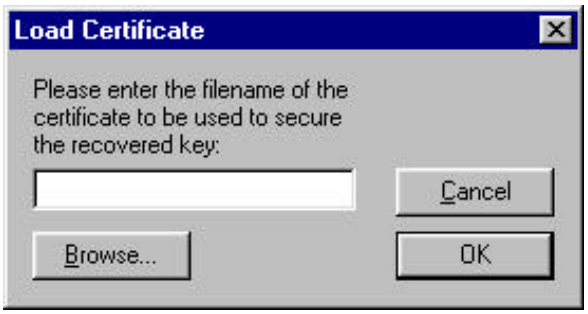

Figure 7. Load Certificate

4. The KRC operator clicks on the **OK** button from the Load Certificate screen. The Law Enforcement Key Recovery Officer Request Monitor screen appears (Figure 8).

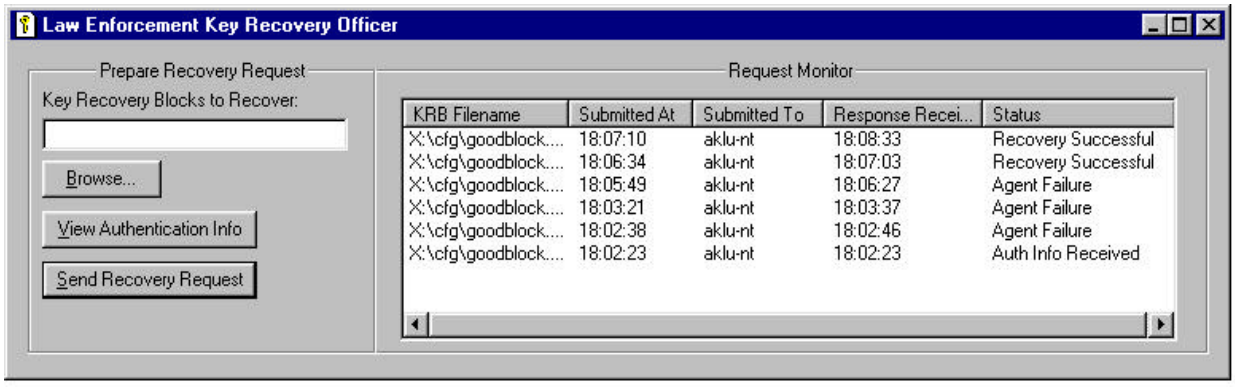

Figure 8. Law Enforcement Key Recovery Officer Request Monitor

5. The KRC operator clicks on the **Browse** button in the Law Enforcement Key Recovery Officer Request Monitor screen to select the KRB file to be sent for recovery. The KRB Browse Directory appears (see Figure 5).

6. The KRC operator clicks on the **View Authentication Info** button in the Law Enforcement Key Recovery Officer Request Monitor screen. The Law Enforcement Authentication Info screen appears (Figure 9), which displays the Authentication Information (AI) within the KRB. The KRC operator verifies the AI information with the warrant for validity. If the AI information is correct, the KRC operator clicks on the **OK** button and returns to the Law Enforcement Key Recovery Officer Request Monitor screen.

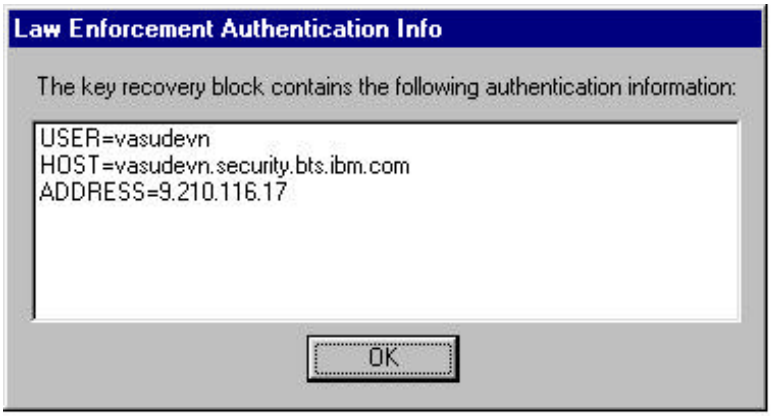

Figure 9. Law Enforcement Authentication Info

7. The KRC operator clicks on the **Send Recovery Request** button in the Law Enforcement Key Recovery Officer Request Monitor screen to initiate the key recovery request. If the key recovery process is successful, the Key Recovery Successful screen appears (Figure 10), which prompts the KRC operator to enter a filename. The KRC operator can also press the **Browse** button to save the recovered key to another directory. The KRC operator clicks on the **OK** button to save the recovered key to a file, and returns immediately to the Law Enforcement Key Recovery Officer Request Monitor screen. The KRC operator will be able to start another key recovery request process from this screen.

Figure 10. Key Recovery Successful

6.1.1.4 Monitoring the Status of Requests

The Key Recovery Officer application provides a request monitoring facility (see Figure 4) that allows the KRO administrator to monitor the status of all recovery requests that are being serviced or have been serviced at any point in time.

The following status information is displayed in the Key Recovery Officer Request Monitor screen (see Figure 4):

- KRB Filename - Identifies the filename of the KRB sent by the KRO.
- Submitted At - Identifies the time of day the key recovery request was sent from the KRO. The time is shown in military format (hh/mm/ss).
- Submitted To - Identifies the IP address of the KRC.
- Response Received At - Identifies the time of day of when the response comes back from the KRC.
- Status - Identifies the current status of the recovery request (e.g., recovery successful, request aborted, server failure, etc).

6.1.2 Batch Mode Operation of the Key Recovery Officer

In the batch mode, the KRO application can be invoked from a DOS box or from the Start->Run menu. The following options are supported by the KRO batch command:

-p password	Allows the enterprise/individual authentication password to be specified.
-privk private key password	Allows the private key password to be specified.
-c LE certificate	Replaces the prompting for certificate.
-a	Get authentication information only.
-b	Blocking call.
-n loopcount	Repeat sending Key Recovery Blocks [krb …].
-t time-out	Will time out in xxx milliseconds, if response is not received.
[krb …]	Filenames for KRB to be recovered.

6.1.3 Decrypting the Original Data

Once the key recovery request has been completed, and the recovered key returned to KRO, the encryption utility can then be used to decrypt the original data. (**Note:** In the law enforcement scenario, the law enforcement agent must decrypt the recovered key using his private key before the recovered key can be used in the encryption utility.) This can be done by selecting the encrypted file in the **Files** field, clicking on the **Use File as Key** button, and selecting the recovered key file in the **Key File** field. The utility will automatically determine the algorithm and mode required to decrypt the data from the contents of the key file. The user will be prompted for a filename with which to save the original clear data. It will be according to particular enterprise policies whether the decrypted data is recovered at the site of the KRO, or at the original client's desktop where the file was encrypted.

6.1.4 Key Recovery Officer Library Module (KROAPI.DLL)

The Key Recovery Officer Library Module defines an API for secure and convenient implementation of key recovery services on the client side. The API hides the underlying protocol needed to communicate with the KRC, as well as many cryptographic operations required to ensure the integrity and privacy of the key recovery requests and replies. This library also frees the application developer to implement the given enterprise policies with respect to key recovery (e.g., checking the credentials of the requestor versus those

of the KRB's owner). Finally, any required privileged access to the Common Data Security Architecture (CDSA) framework also will be encapsulated by this library module.

Operations defined in the APIs include key recovery and retrieval of authentication information embedded in the KRB, in addition to other administrative functions. Since key recovery operations usually require a sequence of interactions among many applications, that are compliant with a KRS messaging protocol, this API provides non-blocking calls coupled with the flexibility of callbacks for tailored dispositions of recovered keys and related information. As a trade-off, however, the application will have the responsibility for setting the appropriate time-out value and controlling the number of key recovery operations that can be performed concurrently.

6.1.4.1 IBM KROAPI Library, Version 1.0

Files required:

- kroapi.dll
- kroapi.h

Additional files:

- Common Security Services Manager (CSSM) framework and all supporting include files
- ibmcl.dll
- ibmswcsp.dll
- ossapi.dll
- ossdmem.dll
- cstrain.dll
- soedapi.dll
- soedber.dll

The following KROAPI functions are provided in Table 2.

Table 2. KROAPI Functions

Function Name	Comments
KRO_Init	See remarks below
KRO_RecoverKey	See remarks below
KRO_AbortRequest	See remarks below
KRO_GetAuthInfo	See remarks below

Remarks:

KRO_Init

An application is expected to invoke this function prior to any other functions in the library. The private key password can also be specified with this function. If the private key is stored in the clear, a NULL pointer can be passed. This function expects the registry string "HKEY_LOCAL_MACHINE\\SOFTWARE\\IBM\\KRS\\CFGDIR" to point to the proper key recovery configuration file. (**Note:** The framework (CSSM) must have already been initialized via a call to CSSM_Init() prior to KRO_Init being invoked.)

KRO_RecoverKey

This function is non-blocking by default. The *credentials*, *krc_name*, *p_KeyProtectKey*, and *callback* parameters can all be set to NULL.

KRO_GetAuthInfo

This function is non-blocking by default. The *krc_name* and *callback* parameter can all be set to NULL.

KRO_AbortRequest

This function will block. The *rhandle* parameter must have been previously returned from a call to KRO_GetAuthInfo or KRO_RecoverKey. This function is applicable only when the previous two functions (KRO_RecoverKey, KRO_GetAuthInfo) were invoked in non-blocking mode.

6.2 Key Recovery Coordinator

The Key Recovery Coordinator (KRC) consists of one main application, the KRC application, which when started, behaves as a daemon or server process that continuously listens for incoming requests, until the process is terminated either explicitly or at system shutdown.

The KRC application uses several files to perform its operation. These files are accessible through utilities that are available on the KRC system. The KRC database file holds entries that contain certificates and network information for KRA entities that are known to the KRC. The KRC database file also contains information about the KRC's own certificates and the names of the files that contain the corresponding private keys.

6.2.1 Starting the Server

The KRC application can be started using two different methods: the application is configured to start as part of system services, or the KRC operator can start up the KRC application manually.

To start the KRC application using system services:

1. The KRC operator copies the **KRC** icon into the start-up window in the program group, where the server will run automatically whenever the computer is booted up.

2. Whenever the KRC application is started up in interactive mode (default), the Key Recovery Coordinator Name screen appears (Figure 11) prompting for the KRC Name and the password used to wrap the KRC's private key. Click on the **OK** button if "KRC" is the name in the configuration file to be used, and if the private key is stored in the clear. Again, click on the **OK** button when prompted as whether to assume that the private key is not encrypted. Otherwise, type in the desired KRC name from the configuration file and the appropriate password. The **KRC** icon will appear in the status area of the taskbar where it will continuously run in the background.

To start the KRC server manually:

1. The KRC operator has the option of starting the application by accessing the **KRC application** icon (Key Recovery Coordinator) using the Windows NT startup menu.

2. The **KRC** icon will appear in the status area of the taskbar where it continuously runs in the background.

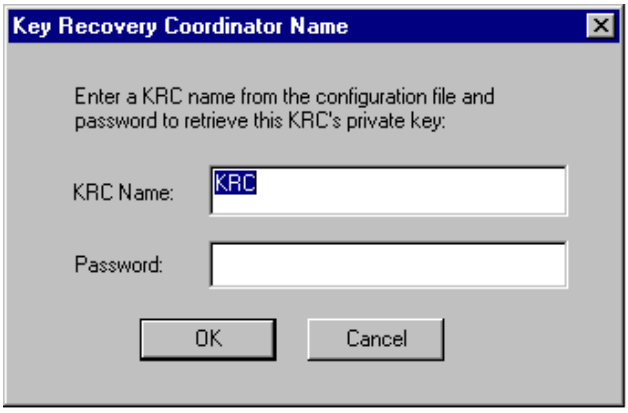

Figure 11. Key Recovery Coordinator Name

6.2.2 Monitoring the Status of Requests

The KRC application provides a request monitoring facility that allows the KRO administrator to monitor the status of all recovery requests that are being serviced or have been serviced at any point in time. The following status information is displayed in the Key Recovery Coordinator Request Monitor screen (Figure 12):

- Session Id - Provides a unique numeric value identifying a single key recovery request.
- Request Received At - Identifies the time of day a key recovery request was received from a KRO.
- Originator - Identifies the IP address of the KRO who initiated the key recovery request.
- Number of KRAs - Identifies the number of KRAs involved in recovering the key from the KRB.
- Responses From KRAs - Identifies the number of KRAs who have responded to the KRC.
- Response Posted At - Identifies the time of day that a response was posted to the originating KRO.
- Status - Identifies the current status of the recovery request (e.g., recovery successful, request aborted, KRC failure, etc).

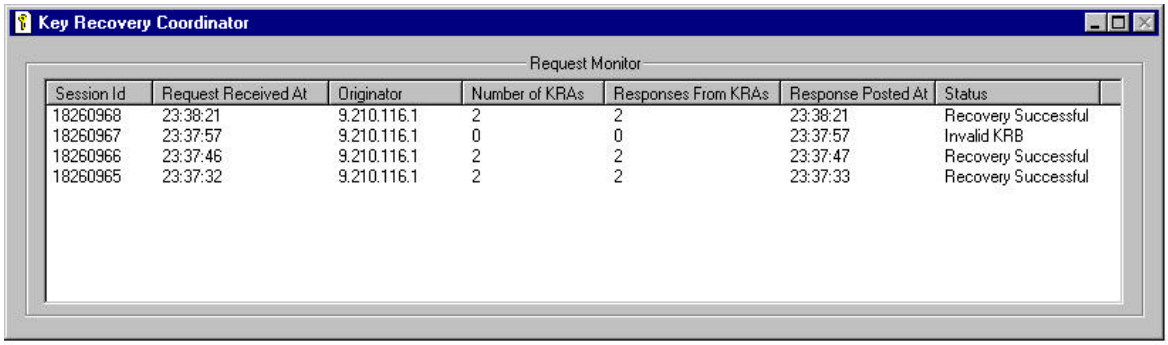

Figure 12. Key Recovery Coordinator Request Monitor

6.2.3 Viewing the Audit Trail

The KRC operator is responsible for viewing the audit trail, which is available under the Microsoft Windows NT Event Viewer administrative tool. The audit trail logs the events that occur during the key recovery operation. These events include the following:

- Successful recoveries
- Communication between key recovery entities
- Authentication failures
- Cryptographic failures
- Network errors
- Server startup and shutdown

6.3 Key Recovery Agent

The Key Recovery Agent (KRA) consists of one main application, the KRA application, which when started behaves as a daemon or server process that continuously listens for incoming requests, until the process is terminated either explicitly or at system shutdown. The system, which serves as the KRA, may be configured to start the KRA application as part of system services; alternatively, the KRA operator can start up the KRA application manually.

The KRA application uses several files to perform its operation. The KRA database file holds entries that contain information about the KRA's own certificates and the names of the files that contain the corresponding private keys.

6.3.1 Starting the Server

The KRA application can be started using two different methods: the application is configured to start as part of system services, or the KRA operator can start up the KRA application manually.

To start the KRA application using system services:

1. The KRA operator copies the **KRA** icon into the start-up window in the program group, where the server will run automatically whenever the computer is booted up.

2. Whenever the KRA application is started up in interactive mode (default), the Key Recovery Agent Name screen appears (Figure 13) prompting for the KRA Name and the password used to wrap the KRA private key. If no password is entered, a dialog box appears, prompting as whether to assume that the private key is not encrypted. Click on the **Yes** button to continue, or **No** to go back to the Key Recovery Agent Name screen. The **KRA** icon will appear in the status area of the taskbar where it continuously runs in the background.

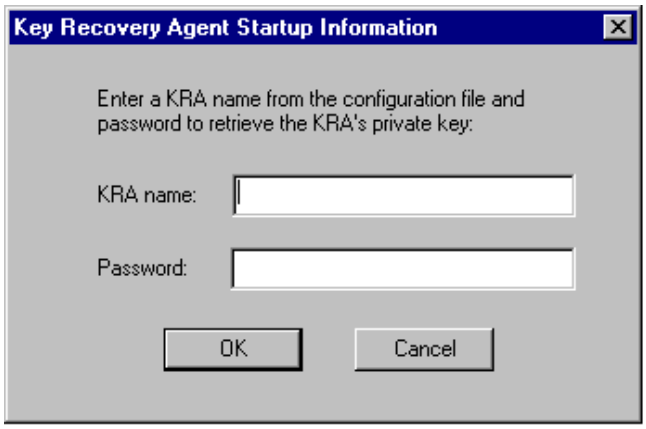

Figure 13. Key Recovery Agent Name

To start the KRA server manually:

1. The KRA operator has the option of starting the application by accessing the **KRA application** icon using the Windows NT startup menu.

2. Whenever the KRA application is started up, the Key Recovery Agent Name screen appears (see Figure 13) prompting for the KRA Name. The **KRA** icon will appear in the status area of the taskbar where it continuously runs in the background.

6.3.2 Monitoring the Status of Requests

The KRA application provides a request monitoring facility that allows the KRA operator to monitor the status of all recovery requests that are being serviced or have been serviced at any point in time. The following status information is displayed in the Key Recovery Agent Request Monitor screen (Figure 14):

- Session Id - Provides a unique numeric value identifying a single key recovery request.
- Received At - Identifies the time of day when a KK (secondary key) request was received from the KRC.
- Originator - Identifies the IP address of the KRO who initiated the key recovery request.
- Response Submitted At - Identifies the time of day the KRA responded to the KRC's request.
- Status - Identifies the current status of the recovery request (e.g., recovery successful, request aborted, KRA failure, etc).

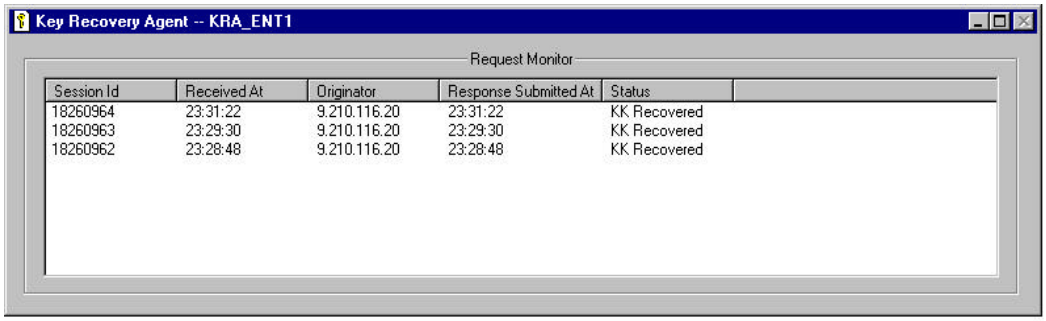

Figure 14. Key Recovery Agent Request Monitor

Appendix A. Pre-Installation Setup

The Key Recovery Server (KRS) component facilities are high-value entities and are thus prime targets for attack. Therefore, elaborate measures need to be taken to ensure that these facilities operate within a secure environment that is highly resistant to penetration and compromise. In addition, the KRC, KRA, and KRO needs to hold, maintain, and use one or more RSA key pairs to perform their operations. These key pairs constitute very sensitive information and need to be maintained and used in a manner that does not undermine the confidentiality of the private keys. This appendix provides guidelines to prepare a secure facility and to load a machine with a fresh copy of Windows NT 4.0 before the KRS software is installed.

A.1 Facility Requirements

This section outlines the physical and administrative security procedures that should be followed at the Key Recovery Server Facility (KRSF). These procedures include the following:

- Key Recovery Server Facility Room
- Limited Personnel Access

A.1.1 Key Recovery Server Facility Room

The KRSF should be a room with a single entrance/exit whose doorway is secured by a combination lock. The room should have no windows and a secured ceiling.

A.1.2 Limited Personnel Access

Access to the facility must be restricted to a limited number of personnel. At all times, a two-person rule should be imposed for access to the KRSF. The combination keyed lock for the KRSF should be split between two designated security officers. Both security officers must be present in order to open the KRSF door.

A.2 Server Physical Requirements

The system requirements to be followed in the KRSF include the following:

- Keyed lock for system
- Dedicated system
- Removable backup media connectivity
- Microsoft Windows NT 4.0

A.2.1 Key Locked for System

The system should have its own keyed lock to ensure no tampering.

A.2.2 Dedicated System

The KRSF should house a dedicated system that runs only the IBM Key Recovery Server.

A.2.3 Removable Backup Media Connectivity

The system should be connected to a removable backup media device to ensure timely backup source for database redundancy.

A.2.4 Microsoft Windows NT 4.0

The KRSF should use a secure platform (NT Workstation 4.0) which supports identification and authentication as well as filesystem level access control. A single nonadministrative account on this system should be set up. The account password should be broken up into two portions and each portion is revealed to one of the two security officers. The goal is to make sure that both security officers are present to log into the KRSF system and perform KRSF operations.

There should be at least 10 characters in the password so that the password can be split between dual entities for access to the system. Passwords must be changed frequently to ensure higher level security. Account lockout should occur after a small number of login attempts, and should stay locked until the security administrator unlocks the account.

The Windows NT 4.0 auditing policies will provide information relevant to the following events:

- User logs on in order to change password
- All logon and logoff to conduct recovery operations
- Changes to user rights
- User and group management rights changes
- Security policy changes

A.3 Installation

Once the above steps are completed to prepare the system, the installation of the key recovery system may proceed. Steps described on the CD should be followed for the complete installation.

The process described above is a recommendation based on IBM experience in using this product. IBM DISCLAIMS ALL WARRANTIES EXPRESS AND IMPLIED WITH RESPECT TO THE USE OF THESE RECOMMENDATIONS INCLUDING (WITHOUT LIMITATION) ANY WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Appendix B. Key Recovery Officer API

RECOVERY_REQUEST_HANDLE KRO_RecoverKey (
KRO_MODE kroMode,
const char* krb_fname,
CSSM_DATA_PTR credentials,
const char *krc_name,
uint32 timeout,
const CSSM_KEY_PTR pKeyProtectKey,
KEY_RECOVERY_CALLBACK callback);

Non-blocking call supporting key recovery for all scenarios, namely enterprise, law enforcement, and individual.

Parameters

kroMode (input)
Enumerated value indicating the type of key recovery (e.g., enterprise or law enforcement).

krb_fname (input)
Name of file containing the KRB.

credentials (input/optional)
Pointer to byte string conveying authentication information to the KRC. Its interpretation depends on the key recovery scenario.

krc_name (input/optional)
Name of the KRC in the configuration file to be used.

timeout (input)
Number of milliseconds to wait before the callback function is called with the timeout response code.

nflags (input)
Specifies the way in which the call is made. Various options can be ORed together.

pKeyProtectKey (input/optional)
Pointer to wrap the recovered key.

callback (input/optional)
Pointer to user function to be called for every change in the KRO recovery states. This function is supplied by the user to specify the final disposition of the recovered key. If not specified, a default callback function is used to save the recovered key and/or display the final recovery status.

callback_param (input/optional)
Pointer to user-defined parameters to the callback function.

Return Value

Returns a handle identifying the key recovery thread. If the handle is INVALID_RECOVERY_HANDLE, the operation was not successfully started.

RECOVERY_REQUEST_HANDLE_KRO_GetAuthInfo (
KRO_MODE kroMode,
const char *krb_fname,
const char *krc_name,
unit32 timeout,
KEY_RECOVERY_CALLBACK callback);

Non-blocking call supporting authentication information retrieval for law enforcement key recovery.

Parameters

kroMode (input)
Enumerated value indicating the type of key recovery. Currently, only LE_USE and LE_MAN are supported.

krb_fname (input)
Name of file containing the KRB.

krc_name (input/optional)
Name of the KRC in the configuration file to be used.

nflags (input)
Specifies the way in which the call is made. Various options can be ORed together.

timeout (input)
Number of milliseconds to wait before the callback function is called with the timeout response code.

callback (input/optional)
Pointer to user function to be called for every change in the KRO retrieval states. This function is supplied by the user to display the authentication information embedded in the specified KRB. If not specified, a default callback function is used to display the authentication information.

callback_param (input/optional)
Pointer to user-defined parameters to the callback function.

Return Value

Returns a handle identifying this authentication information thread. If the handle is INVALID_RECOVERY_HANDLE, the operation was not successfully started.

sint32 KRO_AbortRequest (RECOVERY_REQUEST_HANDLE rHandle);

Function supporting the cancellation of key recovery or authentication information retrieval operations.

Parameters

rHandle (input)
Handle identifying the key recovery or authentication information retrieval operation to be cancelled. This handle must have been previously returned by a call to KRO_RecoverKey or KRO_GetAuthInfo.

Return Value

Returns KRO_OK if the specified operation was successfully cancelled. If the returned value is –1, either the operation has already finished or the rHandle was not valid.

CSSM_RETURN KRO_Init (const char *password);

Function to read the KRO's configuration file and to load the appropriate cryptographic modules. Must be invoked prior to other kroapi calls.

Parameters

password (input/optional)
Null-terminated string used to unwrap the KRO’s private key.

char *KRO_errstr (unit32 error_code);

Function to map returned KRO error codes to error messages.

Parameters

error_code (input)
Error made available to the callback function.

Return Value

Returns corresponding error messages.

Appendix C. List of Acronyms

AI	Authentication Information
API	Application Programming Interface
CA	Certificate Authority
CDSA	Common Data Security Architecture
CSSM	Common Security Services Manager
GUI	Graphical User Interface
KRA	Key Recovery Agent
KRB	Key Recovery Block
KRC	Key Recovery Coordinator
KRF	Key Recovery Field
KRO	Key Recovery Officer
KRS	Key Recovery Server
KRSF	Key Recovery Server Facility
KRSP	Key Recovery Service Provider
PKI	Public Key Infrastructure
TP	Trust Policy

Appendix D. Glossary

Asymmetric algorithms	Cryptographic algorithms, where one key is used to encrypt and a second key is used to decrypt. They are often called public-key algorithms. One key is called the public key, and the other is called the private key or secret key. RSA (Rivest-Shamir-Adelman) is the most commonly used public-key algorithm. It can be used for encryption and for signing.
Authentication Information	Information that is verified for authentication. For example, a Key Recovery Officer (KRO) selects a password which will be used for authentication with the Key Recovery Coordinator (KRC). A KRO operator who has identification information must search the Authentication Information (AI) database to locate an AI value that corresponds to the individual who generated the information.
Certificate	See Digital certificate.
Certificate Authority	An entity that guarantees or sponsors a certificate. For example, a credit card company signs a cardholder's certificate to ensure that the cardholder is who he or she claims to be. The credit card company is a Certificate Authority (CA). CAs issue, verify, and revoke certificates.
Certificate chain	The hierarchical chain of all the other certificates used to sign the current certificate. This includes the CA who signs the certificate, the CA who signed that CA's certificate, and so on. There is no limit to the depth of the certificate chain.
Certificate signing	The CA can sign certificates it issues or co-sign certificates issued by another CA. In a general signing model, an object signs an arbitrary set of one or more objects. Hence, any number of signers can attest to an arbitrary set of objects. The arbitrary objects could be, for example, pieces of a document for libraries of executable code.
Certificate validity date	A start date and a stop date for the validity of the certificate. If a certificate expires, the CA may issue a new certificate.
Cryptographic algorithm	A method or defined mathematical process for implementing a cryptography operation. A cryptographic algorithm may specify the procedure for encrypting and decrypting a byte stream, digitally signing an object, computing the hash of an object, generating a random number, etc. IBM KeyWorks accommodates Data Encryption Standard (DES), RC2, RC4, International Data Encryption Algorithm (IDEA), and other encryption algorithms.
Cryptographic Service Provider	Cryptographic Service Providers (CSPs) are modules that provide secure key storage and cryptographic functions. The modules may be software only or hardware with software drivers. The cryptographic functions provided may include: • Bulk encryption and decryption • Digital signing • Cryptographic hash

- Random number generation
- Key exchange

Cryptography

The science for keeping data secure. Cryptography provides the ability to store information or to communicate between parties in such a way that prevents other non-involved parties from understanding the stored information or accessing and understanding the communication. The encryption process takes understandable text and transforms it into an unintelligible piece of data (called ciphertext); the decryption process restores the understandable text from the unintelligible data. Both involve a mathematical formula or algorithm and a secret sequence of a data called a key. Cryptographic services provide confidentiality (keeping data secret), integrity)preventing data from being modified), authentication (proving the identity of a resource or a user), and non-repudiation (providing proof that a message or transaction was sent and/or received).

There are two type of cryptography:

- In shared/secret key (symmetric) cryptography there is only one key that is a shared secret between the two communicating parties. The same key is used for encryption and decryption.
- In public key (asymmetric) cryptography different keys are used for encryption and decryption. A party has two keys: a public key and a private key. The two keys are mathematically related, but it is virtually impossible to derive the private key from the public key. A message this is encrypted with someone's public key (obtained from some public directory) can only be decrypted with the associated private key. Alternately, the private key can be used to "sign" a document; the public key can be used as verification of the source of the document.

Cryptoki

Short for cryptographic token interface. See Token.

Digital certificate

The binding of some identification to a public key in a particular domain, as attested to directly or indirectly by the digital signature of the owner of that domain. A digital certificate is an unforgettable credential in cyberspace. The certificate is issued by a trusted authority, covered by that party's digital signature. The certificate may attest to the certificate holder's identity, or may authorize certain actions by the certificate holder. A certificate may include multiple signatures and may attest to multiple objects or multiple actions.

Digital signature

A data block that was created by applying a cryptographic signing algorithm to some other data using a secret key. Digital signatures may be used to:

- Authenticate the source of a message, data, or document
- Verify that the contents of a message has not been modified since it was signed by the sender
- Verify that a public key belongs to a particular person

Typical digital signing algorithms include MD5 with RSA encryption, and DSS, the proposed Digital Signature Standard defined as part of the U.S. Government Capstone project.

Enterprise	A company or individual who is authorized to submit on-line requests to the Key Recovery Officer (KRO). In the enterprise key recovery scenario, a process at the enterprise called the KRO is responsible for preparing key recovery requests and communicating them to the KRC. The KRO, acting on behalf of an enterprise or individual, sends an on-line request to the Key Recovery Coordinator (KRC) to recover a key from a Key Recovery Block (KRB).
Graphical User Interface	A type of display format that enables the user to choose commands, start programs, and see lists of files and other options by pointing to pictorial representations (icons) and lists of menu items on the screen. Graphical User Interfaces (GUIs) are used by the Microsoft Windows program for IBM-compatible microcomputers and by other systems.
Hash algorithm	A cryptographic algorithm used to hash a variable-size input stream into a unique, fixed-sized output value. Hashing is typically used in digital signing algorithms. Example hash algorithms include MD and MD2 from RSA Data Security. MD5, also from RSA Data Security, hashes a variable-size input stream into a 128-bit output value. SHA, a Secure Hash Algorithm published by the U.S. Government, produces a 160-bit hash value from a variable-size input stream.
IBM KeyWorks Architecture	A set of layered security services that address communications and data security problems in the emerging PC business space.
IBM KeyWorks Framework	The IBM KeyWorks Framework defines five key service components: • Cryptographic Module Manager • Key Recovery Module Manager • Trust Policy Module Manager • Certificate Library Module Manager • Data Storage Library Module Manager KeyWorks binds together all the security services required by PC applications. In particular, it facilitates linking digital certificates to cryptographic actions and trust protocols.
Key Escrow	The storing of a key (or parts of a key) with a trusted party or trusted parties in case of loss or destruction of the key.
Key Recovery Agent	The Key Recovery Agent (KRA) acts as the back end for a key recovery operation. The KRA can only be accessed through an on-line communication protocol via the Key Recovery Coordinator (KRC). KRAs are considered outside parties involved in the key recovery process; they are analogous to the neighbors who each hold one digit of the combination of the lock box containing the key. The authorized parties (i.e., enterprise or law enforcement) have the freedom to choose the number of specific KRAs that they want to use. The authorized party requests that each KRA decrypt its section of the Key Recovery Fields (KRFs) that is associated with the transmission. Then those pieces of information are used in the process that derives the session key. The KRA will only be able to recover a portion of the key, and reading the original message will require searching the remaining key space in order to find the key that will decrypt the message. The number of KRAs on each end of the communication does not have to be equal.

Key Recovery Block

The Key Recovery Block (KRB) is a piece of encrypted information that is contained within a block. The KRS components (i.e., KRO, KRC, KRA) work collectively to recover a session key from a provided KRB. In the enterprise scenario, the KRO has both the KRB and the credentials that authenticate it to receive the recovered key. This information will be transmitted over the network to the KRC. In the law enforcement scenario, the KRB is presented on a 3.5-inch diskette, and the credentials are in the physical form of a legal warrant. This warrant will specify any information available to the law enforcement agents which can be used to tie the warrant to the identity of the user for whom KRBs were generated (i.e., username, hostname, IP address). The KRC has the ability to check credentials and derive the original encryption key from the KRB with the help of its KRAs.

Key Recovery Coordinator

The Key Recovery Coordinator (KRC) acts as the front end for the key recovery operation. The KRO, acting on behalf of an enterprise or individual, sends an on-line request to the KRC to recover a key from a KRB. The KRC receives the on-line request and services it by interacting with the appropriate set of KRAs as specified within the KRB. The recovered key is then sent back to the KRO by the KRC using an on-line protocol. The KRC consists of one main application which, when started, behaves as a server process. The system, which serves as the KRC, may be configured to start the KRC application as part of system services; alternatively, the KRC operator can start up the KRC application manually. The KRC application performs the following operations:

- Listens for on-line recovery requests from KRO
- Can be used to launch an embedded application that allows manual key recovery for law enforcement
- Monitors and displays the status of the recovery requests being serviced

Key Recovery Field

A Key Recovery Field (KRF) is a block of data that is created from a symmetric key and key recovery profile information. The Key Recovery Service Provider (KRSP) is invoked from the IBM KeyWorks framework to create the KRFs. There are two major pieces of the key recovery fields: block 1 contains information that is unrelated to the session key of the transmitted message, and encrypted with the public keys of the selected key recovery agents; block 2 contains information that is related to the session key of the transmission. The KRSP generates the KRFs for the session key. This information is *not* the key or any portion of the key, but is information that can be used to recover the key. The KRSP has access to location-unique jurisdiction policy information that controls and modifies some of the steps in the generation of the KRFs. Only once the KRFs are generated, and both the client and server sides have access to them, can the encrypted message flow begin. KRFs are generated so that they can be used either by a KRA to recover the original symmetric key, because the user who generated the message has lost the key, or at the warranted request of law enforcement agents.

Key Recovery Module Manager	The Key Recovery Module Manager enables key recovery for cryptographic services obtained through the IBM KeyWorks. It mediates all cryptographic services provided by the IBM KeyWorks and applies the appropriate key recovery policy on all such operations. The Key Recovery Module Manager contains a Key Recovery Policy Table (KRPT) that defines the applicable key recovery policy for all cryptographic products. The Key Recovery Module Manager routes the KR-API function calls made by an application to the appropriate KR-SPI functions. The Key Recovery Module Manager also enforces the key recovery policy on all cryptographic operations that are obtained through IBM KeyWorks. It maintains key recovery state in the form of key recovery contexts.
Key Recovery Officer	An entity called the Key Recovery Officer (KRO) is the focal point of the key recovery process. In the enterprise key recovery scenario, the KRO is responsible for preparing key recovery requests and communicating them to the KRC. The KRO has both the KRB and the credentials that authenticate it to receive the recovered key. The KRO is the entity that acts on behalf of an enterprise to initiate a key recovery request operation. An employee within an enterprise who desires key recovery will send a request to the KRO with the KRB that is to be recovered. The actual key recovery phase begins when the KRO operator uses the KRO application to initiate a key recovery request to the appropriate KRC. At this time, the operator selects a KRB to be sent for recovery, enters the Authentication Information (AI) that can be used to authenticate the request to the KRC, and submits the request.
Key Recovery Policy	Key recovery policies are mandatory policies that are typically derived from jurisdiction-based regulations on the use of cryptographic products for data confidentiality. Often, the jurisdictions for key recovery policies coincide with the political boundaries of countries in order to serve the law enforcement and intelligence needs of these political jurisdictions. Political jurisdictions may choose to define key recovery policies for cryptographic products based on export, import, or use controls. Enterprises may define internal and external jurisdictions, and may mandate key recovery policies on the cryptographic products within their own jurisdictions. Key recovery policies come in two flavors: *key recovery enablement policies* and *key recovery interoperability policies.* Key recovery enablement policies specify the exact cryptographic protocol suites (e.g., algorithms, modes, key lengths, etc.) and perhaps usage scenarios, where key recovery enablement is mandated. Furthermore, these policies may also define the number of bits of the cryptographic key that may be left out of the key recovery enablement operation; this is typically referred to as the *workfactor*. Key recovery interoperability policies specify to what degree a key recovery enabled cryptographic product is allowed to interoperate with other cryptographic products.
Key Recovery Server	The Key Recovery Server (KRS) consists of three major entities: Key Recovery Coordinator (KRC), Key Recovery Agent (KRA), and Key Recovery Officer (KRO). The KRS is intended to be used by enterprise employees and security personnel, law enforcement personnel, and KRSF personnel. The KRS interacts with one or more local or remote KRAs to reconstruct the secret key that can be used to decrypt the ciphertext.

Key Recovery Server Facility	The Key Recovery Server Facility (KRSF) is a facility room that houses the KRS component facilities, ensuring they operate within a secure environment that is highly resistant to penetration and compromise. Several physical and administrative security procedures must be followed at the KRSF such as a combination keyed lock, limited personnel, standalone system, operating system with security features (Microsoft NT Workstation 4.0), NTFS (Windows NT Filesystem), and account and auditing policies.
Key Recovery Service Provider	Key Recovery Service Providers (KRSPs) are modules that provide key recovery enablement functions. The cryptographic functions provided may include: • Key recovery field generation • Key recovery field processing
Law Enforcement	A type of scenario where key recovery is mandated by the jurisdictional law enforcement authorities in the interest of national security and law enforcement. In the law enforcement scenario, the KRB is presented on a 3.5-inch diskette, and the credentials are in the physical form of a legal warrant. This warrant will specify any information available to the law enforcement agents which can be used to tie the warrant to the identity of the user for whom KRBs were generated (i.e., username, hostname, IP address).
Leaf certificate	The certificate in a certificate chain that has not been used to sign another certificate in that chain. The leaf certificate is signed directly or transitively by all other certificates in the chain.
Message digest	The digital fingerprint of an input stream. A cryptographic hash function is applied to an input message arbitrary length and returns a fixed-size output, which is called the digest value.
Owned certificate	A certificate whose associated secret or private key resides in a local Cryptographic Service Provider (CSP). Digital-signing algorithms require using owned certificates when signing data for purposes of authentication and non-repudiation. A system may use certificates it does not own for purposes other than signing.
Private key	The cryptographic key is used to decipher messages in public-key cryptography. This key is kept secret by its owner.
Public key	The cryptographic key is used to encrypt messages in public-key cryptography. The public key is available to multiple users (i.e., the public).
Random number generator	A function that generates cryptographically strong random numbers that cannot be easily guessed by an attacker. Random numbers are often used to generate session keys.

Root certificate	The prime certificate, such as the official certificate of a corporation or government entity. The root certificate is positioned at the top of the certificate hierarchy in its domain, and it guarantees the other certificates in its certificate chain. Each Certificate Authority (CA) has a self-signed root certificate. The root certificate's public key is the foundation of signature verification in its domain.
Secure Electronic Transaction	A mechanism for securely and automatically routing payment information among users, merchants, and their banks. Secure Electronic Transaction (SET) is a protocol for securing bankcard transactions on the Internet or other open networks using cryptographic services. SET is a specification designed to utilize technology for authenticating parties involved in payment card purchases on any type of on-line network, including the Internet. SET was developed by Visa and MasterCard, with participation from leading technology companies, including Microsoft, IBM, Netscape, SAIC, GTE, RSA, Terisa Systems, and VeriSign. By using sophisticated cryptographic techniques, SET will make cyberspace a safer place for conducting business and is expected to boost consumer confidence in electronic commerce. SET focuses on maintaining confidentiality of information, ensuring message integrity, and authenticating the parties involved in a transaction. The significance of SET, over existing Internet security protocols, is found in the use of digital certificates. Digital certificates will be used to authenticate all the parties involved in a transaction. SET will provide those in the virtual world with the same level of trust and confidence a consumer has today when making a purchase at any of the 13 million Visa-acceptance locations in the physical world. The SET specification is open and free to anyone who wishes to use it to develop SET-compliant software for buying or selling in cyberspace.
Security Context	A control structure that retains state information shared between CSP and the application agent requesting service from the CSP. Only one context can be active for an application at any given time, but the application is free to switch among contexts at will, or as required. A security context specifies CSP and application-specific values, such as required key length and desired hash functions.
Security-relevant event	An event where a CSP-provided function is performed, a security module is loaded, or a breach of system security is detected.
Session key	A cryptographic key used to encrypt and decrypt data. The key is shared by two or more communicating parties, who use the key to ensure privacy of the exchanged data.
Signature	See Digital signature.
Signature chain	The hierarchical chain of signers, from the root certificate to the leaf certificate, in a certificate chain.
Smart Card	A device (usually similar in size to a credit card) that contains an embedded microprocessor that could be used to store information. Smart cards can store credentials used to authenticate the holder.

S/MIME	Secure/Multipurpose Internet Mail Extensions (S/MIME) is a protocol that adds digital signatures and encryption to Internet MIME messages. MIME is the official proposed standard format for extended Internet electronic mail. Internet e-mail messages consist of two parts, the header and the body. The header forms a collection of field/value pairs structured to provide information essential for the transmission of the message. The body is normally unstructured unless the e-mail is in MIME format. MIME defines how the body of an e-mail message is structured. The MIME format permits e-mail to include enhanced text, graphics, audio, and more in a standardized manner via MIME-compliant mail systems. However, MIME itself does not provide any security services. The purpose of MIME is to define such services, following the syntax given in PKCS#7 for digital signatures and encryption. The MIME body part carries a PKCS #7 message, which itself is the result of cryptographic processing on other MIME body parts.
Symmetric algorithms	Cryptographic algorithms that use a single secret key for encryption and decryption. Both the sender and receiver must know the secret key. Well-known symmetric functions include Data Encryption Standard (DES) and International Data Encryption Algorithm (IDEA). The U.S. Government endorsed DES as a standard in 1977. It is an encryption block cipher that operates on 64-bit blocks with a 56-bit key. It is designed to be implemented in hardware, and works well for bulk encryption. IDEA, one of the best known public algorithms, uses a 128-bit key.
Token	The logical view of a cryptographic device, as defined by a CSP's interface. A token can be hardware, a physical object, or software. A token contains information about its owner in digital form, and about the services it provides for electronic-commerce and other communication applications. A token is a secure device. It may provide a limited or a broad range of cryptographic functions. Examples of hardware tokens are smart cards and Personal Computer Memory Card International Association (PCMCIA) cards.
Verification	The process of comparing two message digests. One message digest is generated by the message sender and included in the message. The message recipient computes the digest again. If the message digests are exactly the same, it shows or proves there was no tampering of the message contents by a third party (between the sender and the receiver).
Web of trust	A trust network among people who know and communicate with each other. Digital certificates are used to represent entities in the web of trust. Any pair of entities can determine the extent of trust between the two, based on their relationship in the web. Based on the trust level, secret keys may be shared and used to encrypt and decrypt all messages exchanged between the two parties. Encrypted exchanges are private, trusted communications.