Product Guide

E-Business Server™

version 8.5

McAfee®
System Protection

Industry-leading intrusion prevention solutions

McAfee®

Product Guide

E-Business Server™

version 8.5

McAfee®
System Protection

Industry-leading intrusion prevention solutions

McAfee®

Contents

Appendices and Index

1 Introducing E-Business Server

These topics are included in this section:

- Using this guide
- Resources

Using this guide

This guide provides information on configuring and using your product. For system requirements and installation instructions, refer to the *Installation Guide*.

Audience

This information is intended for network administrators who are responsible for their company's anti-virus and security program.

Conventions

This guide uses the following conventions:

Bold Serif — All words from the user interface, including options, menus, buttons, and dialog box names.

Example:
Type the **User** name and **Password** of the desired account.

`Courier` — The path of a folder or program; a web address (URL); text that represents something the user types exactly (for example, a command at the system prompt).

Examples:
The default location for the program is:

```
C:\Program Files\McAfee\EPO\3.5.0
```

Visit the McAfee web site at:

```
http://www.mcafee.com
```

Run this command on the client computer:

```
C:\SETUP.EXE
```

Italic	For emphasis or when introducing a new term; for names of product documentation and topics (headings) within the material. Example: Refer to the *VirusScan Enterprise Product Guide* for more information.
<TERM>	Angle brackets enclose a generic term. Example: In the console tree under **ePolicy Orchestrator**, right-click <SERVER>.
	**Note:** Supplemental information; for example, an alternate method of executing the same command.
	**Tip:** Suggestions for best practices and recommendations from McAfee for threat prevention, performance and efficiency.
	**Caution:** Important advice to protect your computer system, enterprise, software installation, or data.
	**Warning:** Important advice to protect a user from bodily harm when interacting with a hardware product.

Resources

McAfee® products denote years of experience, and commitment to customer satisfaction. The McAfee PrimeSupport® team of responsive, highly skilled support technicians provides tailored solutions, delivering detailed technical assistance in managing the success of mission critical projects — all with service levels to meet the needs of every customer organization. McAfee Research, a world leader in information systems and security research, continues to spearhead innovation in the development and refinement of all our technologies.

Refer to these sections for additional resources:

- Getting product information
- Product services
- Contact information

Getting product information

Unless otherwise noted, the product documentation is provided as Adobe Acrobat .PDF files available on the product CD or from the McAfee download site

Installation Guide — System requirements and instructions for installing and starting the software.

Product Guide — Product introduction and features, detailed instructions for configuring the software, information on deployment, recurring tasks, and operating procedures.

Release Notes^ — *ReadMe*. Product information, resolved issues, any known issues, and last-minute additions or changes to the product or its documentation.

Contacts^ — Contact information for McAfee services and resources: technical support, customer service, Security Headquarters (AVERT Anti-Virus & Vulnerability Emergency Response Team), beta program, and training. This file also includes phone numbers, street addresses, web addresses, and fax numbers for company offices in the United States and around the world.

License — The McAfee License Agreement that includes all of the license types you can purchase for your product. The License Agreement sets forth general terms and conditions for the use of the licensed product.

---

^ Text files included with the software application and on the product CD.

Product services

The following services are available to help you get the most from your McAfee products:

- Beta program
- HotFixes and Patches

Beta program

The McAfee beta program enables you to try our products before full release to the public — you can learn about and test new features for existing products, as well as try out entirely new products. This program can help you test and implement updated and new features earlier, and in a safe environment. You get the chance to suggest new product features, as well as deal directly with McAfee engineering staff.

To find out more, visit:

http://www.mcafeesecurity.com/us/downloads/beta/mcafeebetahome.htm

HotFixes and Patches

HotFixes and Patches are released with updated files, drivers, executables, etc., between the major releases of a product. To access the latest HotFixes and Patches, visit:

http://www.mcafeesecurity.com/us/downloads/updates/hotfixes.asp

Contact information

Security Headquarters: AVERT

Home Page
http://www.mcafeesecurity.com/us/security/home.asp

Virus Information Library
http://vil.mcafeesecurity.com

AVERT WebImmune, Submitting a Sample *(Logon credentials required)*
https://www.webimmune.net/default.asp

AVERT DAT Notification Service
http://vil.mcafeesecurity.com/vil/join-DAT-list.asp

Download Site

Home Page
http://www.mcafeesecurity.com/us/downloads/

Anti-Virus DAT File and Engine Updates
http://www.mcafeesecurity.com/us/downloads/updates/default.asp
ftp://ftp.mcafee.com/pub/antivirus/datfiles/4.x

Anti-Spam Rules File and Engine Updates
ftp://ftp.mcafee.com/spamdefs/1.x/

Product Upgrades *(Logon credentials required)*
https://secure.nai.com/us/forms/downloads/upgrades/login.asp

HotFix and Patch Releases for Security Vulnerabilities *(Available to the public)*
http://www.mcafeesecurity.com/us/downloads/updates/hotfixes.asp

HotFix and Patch Releases for Products *(ServicePortal account and McAfee Technical Support grant number required)*
https://mysupport.nai.com/products/products.asp

Product End-of-Life Support
http://www.mcafeesecurity.com/us/products/mcafee/end_of_life.htm

Software and Hardware Technical Support

Home Page
http://www.mcafeesecurity.com/us/support/technical_support

KnowledgeBase Search
http://knowledgemap.nai.com/

McAfee Technical Support ServicePortal *(Logon credentials required)*
https://mysupport.mcafeesecurity.com

McAfee Security Alerting Service (MSAS)
http://mysupport.nai.com/supportinfo/psvans_info.asp

Customer Service

E-mail
https://secure.nai.com/us/forms/support/request_form.asp

Web
http://www.mcafeesecurity.com/us/support/default.asp

Phone — US, Canada, and Latin America toll-free:
+1-888-VIRUS NO or **+1-888-847-8766** Monday – Friday, 8 a.m. – 8 p.m., Central Time

For information on contacting McAfee worldwide offices:
http://www.mcafeesecurity.com/us/contact/home.htm

McAfee Beta Program

http://www.mcafeesecurity.com/us/downloads/beta/mcafeebetahome.htm

Training: McAfee University

http://www.mcafeesecurity.com/us/services/education/mcafee/university.htm

SECTION 1

Using E-Business Server

2 E-Business Server Basics

Welcome to E-Business Server. With E-Business Server, you can protect your data by encrypting it so that only intended co-workers and business partners can read it. You can also digitally sign data, which ensures its authenticity and that it has not been altered along the way.

What can E-Business Server do for you?

This product was designed to seamlessly integrate into existing e-Business processes (or enable new ones) to protect your corporate information while in storage or transit. The product's flexible command line interface allows you to quickly integrate E-Business Server with automated processes and web-based applications. The following are examples how you can use this product to protect your e-Business processes:

- A company's Human Resources (HR) group uses E-Business Server to securely send employee records over the Internet to a benefits provider. Prior to sending the records, an automated process on one of the company's HR servers uses E-Business Server to encrypt the records to the public key corresponding to the company's benefits provider. After the data has been encrypted, the server automatically establishes a connection to the benefits provider and transfers the data. A separate process on the benefits provider's server detects the new files, decrypts them with E-Business Server, and sends them to their final destination.
- A graphics card manufacturer shares large, confidential engineering designs with a business partner who is going to manufacture several key components for the card. The manufacturer's server automatically transfers the designs on a nightly basis via FTP to the partner's server over a firewalled leased line connection between the two companies. By leveraging the benefits of E-Business Server, these companies can now safely transfer these files over the Internet. This results in the immediate cost savings of getting rid of the inter-company leased line. E-Business Server also provides Internet bandwidth savings because it automatically compresses data before it encrypts it. Instead of sending 50 MB CAD files over the leased line, now they can send 10 MB CAD files that are encrypted to their business partner's public key.

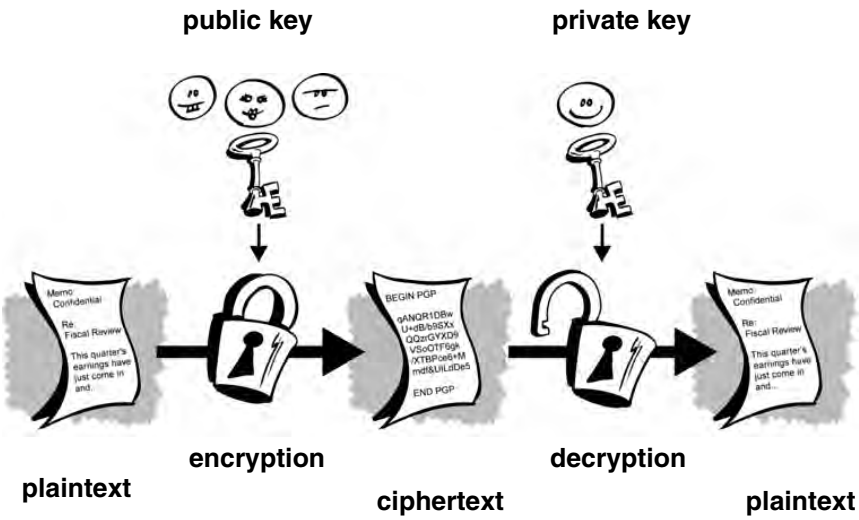
- An Internet e-Commerce site uses E-Business Server to protect all their customer demographic and credit card information as soon as they receive it. Using the command line interface, they easily integrate into their existing web server processes. Now all customer data is secured immediately. Even if a hacker could penetrate their firewall and compromise the web server this data resides on, the hacker would only be able to obtain meaningless encrypted data.

'Key' concepts

E-Business Server is based on a widely accepted and highly trusted *public key encryption* system, as shown in Figure 2-1 on page 14, in which two complementary keys, called a *key pair*, are used to maintain secure communications. One of the keys is designated as a *private key* to which only you have access and the other is a *public key* which you freely exchange with other E-Business Server users. Both your private and your public keys are stored in *keyring files*.

As its name implies, only you have access to your private key, but in order to correspond with other E-Business Server users you need a copy of their public keys and they need a copy of yours. You use your private key to sign the email messages and file attachments you send to others and to decrypt the messages and files they send to you. Conversely, you use the public keys of others to send them encrypted email and to verify their digital signatures.

Figure 2-1 Public Key Cryptography diagram



For a comprehensive overview of encryption technology, refer to *"An Introduction to Cryptography."*

Basic steps for using E-Business Server

This section takes a quick look at the procedures a user would normally follow in the course of using E-Business Server.

For details concerning any of these procedures, refer to the appropriate chapters in this book.

The order in which you perform the following tasks will vary.

1 Install E-Business Server on your server.

You will find detailed installation instructions in an accompanying *Installation Guide* or ReadMe file.

2 Configure E-Business Server to meet your needs.

You can configure E-Business Server to perform in a specific way. For example, you can specify what encryption and hash algorithms E-Business Server should use, tell E-Business Server to add a specific comment to everything you encrypt, always encrypt a copy of your data to your own key as well as your recipient's, set the level of skepticism E-Business Server should use in determining whether others' keys are valid, and so on.

You do this by setting values in E-Business Server's configuration files as described in *Using the Configuration File* on page 110

3 Create a private and public key pair.

To use E-Business Server, you need a key pair. A key pair, as described in the section "'Key' concepts," above, is composed of a private key to which only you have access and a public key that you can copy and make freely available to everyone with whom you exchange information.

You can create a new key pair any time after you have finished the E-Business Server installation procedure.

For more information about creating a private and public key pair, refer to *Creating a key pair* on page 26.

4 Exchange public keys with others (optional).

After you have created a key pair, you can begin corresponding with other E-Business Server users or users of OpenPGP-compliant applications. You will need a copy of their public key and they will need yours. Your public key is just a block of text, so it's quite easy to trade keys with someone. You can include your public key in an email message, copy it to a file, or post it on a public or corporate key server where anyone can get a copy when they need it.

You can, of course, use so me of E-Business Server's functionality without exchanging keys with others. However, to encrypt information to another person, you need his or her public key, and conversely, others will need your key if they wish to encrypt information to you.

You can keep copies of others' keys stored in your public keyring file.

For more information about exchanging public keys, refer to and *Exchanging keys with others* on page 33.

5 Validate public keys.

Validation is a tricky concept within E-Business Server, and you might want to read about it in more detail in the accompanying *Introduction to Cryptography*. Validation goes hand-in-hand with a concept called *trust*. In a nutshell, once you've obtained a copy of someone's public key, you can tell E-Business Server how you feel about the key–whether or not you've made sure that the key has not been tampered with

and that it really belongs to the purported owner. You can also tell E-Business Server whether or not and to what degree you *trust* the owner of the key to make such checks on other keys. If you tell E-Business Server you trust the key's owner, then E-Business Server will consider valid any keys the trusted person validates. Persons you trust are called trusted introducers.

To **validate** a key, you compare the unique *fingerprint* on your copy of someone's public key to the fingerprint on that person's original key. If it checks out, you can then *digitally sign* the valid key (using your own private key) to tell others (and E-Business Server) that you consider it valid.

To **trust** a key's owner (not the key, the owner), you set *trust values* in E-Business Server.

Your Corporate Security Officer can act as a trusted introducer, and you may then consider any keys signed by the corporate key to be valid keys. If you work for a large company with several locations, you may have regional introducers, and your Security Officer may be a *meta-introducer*, which is a trusted introducer of trusted introducers.

For more information checking validation and setting trust, see *Validity and trust* on page 61.

6 Encrypt and sign your files.

After you have generated your key pair and have exchanged public keys, you can begin encrypting and digitally signing files.

For more information on encryption, see *Encrypting information* on page 78.

For more information on digital signatures, see *Signing information* on page 58.

7 Decrypt and verify your files.

When someone sends you encrypted data, you can decrypt the contents and verify any appended signature to make sure that the data originated with the alleged sender and that it has not been altered.

For more information on decryption, see *Decrypting information* on page 85.

For more information on verifying digital signatures, see *Verifying a digital signature* on page 60.

8 Wipe files.

When you need to permanently delete a file, you can use the wipe command to ensure that the file is unrecoverable. The file is immediately overwritten so that it cannot be retrieved using disk recovery software.

For more information on wiping files, see *Wiping your disk* on page 92.

3 Getting Started

Setting up E-Business Server

This chapter describes where E-Business Server files are located on your machine. It also explains how to start the E-Business Server program and how to enter information on the command line.

Setting the location of E-Business Server files

E-Business Server needs to know where the following files are located:

- **Your keyring files.**

  E-Business Server stores your key pair in two files: the public portion is stored in `pubring.pkr` and the private portion in `secring.skr`. If you add another user's public key to your keyring, it is stored in the public portion of the keyring. The files are created when you run E-Business Server for the first time. Specify the path to your keyring files using the `PUBRING` (see *PUBRING* on page 133) and `SECRING` (see *SECRING* on page 135) parameters in the E-Business Server configuration file.

- **The random number seed file.**

  E-Business Server uses the data in the random seed file (`randseed.rnd`) when it generates session keys. `randseed.rnd` is created when you run E-Business Server for the first time. Specify the path to the random seed file using the `RANDSEED` parameter (see *RANDSEED* on page 134) in the E-Business Server configuration file. (See the book, *An Introduction to Cryptography* for more information on the role of *session keys* when using E-Business Server.)

- **The E-Business Server groups file.**

  E-Business Server stores any groups you create in the file `pgpgroup.pgr`. Groups are like email distribution lists—you use groups to create a list of recipients for your encrypted information. Encrypting information to the group encrypts the information to every key in the group in one operation. `pgpgroup.pgr` is created when you run E-Business Server for the first time. Specify the path to the groups file using the `GROUPSFILE` parameter (see *GROUPSFILE* on page 126) in the E-Business Server configuration file.

- **The E-Business Server configuration file.**

  E-Business Server stores a number of user-defined parameters in the configuration text file `pgp.cfg`. A configuration file enables you to define flags and parameters for E-Business Server, eliminating the need to define these parameters at the command line. The E-Business Server configuration file is created when you run E-Business Server for the first time. You can specify the path to the E-Business Server configuration file using the environment variable `PGPPATH` (see *PGPPATH* on page 18).

PGPPATH

`PGPPATH` is an environment variable that identifies the location of your E-Business Server configuration file.

Syntax

```
SET PGPPATH=<pgppathname>
```

Notes

You can also specify the location of the E-Business Server configuration file using `--pgppath` from the command line. If you use the command line, you must specify both the path and the filename. This option is especially useful to CGI developers who can't set environment variables.

Default file locations on Unix

The first time you start E-Business Server, the software checks to see if `PGPPATH` is set to a particular pathname.

- If `PGPPATH` is defined, the software looks for the E-Business Server configuration file (`pgp.cfg`) in the directory specified by `PGPPATH`. If `pgp.cfg` does not exist in the directory specified, E-Business Server creates it using `/usr/local/ebs/pgp-template.cfg` or a `pgp-template.cfg` file in the same location as the E-Business Server executable.

- If `PGPPATH` is not defined, the software looks for `pgp.cfg` in the user's home directory, as defined by the environment variable `HOME`. If `pgp.cfg` does not exist, E-Business Server creates the `.pgp` directory within the home directory and creates the `pgp.cfg` file within `.pgp`.

  E-Business Server then places the keyring files, the randseed file, and the group file in the `.pgp` directory off your home directory (`HOME/.pgp`) *after* you run E-Business Server for the first time.

As the administrator of E-Business Server for Unix, you can use multiple configuration files for flexibility in configuration. You can set default options in a system preferences configuration file, `/usr/local/ebs/pgp.cfg`. If this file exists, the default settings in this file are used unless overridden by settings specified in the other E-Business Server configuration files.

E-Business Server checks for the existence of /usr/local/ebs/pgp-policy.cfg. If this file is present, E-Business Server reads this configuration file after parsing the normal configuration file and command line options. The settings in the pgp-policy.cfg override most settings in the normal configuration file, pgp.cfg. The only parameters that the pgp-policy.cfg file cannot override are the following parameters—PUBRING, SECRING, RANDSEED, MYNAME, DEFAULT-KEY and ALIAS.

Default file locations on Windows NT

The first time you start E-Business Server, the software checks to see if PGPPATH is defined.

- If PGPPATH is defined, the software puts the pgp.cfg file in the directory specified by PGPPATH.
- If PGPPATH is not defined, the software checks to see if the environment variable USERPROFILE is defined.
  - If USERPROFILE is defined, the software puts the pgp.cfg file in the <USERPROFILE>\Personal\pgp directory. If pgp.cfg does not exist, E-Business Server creates it within this directory.
  - If USERPROFILE is not defined, the software puts the pgp.cfg file in <SYSTEMROOT>\pgp.
  - E-Business Server then places the keyring and group files in the <USERPROFILE>\Personal\pgp directory *after* you run E-Business Server for the first time.
  - E-Business Server places the randseed file in the <SYSTEMROOT>\Profiles\All Users\Application Data\Network Associates\pgp directory.

Default file locations on Windows 2000

The first time you start E-Business Server, the software checks to see if PGPPATH is defined.

- If PGPPATH is defined, the software puts the pgp.cfg file in the directory specified by PGPPATH.
- If PGPPATH is not defined, the software checks to see if the environment variable USERPROFILE is defined.
  - If USERPROFILE is defined, the software puts the pgp.cfg file in the <USERPROFILE>\My Documents\pgp directory. If pgp.cfg does not exist, E-Business Server creates it within this directory.
  - If USERPROFILE is not defined, the software puts the pgp.cfg file in <SYSTEMROOT>\pgp.
  - E-Business Server then places the keyring and group files in the <USERPROFILE>\My Documents\pgp directory *after* you run E-Business Server for the first time.
  - E-Business Server places the randseed file in the <ALLUSERSPROFILE>\Application Data\Network Associates\pgp directory.

Starting E-Business Server

Before you can use E-Business Server in Unix, you may need to add the E-Business Server installation directory to your PATH using the `PATH` variable.

E-Business Server is available at all times. To use E-Business Server, type "`ebs`" and then the command option for the operation you want to perform. You do not need to specifically start or end the program.

```
ebs <option>
```

The following text appears:

```
McAfee E-Business Server Version #.#

(c) 1991-2002 McAfee, Inc. All Rights Reserved.


Help for basic operations.

Use "--help" with the following options for individual usage.

 --armor               Encode a file with E-Business Server's
                         base-64 encoding, with optional
                         compression

 --decrypt             Decrypt data that was previously encrypted

 --encrypt             Encryption

 --help                Display help

 --key-edit            Specifies a keypair to be updated

 --key-export          Exports a key from the keyring

 --key-gen             Generate a new keypair

 --key-list            Display keys on the keyring

 --key-sign            Sign a key

 --list-aliases        Show the active aliases

 --sign                Perform a cryptographic signature on input
                         data

 --version             Displays version information about the
                         E-Business Server executable

 --wipe                Performs a secure deletion of files

For help on key management operations:    ebs --help --key
For help on key editing operations:       ebs --help --key-edit
For help on keyserver operations:         ebs --help --keyserver
For help on group operations:             ebs --help --group
For help on smartcard operations:         ebs --help--smartcard
For help on X.509 operations:             ebs --help --x509
```

Checking the version you have installed

To find out which version you are running, use the following syntax:

```
ebs --version
```

The following information appears:

```
McAfee E-Business Server Version #.#

(c) 1991-2001 McAfee, Inc. All Rights Reserved.

Product Name: McAfee E-Business Server

Product Mode: Full

Version     : #.#

Full Version: #.#.#

Build Stage : Release

Build Number: #

Debug Info  : Not present
```

E-Business Server command syntax

In versions of E-Business Server prior to 7.1.0, E-Business Server command line options were typically one or two letters (for example, `-kg` was used for key generation). This older format is now known as *legacy mode*. In current versions of E-Business Server, the command line options are longer and more descriptive of the operation (for example, `--key-gen` is used for key generation). These newer options are known as *long* options.

You cannot use the long options and the legacy options interchangeably. To specify the command line format you want to use, set the `CMDLINE-FORMAT` parameter in the E-Business Server configuration file:

```
cmdline-format = <legacy | long>
```

The default format is `long`.

Entering long options on the command line

If E-Business Server is set to work in the long format, the default format, then you use E-Business Server by typing `ebs` followed by arguments that start with two dashes (`--`):

```
ebs --<long-option>
```

For example, to create a new key pair, you would enter the following on the command line:

```
ebs --key-gen
```

Entering legacy options on the command line

If E-Business Server is set to work in legacy mode, then you use E-Business Server by typing `ebs` followed by one dash (-) and then enter the options/parameters you need to perform the operation.

```
ebs -<legacy option>
```

For example, to create a new key pair, you would enter the following on the command line:

```
ebs -kg
```

The command parser converts the legacy options you enter on the command line to the current long options for the operation before passing them on for processing. E-Business Server displays the long-option equivalent of all legacy commands when `INFO=Verbose` in the E-Business Server configuration file (the same as `VERBOSE=2` in legacy mode). For more information on setting the `INFO` parameter, see *INFO* on page 127.

Single dash options are allowed in long mode if an alias (shortcut) exists that maps it to a long option. This allows you to set the legacy options that you've become accustomed to as aliases for the equivalent long options. Aliases are set in the E-Business Server configuration file using the `ALIAS` parameter.

For example, you might set `-kg` as an alias for the key generation long option as shown below:

```
ALIAS -kg --key-gen
```

Once this is set, you can enter `-kg` on the command line instead of entering `--key-gen` when you want to generate a new key pair.

For detailed instructions on how to create aliases, see *ALIAS* on page 112.

Entering configuration parameters on the command line

Note that any of the E-Business Server configuration parameters described in *Using the Configuration File* on page 110 can also be entered as long options on the command line.

If you are working in non-legacy mode, then you can set options on the command line by using the following syntax:

```
ebs --<option> <value>
```

For example, if the `ARMOR` parameter is set to `on` in the E-Business Server configuration file, you can override this setting by using the `--armor` option on the command line:

```
ebs --encrypt --armor off message.txt --user smith
```

If you are working in legacy mode, then you must precede the parameter setting with a plus (+) character. For example, if the `ENCRYPT-TO-SELF` parameter is turned `off` in the configuration file, but you want to use it in a single legacy operation, then enter the following on the command line:

```
ebs -e +ENCRYPT-TO-SELF=on message.txt smith
```

For the location of the `pgp.cfg` file, please refer to *Setting the location of E-Business Server files* on page 17.

Specifying keys using the user ID

The user ID is part of every key. When performing tasks with E-Business Server, you typically identify the key you want to use by specifying the key's user ID or a fragment of the user ID.

When specifying the user ID, keep the following in mind:

- User IDs can be up to a maximum of 149 characters.
- Be as specific as you can. If you have three keys on your keyring whose user IDs contain "John," (Dr. John Banner, John Huang, and John Schwartz) then specifying "John" as the user ID results in a list of all matching keys and an error message.
- To specify multiple word user IDs, enclose the text in quotes. For example:

  `"Sophie Luu"`
- E-Business Server is not case-sensitive. `"John"` and `"john"` are identical to E-Business Server.
- Specifying a user ID that begins with a dash results in an error message. Dashes are used to introduce commands, and therefore cannot be used at the beginning of a user ID.
- When performing manual tasks with E-Business Server, user IDs can be very convenient; however, for automated tasks, it is recommended that you always specify the key ID. See *Specifying keys using the key ID* below.

Specifying keys using the key ID

In most cases, you enter a user ID or part of a user ID to select a key. However, you can also use the hexadecimal *key ID* to select a key. To do so, enter the key ID, with a prefix of "`0x`", instead of the user ID:

```
ebs --key-list 0x67F796C2
```

This command instructs E-Business Server to display the key with the ID `0x67F796C2`.

This feature is particularly useful if you have two different keys from the same person, with the same user ID. You can pick the correct key by specifying the specific key ID.

Most command syntax in this guide specifies `<userID>`. Unless otherwise specified, `<userID>` and `<keyID>` can be used interchangeably.

Getting help while you work

Use the following commands to get help while you work:

Command	Help provided
ebs --help	General E-Business Server help.
ebs --help --key	Usage information on key management operations.
ebs --help --key-edit	Usage information on key editing operations.
ebs --help --keyserver	Usage information on key server operations.
ebs --help --group	Usage information on group operations.
ebs --help --x509	Usage information on X.509 operations.

Command	Help provided
ebs --help --smartcard	Usage information on operations you can perform on a smart card.
ebs --help <primary option>	Individual usage information for primary command line options. For example, to display syntax for the encryption command, enter: ebs --help --encrypt

Cancelling an operation

To cancel the current operation or a long running operation, press `Ctrl-C` at any time.

4 Creating and Exchanging Keys

This section describes how to generate, view, and manage the public and private key pair that you need to correspond with other E-Business Server users. It also explains how to distribute your public key and obtain the public keys of others so that you can begin exchanging private and authenticated information.

Choosing a key type

E-Business Server provides you with two key types to choose from: Diffie-Hellman/DSS and RSA. Versions of E-Business Server prior to 5.0 used RSA keys exclusively. Versions later than 5.0 introduced the ElGamal variant of Diffie-Hellman technology.

With E-Business Server versions 7.0 and above, the RSA key format has been improved to provide support for features previously available only to Diffie-Hellman/DSS keys: support for Additional Decryption Keys (ADKs), designated revokers, multiple encryption subkeys, and photo ID features. These features are not available to users with RSA keys created prior to Version 7.0, now known as RSA Legacy keys.

Which key type is the right choice for you?

- Choose **Diffie-Hellman/DSS** or **RSA** if you want to take advantage of many E-Business Server key features; including, Additional Decryption Keys (ADKs), designated revokers, multiple encryption subkeys, and photo IDs.
- Choose **RSA** or **RSA Legacy** if you plan to correspond with people who are using RSA keys.
- Choose **RSA Legacy** only if those you communicate with are using older versions of E-Business Server; otherwise choose the new **RSA** key format. (The two versions are not compatible with each other.)

The RSA key type is only fully compatible with E-Business Server versions 7.0 and above, and some other OpenPGP applications.

If you plan to correspond with people who are still using RSA Legacy keys, you might want to generate an RSA Legacy key pair, which is compatible with older versions of the program.

Creating a key pair

Unless you have already done so while using another version of E-Business Server, the first thing you need to do before sending or receiving encrypted and signed email is create a new key pair. A key pair consists of two keys: a private key that only you possess and a public key that you freely distribute to those with whom you correspond. You generate a new key pair from the E-Business Server command line.

If you are upgrading from an earlier version of E-Business Server, you have probably already generated a private key and have distributed its matching public key to those with whom you correspond. In this case, you don't have to make a new key pair (as described in the next section). Instead, use the `PUBRING` and `SECRING` parameters in the E-Business Server configuration file to point to your keyrings. For more information, see *Specifying configuration values* on page 110.

It's best to create the fewest number of key pairs possible. You generally need only one key pair. However, if you want one key pair for office use and one for home use, consider the potential disadvantages—if you place both public keys on a public key server, will someone who wants to send you encrypted information know which key to use? Will you remember the passphrases for both keys? It's tempting to create multiple sets of keys, but later you might find yourself wishing you hadn't.

To create a key pair:

1 Enter the following at the command line:

```
ebs --key-gen
```

2 Choose a key type.

- Enter `1`, the default option, to create a **DH/DSS** key.
- Enter `2` to create an **RSA** key.
- Enter `3` to create an **RSA Legacy** key.

RSA Legacy keys do not support subkeys.

3 Select the size you want the key to be. A larger key size may take a long time to generate, depending on the speed of the computer you are using.

For DH/DSS key pairs, the signing key can only be 1024 bits, so the size you enter applies to the encryption subkey. For RSA v4 key pairs, the size you enter applies to both the signing key and the encryption subkey. For RSA Legacy keys, only one key is used for both signing and encryption, so the size you enter applies to that key.

The key size corresponds to the number of bits used to construct your digital key. A larger key is stronger. However, when you use a larger key, it takes more time to encrypt and decrypt. You need to strike a balance between the convenience of performing E-Business Server functions quickly with a smaller key and the increased level of security provided by a larger key.

Unless you are exchanging extremely sensitive information that is of enough interest that someone would be willing to mount an expensive and time-consuming cryptographic attack in order to read it, you are safe using a key composed of 1024 bits.

For a **DH/DSS** key or a new **RSAv4** key:

- Enter `1` to create a key of 1024 bits.
- Enter `2` to create a key of 2048 bits.
- Enter `3` to create a key of 3072 bits.
- Enter any key size you want between 1024 bits and 4096 bits.

For an **RSA Legacy** key:

- Enter `1` to select a key size of 1024 bits.
- Enter `2` to select a key size of 2048 bits.
- Enter any key size you want between 1024 bits and 2048 bits.

4 Enter the text that will comprise your user ID (149 characters, maximum). E-Business Server prompts you with instructions. It's not absolutely necessary to enter your real name or even your email address. However, using your real name makes it easier for others to identify you as the owner of your public key. For example:

```
Robert M. Huang <rhuang@mcafee.com>
```

If you do not have an email address, use your phone number or some other unique information that would help ensure that your user ID is unique.

Do not create a user ID that starts with a dash. You cannot specify a user ID that starts with a dash in key commands.

5 Enter a passphrase, a string of characters or words you want to use to maintain exclusive access to your private key.

For more information on creating an effective passphrase, see *Creating a passphrase that you will remember* on page 30.

6 When prompted, enter the same passphrase again for confirmation.

7 If prompted, enter random text to help the E-Business Server software accumulate some random bits to create your keys. Enter keystrokes that are reasonably random in their timing.

The generated key pair is placed on your public and private keyrings.

To view your new key pair, use the `--key-list` option. For more information see, *Viewing your keys* on page 39.

We recommend that you use the `--key-export` command option to copy your new public key from your public keyring and place it in a separate public key file suitable for distribution to your friends. The public key file can be sent to your friends for inclusion in their public keyrings. For more information, see *Exchanging keys with others* on page 33.

Creating subkeys

RSA Legacy keys do not support subkeys.

Every key is actually two keys: a signing key and an encryption subkey. E-Business Server provides the ability to create and revoke new encryption keys without sacrificing your master signing key and the signatures collected on it. One of the most common uses for this feature is to create multiple subkeys that are set to be used during different periods of the key's lifetime.

For example, if you create a key that will expire in three years, you might also create 3 subkeys and use each of them for one of the years in the lifetime of the key. This can be a useful security measure and provides an automatic way to periodically switch to a new encryption key without having to recreate and distribute a new public key.

To avoid confusion later, do not overlap the validity periods of your subkeys.

To create an encryption subkey:

1 Enter the following on the command line:

```
ebs --key-gen --subkey
```

2 Enter the user ID for the existing master key. For example:

```
rhuang@mcafee.com
```

3 Enter the passphrase for the existing master key.

4 Choose a size for the encryption subkey, or enter the desired key size in bits.

- Enter `1` to select a key size of 1024 bits.
- Enter `2` to select a key size of 2048 bits.
- Enter `3` to create a key of 3072 bits.
- Enter any key size you want between 1024 bits and 4096 bits.

5 If prompted, enter random data to use for the key generation process.

E-Business Server creates the subkey. For information on viewing keys, see *Viewing your keys* on page 39.

Creating a key pair on a smart card

This section applies to Windows installations only, and assumes that you have a supported smart card reader installed with appropriate driver software. .

You can create and store your keys on a smart card and access them using a PIN number rather than a passphrase. The smart card has the added protection of being with you at all times—a key on a smart card is less vulnerable than the same key stored on your computer.

The private portion of your key pair never leaves your smart card—it's non-exportable. Therefore, decryption and signing operations take place directly on the card. The exception to this would be if you generate a key pair on your desktop, rather than on the card, and then copy the key pair to your card.

Before you can generate a key on a smart card, you must specify the smart card type using the `SMARTCARD-TYPE` parameter in the E-Business Server configuration file or by setting it on the command line using `--smartcard-type`. For more information, see *SMARTCARD-TYPE* on page 137.

If you want to use a smart card other than one that we have listed as being supported, then you must set the `SMARTCARD-TYPE` to `other`, as well as specify the path to the DLL to use with it by setting the `SMARTCARD-DLL` parameter. You can also set this on the command line using `--smartcard-dll`. For more information on specifying the DLL, see *SMARTCARD-DLL* on page 137.

To create a key pair on a smart card:

1 Put your smart card in the smart card reader.

Removing your smart card from the smart card reader while generating a new key pair on the smart card, or when using the keys on your smart card in later commands, may result in unpredictable behavior.

2 Enter the following on the command line:

```
ebs --key-gen [--smartcard-type <type> [--smartcard-dll <path to
dll>] --smartcard
```

3 Do one of the following:

- Enter `2` to create an **RSA** key.
- Enter `3` to create an **RSA Legacy** key.

Diffie-Hellman/DSS keys are not supported on smart cards.

4 Enter `1024` for the key size. (Due to the limited space on current smart cards, key sizes other than 1024 bits may not be supported.)

5 Enter the text that will comprise your user ID (149 characters, maximum). E-Business Server prompts you with instructions. It's not absolutely necessary to enter your real name or even your email address. However, using your real name makes it easier for others to identify you as the owner of your public key. For example:

```
Robert M. Huang <rhuang@mcafee.com>
```

If you do not have an email address, use your phone number or some other unique information that would help ensure that your user ID is unique.

6 Enter your smart card PIN number as your passphrase.

Your new key pair is generated and stored directly on your smart card.

For information on viewing the contents of your smart card, see *Viewing your keys on page 39*.

Creating a passphrase that you will remember

Encrypting a file and then finding yourself unable to decrypt it is a painful lesson in learning how to choose a passphrase you will remember. Most applications require a single word password between three and eight letters. For a couple of reasons we do not recommend that you use a single-word passphrase. A single word password is vulnerable to a dictionary attack, which consists of having a computer try all the words in the dictionary until it finds your password. To protect against this manner of attack, it is widely recommended that you create a word that includes a combination of upper and lowercase alphabetic letters, numbers, punctuation marks, and spaces. This results in a stronger password, but an obscure one that you are unlikely to remember easily.

Trying to thwart a dictionary attack by arbitrarily inserting a lot of funny non-alphabetic characters into your passphrase has the effect of making your passphrase too easy to forget and could lead to a disastrous loss of information because you can't decrypt your own files. A multiple word passphrase is less vulnerable to a dictionary attack. However, unless the passphrase you choose is something that is easily committed to long-term memory, you are unlikely to remember it verbatim. Picking a phrase on the spur of the moment is likely to result in forgetting it entirely. Choose something that is already residing in your long-term memory. It should not be something that you have repeated to others recently, nor a famous quotation, because you want it to be hard for a sophisticated attacker to guess. If it's already deeply embedded in your long-term memory, you probably won't forget it. Of course, if you are reckless enough to write your passphrase down and tape it to your monitor or to the inside of your desk drawer, it won't matter what you choose.

Working with public and private keyrings

Your keys are stored in two files, called the public and private keyrings:

- `secring.skr` contains the private portion of your key pair. To protect it, E-Business Server stores the key encrypted to your passphrase.
- `pubring.pkr` contains your public key. You can add to the keyring the public keys of everyone with whom you exchange messages.

The keyrings contain binary information, and thus you can't view or manipulate their contents directly.

All operations on your keyrings actually apply to *both* keyrings at once. E-Business Server cannot open just the private keyring or just the public keyring.

To learn how to view keys on a keyring, see *Viewing your keys* on page 39.

Changing the location or names of your keyrings

By default, E-Business Server looks for the files `pubring.pkr` and `secring.skr`. If you choose to rename your keyrings, you must specify the keyrings' names in E-Business Server's configuration file (using the `PUBRING` and `SECRING` parameters).

Unix

- The default path for `pubring.pkr` is `<HOME>/.pgp/pubring.pkr`
- The default path for `secring.skr` is `<HOME>/.pgp/secring.skr`

Windows NT

- The default path for `pubring.pkr` is `<USERPROFILE>\Personal\pgp\pubring.pkr`
- The default path for `secring.skr` is `<USERPROFILE>\Personal\pgp\secring.skr`

Windows 2000

- The default path for `pubring.pkr` is `<USERPROFILE>\My Documents\pgp\pubring.pkr`
- The default path for `secring.skr` is `<USERPROFILE>\My Documents\pgp\secring.skr`

You can copy your keyring files to another location on your hard drive or to a floppy disk. By default, the keyrings are stored along with the other program files in the directory identified by the `PGPPATH` environment variable, but you can save backups in any location you like. For more information, see *PGPPATH* on page 18.

Backing up your keys

Keys generated on a smart card cannot be backed up because the private portion of your keypair is non-exportable.

E-Business Server does not automatically back up your keyrings. Once you have generated a key pair, it is wise to put a copy of it in a safe place in case something happens to the original. Copy your keyring files as you would any other file.

Your private and public keys are stored in separate keyring files. You can copy them to another location on your hard drive or to a floppy disk. You can save your backups in any location you like. For more information on the default keyring locations, see *Changing the location or names of your keyrings* on page 31.

Protecting your keys

Besides making backup copies of your keys, you should be especially careful about where you store your private key. Even though your private key is protected by a passphrase that only you should know, it is possible that someone could discover your passphrase and then use your private key to decipher your email or forge your digital signature. For instance, somebody could look over your shoulder and watch the keystrokes you enter or intercept them on the network.

To prevent anyone who might happen to intercept your passphrase from being able to use your private key, you should store your private key only on your own computer. If your computer is attached to a network, you should also make sure that your files are not automatically included in a system-wide backup where others might gain access to your private key. Given the ease with which computers are accessible over networks, if you are working with extremely sensitive information, you may want to keep your private key on a floppy disk, which you can insert like an old-fashioned key whenever you want to read or sign private information.

As another security precaution, consider assigning a different name to your private keyring file and then storing it somewhere other than in the default E-Business Server folder where it will not be so easy to locate. However, be aware that you need to let E-Business Server know where it is.

What if I forget my passphrase or lose my key?

If you lose your key or forget your passphrase and do not have a backed up copy from which to restore your key, you will never again be able to decrypt any information encrypted to your key. You can, however, reconstruct your key if you implement a key restoration policy, where your key is encrypted and stored on a key reconstruction server.

What is key reconstruction?

You cannot reconstruct a key that was generated on a smart card because the private portion of the key pair is non-exportable.

You can set up a key reconstruction server to act as sort of a safety-net for you if you lose your private key or passphrase. The reconstruction server stores your key in such a way that only you can access it. Your company does not have the ability to decrypt your key.

You need to provide recovery information—five questions and five secret answers—and send your key to the key reconstruction server.

Once your key is on the server, you can restore it at anytime. To learn how to reconstruct your key, see *Reconstructing your key* on page 56.

To send your key to your company's key reconstruction server:

1 Enter the following at the command line:

```
ebs --reconstruct-data <userID or keyID>
```

Where `<userID or keyID>` is the ID belonging to the key you want to store on the key reconstruction server. For example:

```
ebs --reconstruct-data rhuang@mcafee.com
```

2 When asked for a prompt, enter a question that only you can answer. Choose an obscure personal question with an answer that you are not likely to forget. Your question can be up to 95 characters in length.

An example of a good question might be, "Who took me to the beach?" or "Why did Fred leave?"

An example of a bad question would be, "What is my mother's maiden name?" or "Where did I go to high school?"

If you prefer, you can also leave the prompts blank and simply provide answers.

3 Type an answer to the corresponding question. Your answers can be up to 255 characters in length.

Your answers are case-sensitive.

4 Repeat steps 2 and 3 until you have 5 sets of prompts and answers.

5 Enter your passphrase.

6 Enter the name of the server on which you want to store your reconstruction data using the following format:

```
ldap://<IP address or DNS name of key server>
```

For example:

ldap://keyserver.mycorp.com

The server can also be specified in the E-Business Server configuration file. For more information, see *Specifying configuration values* on page 110.

Your private key is split into five pieces, using Blakely-Shamir key splitting. Three of the five pieces are needed to reconstruct the key. Each piece is then encrypted with the *hash*, the uniquely identifying number, of one answer. If you know any 3 answers, you can successfully reconstruct the whole key. To learn how to reconstruct your key, see *Reconstructing your key* on page 56.

Exchanging keys with others

After you create your keys, you need to make them available to others so that they can send you encrypted information and verify your digital signature. You have three alternatives for distributing your public key:

- Export your public key to a file.
- Make your public key available through a public key server.

- Include your public key in an email message.

Your public key is basically composed of a block of text, so it is quite easy to make it available through a public key server, include it in an email message, or export (copy) it to a file. The recipient can then use whatever method is most convenient to add your public key to their public keyring.

Exporting (copying) your key to a file

To export your key to a file, which you can then freely distribute to others, use the `--key-export` option.

```
ebs --key-export <userID> --output <key_filename>
```

This exports the public portion of your key. For example:

```
ebs --key-export "John Lee" --output johnkey.pgp
```

You can then give the file "`johnkey.pgp`" to anyone who wants a copy of your key. If the filename does not have a `.pgp` extension, then EBS automatically adds it. For example:

```
ebs --key-export "John Lee" --output johnkey.bin
```

In this example, E-Business Server creates a file called `johnkey.bin.pgp`.

With your secret key

To export both parts of your key pair—your public and private key—add the `--with-private` modifier to the `--key-export` option. For example, you may use this option when backing up your keys to a different location.

```
ebs --key-export <userID> --output <key_filename> --with-private
```

For example:

```
ebs --key-export "John Lee" --output johnkey.pgp --with-private
```

The `--key-export` option produces a file with a single, binary key on it.

In a format you can email

To extract the key in ASCII-armored format, which makes it easy to paste into email, add the `--armor` modifier to the `--key-export` option.

```
ebs --key-export <userID> --output <key_filename> --armor
```

For example:

```
ebs --key-export "John Lee" --output johnkey.pgp --armor
```

In this example, E-Business Server creates a file called `johnkey.pgp.asc`, which you can paste into email.

Exporting multiple keys

If you have multiple keys on your keyring with the same or similar user ID and you want to export all matching keys, then you must use the `--multi` modifier with the `--key-export` option.

```
ebs --key-export <userID> --output <key_filename> --multi
```

For example:

```
ebs --key-export John --output keys.pgp --multi
```

In this example, all keys with "`John`" in the user ID are exported to the file `keys.pgp`. So, if your keyring included a key with the user ID "John Lee" and a key with the user ID "John Peterson", both keys are exported.

Adding a key to your keyring

You can add someone else's public key to your keyring using the `--key-add` option and including the name of the file containing the key you want to add on the command line. E-Business Server uses your default public keyring specified by the `PUBRING` parameter in the configuration file (see *PUBRING* on page 133).

To add a key to your keyring:

1 Enter the following command at the command line:

```
ebs --key-add <key_filename>
```

For example:

```
ebs --key-add bobkey.pgp
```

E-Business Server finds the new key and asks if you want to add the key to your keyring.

2 Enter `y` to add the new key.

E-Business Server adds the key to your keyring and lists the key ID, user ID and signatures belonging to the key that was added.

E-Business Server does not allow you to add duplicate keys to your keyring. If the second key has any differences, such as an additional user ID, E-Business Server merges the changes.

Exchanging keys using a key server

By default, E-Business Server uses the key server specified by the `KEYSERVER` parameter in the E-Business Server configuration file (see *KEYSERVER* on page 128). Optionally, you can specify a key server URL on the command line by using the following syntax:

```
--keyserver <keyserver_URL>
```

For example, you might enter the following:

```
--keyserver ldap://keyserver.mycorp.com
```

Occasionally, you may need to update the keys on your keyring and get the most recent versions from a key server. For information on updating the keys on your keyring from a key server, see *Updating keys on your keyring* on page 43.

Adding your key to a key server

You can add your key to a key server so that it is available to others. To add your key to a server, you use the `--keyserver-send` option. E-Business Server copies the key from the keyring and places it on the server.

If you do not specify a key server URL on the command line, E-Business Server uses the default URL specified in the E-Business Server configuration file. For more information on setting the key server URL in the configuration file, see *Specifying configuration values* on page 110.

```
ebs --keyserver-send <keyID or userID> [--keyserver
<keyserver_URL>]
```

For example:

```
ebs --keyserver-send "John Lee" --keyserver
ldap://keyserver.pgp.com
```

If you want to send several keys to a key server in a single operation, then simply list all the user IDs at the end of the command line.

```
ebs --keyserver-send [--keyserver <keyserver_URL>] <userID1>
<userID2> <userID3>...
```

If more than one key matches any of the specified user IDs, then you must also include `--multi` to send all matching keys to the key server.

Removing your key from a key server

You can remove your key from a key server so that it is no longer available to others. To remove your key from a server, you use the `--keyserver-delete` option. For more information, see *Deleting your key from a key server* on page 56.

Searching for a key on a key server

You can search a public key server to find someone's public key. To search a key server for a key, use the `--keyserver-search` option.

```
ebs --keyserver-search <keyID or userID> [--keyserver
<keyserver_URL>]
```

E-Business Server displays all matching keys found. You can show more information about the keys found by including the following modifiers:

- To display the additional user IDs for each matching key, add the `--with-userids` modifier to the command line.
- To display the signatures attached to each matching key, add the `--with-sigs` modifier to the command line.
- To display more details about keys, such as the key's fingerprint, creation date, expiration date, photo IDs and tokens present, add the `--key-detail` modifier to the command line. If more than one key is found during the search, and you want to display information for all matching keys, then you must also include `--multi`.

Once you find the key you want, you can add it to your keyring using the `--keyserver-fetch` option. For more information, see *Getting someone's public key from a key server* below.

Getting someone's public key from a key server

You can copy a key to your keyring directly from a key server. To get someone's key from a key server and automatically add it to your keyring, you must use the `--keyserver-fetch` option.

```
ebs --keyserver-fetch <userID> [--keyserver <keyserver_URL>]
```

For example:

```
ebs --keyserver-fetch "John Lee"
```

In this example, E-Business Server searches `keyserver.pgp.com` for the key belonging to `John Lee`. If only one matching key is found, then E-Business Server adds it to your keyring.

If more than one key matches the information you provided, then E-Business Server displays each key—one at a time—and asks if you want to add it to your keyring.

To display all additional user IDs belonging to each matching key, add the `--with-userids` modifier to the command line. To display the signatures for each matching key, add the `--with-sigs` modifier to the command line.

If you want E-Business Server to automatically add all matching keys to your keyring without first prompting you, include the `--add-all` modifier on the command line.

5 Managing Keys

Key management is the secure administration of keys or keyrings. Administrative tasks you might perform on your keys include the following:

To:	See:
List the keys on your keyring	*Viewing your keys* on page 39
Remove keys from your keyring	*Removing keys from your keyring* on page 41
Verify the contents of your keyring	*Verifying the contents of your public keyring* on page 43
Update keys on your keyring using a key server	*Updating keys on your keyring* on page 43
Add or change user IDs on your key	*Adding and removing user IDs* on page 45
Add a designated revoker to your key	*Adding a designated revoker to your key* on page 47
Change your passphrase	*Changing your passphrase* on page 46
Change trust parameters	*Editing trust options for your key* on page 46
Remove a signature from your key	*Remove a signature from a key* on page 47
Add or remove photo IDs	*Adding and removing photo IDs* on page 48
Disable/enable your key	*Disabling and enabling a key* on page 50
Revoke your key	*Revoking a key* on page 49
Split and rejoin your key	*Splitting and rejoining a key* on page 50
Create additional decryption keys (ADKs)	*Additional Decryption Keys* on page 53
Delete your key from a key server	*Deleting your key from a key server* on page 56
Reconstruct your key	*Reconstructing your key* on page 56

Managing your keyring

You may accumulate many keys in the course of using E-Business Server. Over time, you may want to view, update, or remove the keys on the keyring.

All operations on your keyrings actually apply to *both* keyrings at once. E-Business Server cannot open just the private keyring or just the public keyring.

Viewing your keys

Viewing your keys is the most basic key management operation. You can list them using variations of the `--key-list` option. The key list includes the following information for each key: algorithm, type, size, flags, key ID, and primary user ID.

For a better understanding of the information and flags displayed in a key list operation and for examples of the various `--key-list` options, see *Understanding Key List Displays*.

To list all the keys—private and public—on your default keyring use the following syntax:

```
ebs --key-list
```

E-Business Server lists all the keys on your keyring. Public keys are represented with "`pub`" in the `Type` column, and key pairs are represented with "`pair`" in the `Type` column.

To display specific keys on your keyring, specify the user IDs using the following syntax.

```
ebs --key-list <userID1> <userID2> <userID3>...
```

For example:

```
ebs --key-list "Lisa Jameson" tschlubb "Brendan Chriss"
```

All keys matching these user IDs appear.

For an example of a basic key list display, see *Example of --key-list option* on page 206.

On another keyring

To view keys on another keyring, specify the keyring's filename.

Keep in mind that E-Business Server will try to open both the public and private keyrings at once; the private portion of the keyring must be in the same directory as the public keyring and must either have the filename `secring.skr`, or you must specify its new name using the `SECRING` configuration parameter. You can set the `SECRING` parameter's value either in the configuration file or on the command line as shown below.

```
ebs --secring <path>/<secret_key_filename>
--pubring <path>/<public_key_filename>
```

For example:

```
ebs --secring <HOME>/.pgp/secring.skr --pubring
<HOME>/.pgp/pubring.pkr
```

Where `<HOME>` is your home directory.

Display all user IDs associated with each key

To display all user IDs associated with each key on a keyring, use the following syntax:

```
ebs --key-list --with-userids
```

E-Business Server sorts the key list by size. Keys of the same size are sorted alphabetically using the first letter of the key's primary user ID. Additional user IDs appear under the primary user ID for each key. Each of the additional user IDs is represented with "`uid`" in the `Type` column. A photo ID on your key is represented with "`pid`" in the `Type` column.

For an example of a key list display showing additional user IDs, see *Example of --key-list --with-userids option* on page 206.

Display all signatures associated with each key

To display the signatures attached to each key on your keyring, use the following syntax:

```
ebs --key-list --with-sigs
```

E-Business Server lists the keys on your keyring. The signatures on each key appear below the user ID they belong to. The signatures are represented by "`sig`" in the `Type` column.

For an example of a key list display showing all signatures associated with each key, see *Example of --key-list --with-sigs option* on page 206.

Getting more information about keys

To see additional information about keys—creation dates, fingerprints, expiration dates, subkeys, ADKs, Revokers, tokens, or photo IDs present—use the `--key-detail` option. (If more than one key matches a specified user ID, then you must also include the `--multi` modifier on the command line.)

```
ebs --key-detail <userID1> <userID2> <userID3>...
```

For example, if you wanted to view the properties of the key belonging to Odette Richards, then you would enter the following:

```
ebs --key-detail "Odette Richards"
```

E-Business Server finds the matching key on your keyring, and displays the following information about the key:

```
Primary User ID: Odette Richards <orichards@mcafee.com>

Key ID:              0x2A95C822B56702B3

Type:                DH/DSS key pair

Size:                2048/1024

Validity:            Valid

Trust:               Implicit

Created:             2000-03-22

Expires:             Never

Cipher:              CAST

Photo:               Not present

Token:               Not on token
```

```
Fingerprint:

5078 6543 2006 CEE2  9941 8A62 2A95 C822  B567 02B3

Subkeys:

Key ID         Valid From     Expires        Size           Status

0x74EC64FA     2000-04-20     Never          2048

ADKs: None

Revokers: None
```

If you want to view the fingerprint in words instead of in hexadecimal format, set the `--fingerprint-view` option in the configuration file or on the command line to `words`.

Getting more information about signatures on a key

Use the `--sig-detail` option with the `--signer` modifier to list additional information about the signature.

```
ebs --sig-detail <userID> --signer <userID of signing key>
```

For example, to view the properties of David Gibson's signature on Odette Richards' key, you would enter the following:

```
ebs --sig-detail "Odette Richards" --signer "David Gibson"
```

E-Business Server finds the key on the keyring, and displays information about David's signature.

For information on verifying a signature, see *Verifying a digital signature* on page 60.

For information on signing a key, see *Signing a key* on page 63.

Removing keys from your keyring

You can delete someone's public key from your public keyring, or delete your own key pair—the public portion from your public keyring and the private portion from your secret keyring. These options are described in the following sections.

For information on adding keys to your keyring, see *Adding a key to your keyring* on page 35.

Removing a public key from your keyring

You can delete someone's public key from your keyring with the `--key-remove` option. If you want to delete several keys with the same user ID, include the `--multi` modifier. You are prompted to confirm the deletion of each key with the specified user ID. If you want E-Business Server to automatically delete all matching keys without first asking for confirmation, include the `--force` modifier as well.

If you specify a user ID that matches more than one key, and did not specify `--multi` on the command line, an error message appears.

To remove someone's public key from your keyring:

1 Enter the following command at the command line:

```
ebs --key-remove <userID or keyID>
```

For example, suppose you want to remove Gina Marala's key from your public keyring.

```
ebs --key-remove "Gina Marala"
```

E-Business Server prompts you to confirm you want to remove the key.

2 Enter `y` to delete the key.

E-Business Server removes the key from your keyring.

Removing a key pair from your keyring

If the key that you want to delete is part of your key pair, add the `--with-private` modifier to the `--key-remove` option.

To remove a key pair from your keyring:

1 Enter the following command at the command line:

```
ebs --key-remove <userID or keyID> --with-private
```

For example, suppose you want to remove a test *key pair* from your keyring.

```
ebs --key-remove "Test 1" --with-private
```

E-Business Server prompts you to confirm you want to remove the private key.

2 Enter `y` to delete the key.

E-Business Server deletes the private portion from the private keyring as well as the public portion from the public keyring.

Removing all keys from your smart card

The smart card must be in your smart card reader. Removing your smart card from the smart card reader while wiping it, may result in unpredictable behavior.

Before you generate a E-Business Server key pair on a smart card, you should specify the smart card type in the configuration file, `pgp.cfg`.

If you do not specify the smart card type in the configuration file using the `SMARTCARD-TYPE` parameter, then you must add the `--smartcard-type` modifier to the command line each time you perform an operation on a smart card. For more information, see *SMARTCARD-TYPE* on page 137.

To wipe the contents of your smart card, use the `--smartcard` modifier with the `--wipe` option. This deletes all keys and data from your smart card. Optionally, you can specify your smart card PIN number on the command line.

```
ebs --wipe --smartcard [--pin <smart card PIN>]
```

For information on how to view the contents of your smart card, see *Viewing your keys* on page 39. For more information on various ways to supply E-Business Server with your passphrase, see *Alternative ways to work with passphrases* on page 93.

Removing a specific key pair from your smart card

The smart card must be in your smart card reader.

To delete specific key pair from your smart card, use the `--smartcard` modifier with the `--key-remove` option. This deletes only the matching keys from your smart card. Optionally, you can specify your smart card PIN number on the command line.

```
ebs --key-remove <userID1> <userID2> <userID3>... --smartcard
[--pin <smart card PIN>]
```

For information on how to view the contents of your smart card, see *Viewing your keys* on page 39. For more information on various ways to supply E-Business Server with your passphrase, see *Alternative ways to work with passphrases* on page 93.

Verifying the contents of your public keyring

E-Business Server automatically checks any new keys or signatures on your public keyring and updates all the trust parameters and validity scores. In theory, it keeps all the key validity status information up-to-date as material is added to or deleted from your public keyring.

At some point, however, you may want to explicitly force E-Business Server to perform a comprehensive analysis of your public keyring, checking all the certifying signatures, checking the trust parameters, updating all the validity scores, and checking your own ultimately-trusted key against a backup copy on a write-protected floppy disk. It may be a good idea to do this hygienic maintenance periodically to make sure nothing is wrong with your public keyring.

To force E-Business Server to perform a full analysis of your public keyring, use the `--key-check` command:

```
ebs --key-check
```

You can also use the following command to make E-Business Server check all the signatures for a single selected public key:

```
ebs --key-check <userID> [<keyring_filename>]
```

Updating keys on your keyring

As you add to or change information on your key pair, it is recommended that you send your updated key to a key server so that your most current key is always available to others.

Likewise, to ensure that you are always using the most current keys belonging to other E-Business Server users, you should periodically update your local keyring with the keys on a key server. To do so, enter the following at the command line:

```
ebs --key-update [--keyserver <url>]
```

E-Business Server searches the specified key server or generic LDAP server for all keys on your local keyring and merges the matching keys back into your keyring.

By default, E-Business Server searches the key server specified by the `KEYSERVER` parameter in the E-Business Server configuration file (see *KEYSERVER* on page 128).

To update additional key information, add various modifiers to the `--key-update` command using the following syntax:

```
ebs --key-update [--adk | --keys | --revokers | --introducers | --x509
| --crl]
```

A description of each of these modifiers is listed below:

`--adk` updates and adds Additional Decryption Keys (ADKs) associated with a key on your keyring. If **ADK-KEY** is set in the E-Business Server configuration file (see *ADK-KEY* on page 111), then that key is also updated or added to your local keyring.

`--keys` specifies that all keys on your keyring are updated from the key server. This is the default operation if no modifiers are supplied.

`--revokers` specifies that all designated revokers associated with keys on your keyring are also updated from the key server. If a designated revoker's key is not currently on your keyring, E-Business Server adds it from the key server.

`--introducers` specifies that E-Business Server updates or adds introducer keys to your keyring for all keys with meta-introducer signatures on them.

E-Business Server searches your local keyring for keys with valid meta-introducer signatures. Then, E-Business Server searches the key server for all keys signed by this set of introducer keys and all matching keys are added to your keyring.

For example, if CorpKey signs WestCoastKey and EastCoastKey as meta-introducers and you currently only have CorpKey and WestCoastKey on your local keyring, then E-Business Server adds EastCoastKey to your keyring from the key server. Thus, any company keys certified by the EastCoast certifier will be trusted by E-Business Server.

`--x509` specifies that all keys with x.509 signature certificates associated with them are also updated on your keyring. This ensures that any revocations from key servers are merged into the key.

`--crl` fetches a certificate revocation list from a Certificate Authority and applies the revocations to keys on the keyring.

Editing your key

When you think of key management, you probably think of maintenance, such as updates or changes to your key.

For example, you may need to change your passphrase, perhaps because someone looked over your shoulder while you typed it on the keyboard. You may need to change your user ID, because you changed your name or your email address. You may need to add a second or third user ID to your key, because you are known by more than one name, email address, or job title.

You may also need to make an existing key your default signing key. In order to do so, you must set the `DEFAULT-KEY` parameter in your E-Business Server configuration file, `pgp.cfg`. For more information, see *DEFAULT-KEY* on page 120.

Key editing tasks you might perform on your keys include those listed in the following table.

To:	See:
Add and remove user IDs	*Adding and removing user IDs* on page 45
Set your primary user ID	*Setting your primary user ID* on page 46
Change your passphrase	*Changing your passphrase* on page 46
Change the trust setting on a key	*Editing trust options for your key* on page 46
Remove a signature from your key	*Remove a signature from a key* on page 47
Add a designated revoker to your key	*Adding a designated revoker to your key* on page 47
Add and remove photo IDs	*Adding and removing photo IDs* on page 48
Revoke a key	*Revoking a key* on page 49
Disable and enable a key	*Disabling and enabling a key* on page 50

Adding and removing user IDs

You can add and remove additional user IDs associated with your key. E-Business Server actually adds a new user ID, without deleting the old one. If you want to delete an old user ID, you must do that in a separate operation.

The key you are editing must be your own user ID, which E-Business Server knows is yours because it appears on both your public keyring and your secret keyring.

To add a new user ID:

1 Enter the following on the command line:

```
ebs --key-edit <userID> --add-userid <new userID>
```

E-Business Server prompts for your passphrase.

2 Type your passphrase, and hit `Enter`.

E-Business Server adds the new user ID to your key.

For information on displaying keys with their additional user IDs, see *Display all user IDs associated with each key* on page 39.

To remove a user ID:

1 Enter the following on the command line:

```
ebs --key-edit <userID> --remove-userid <userID to remove>
```

E-Business Server removes the user ID from your key.

For information on displaying keys with their additional user IDs, see *Display all user IDs associated with each key* on page 39.

Setting your primary user ID

Occasionally, you may need to change your primary user ID (the default user ID); perhaps because you changed your name or your preferred email address.

To set your default user ID:

1 Enter the following on the command line:

```
ebs --key-edit <userID> --set-primary-userid <new primary userID>
```

E-Business Server prompts for your passphrase.

2 Type your passphrase, and hit Enter.

E-Business Server sets the requested user ID as your primary user ID.

For information on displaying keys with their additional user IDs, see *Display all user IDs associated with each key* on page 39.

Changing your passphrase

Your security is only as good as your passphrase. If you feel that your passphrase has been compromised, then you should change your passphrase immediately.

To change your passphrase:

1 Enter the following on the command line:

```
ebs --key-edit <userID or keyID> --change-passphrase
```

E-Business Server prompts for your current passphrase.

2 Enter your current passphrase to gain access to the key, and hit Enter.

E-Business Server prompts for your new passphrase.

3 Enter your new passphrase.

4 Enter your new passphrase again for confirmation.

E-Business Server changes your passphrase.

Editing trust options for your key

Use the `--key-edit` option to edit trust options for a key on your keyring. You can turn off "implicit trust" for your own key pair, or you can edit a public key on your keyring, designating someone as a trusted introducer.

If you designate someone a *trusted introducer*, then all keys validated by the trusted introducer are considered to be valid to you.

This means that if you ever get a key from someone that has been signed by an individual whom you have designated as trustworthy, the key is considered valid even though you have not done the check yourself.

A key must be valid in order for you to edit it's level of trust. For more information on signing someone's key, see *Validity and trust* on page 61.

To edit trust options for your key:

1 Enter the following on the command line:

```
ebs --key-edit <userID> --trust <level>
```

Your trust level options are:

- Enter `none`, if you do not know if you trust the owner of this key to act as a trusted introducer, or if you do not trust the owner of this key.
- Enter `marginal`, if you usually trust the owner of this key to act as a trusted introducer.
- Enter `complete`, if you always trust the owner of this key to act as a trusted introducer.
- Enter `implicit`, if the key is your own key.

To view a key's trust level, use the `--key-detail` option. For more information, see *Getting more information about keys* on page 40.

Remove a signature from a key

Use the `--remove-sig` modifier with the `--key-edit` option to remove signatures from a local copy of your key. Bear in mind, however, that if others have signed a copy of your key that is residing on a public key server, the signatures will reappear on your key when you synchronize your key with the one on the key server.

To remove selected signatures from a user ID on a key:

1 Enter the following on the command line:

```
ebs --key-edit <userID of key being edited> --remove-sig <userID or
keyID of signature to remove>
```

For example, the following would delete The Joker's signature from Adam West's key:

```
ebs --key-edit "Adam West" --remove-sig "The Joker"
```

Adding a designated revoker to your key

It is possible that you might forget your passphrase someday or lose your private key. If this happens, then you would be unable to use your key again, and you would have no way of revoking it to show others not to encrypt to it. To safeguard against this possibility, you can appoint a third-party key revoker. The third-party you designate is then able to revoke your key just as if you had revoked it yourself. For more information on revoking keys, see *Revoking a key* on page 49.

For a key to appear revoked to another user, both the revoked key and the Designated Revoker key must be on his/her keyring. Thus, the designated revoker feature is most effective in a corporate setting, where all users' keyrings contain the company's Designated Revoker key. If the revoker's key is not present on a person's keyring, then the revoked key does not appear revoked to that user and he/she may continue to encrypt to it.

This feature is available for Diffie-Hellman/DSS and RSA keys. Designated revokers are not supported by RSA Legacy keys.

To add a designated revoker to your key:

1 Ensure that the designated revoker's key is on your keyring.

2 Enter the following on the command line:

```
ebs --key-edit <your userID> --add-revoker <revoker's userID or keyID>
```

3 Enter your current passphrase to gain access to the key.

E-Business Server adds the designated revoker to your key.

Adding and removing photo IDs

You can attach a JPEG file containing a photo to your key. The photo must be in JPEG format. Any size image is allowed, but images larger than 120 pixels wide by 144 pixels tall are shrunk down to this size when displayed. Also, the larger the JPEG image, the larger the key will be.

E-Business Server does not directly support viewing a photo ID, but you can view photo IDs when using other versions of E-Business Server or you can export the photo ID to a file and use a third-party JPEG viewer.

If a photo ID already exists on the key, then you must remove it before adding a new photo ID. See *To remove a photo ID from your key:* on page 49 for instructions.

To add a photo ID to your key:

1 Enter the following on the command line:

```
ebs --key-edit <your userID> --add-photoid <filename>
```

For example, you might enter:

```
ebs --key-edit wilma@mcafee.com --add-photoid /pictures/wilma.jpg
```

2 Enter your current passphrase to gain access to the key.

E-Business Server adds the JPEG-formatted photo to your key. To see if the photo is present, check the key's details. For more information, see *Getting more information about keys* on page 40.

You can also see if the photo is present by using the `--extract-photoid` option and viewing the resulting file.

To remove a photo ID from your key:

If you have more than one photo on your key (which is not supported by current versions of E-Business Server), E-Business Server removes the most recently added photo.

1 Enter the following on the command line:

```
ebs --key-edit <your userID> --remove-photoid
```

E-Business Server removes the JPEG-formatted photo from your key. To confirm that the photo is no longer present, check the key's details.

For more information, see *Getting more information about keys* on page 40.

To extract a photo ID to a file:

1 Enter the following on the command line:

```
ebs --extract-photoid <userID> --output <filename>
```

For example:

```
ebs --extract-photoid jbond --output james.jpg
```

E-Business Server extracts the photo ID belonging to `jbond` and copies it to the `james.jpg` file. If the output file already exists, then E-Business Server prompts you for confirmation to overwrite it.

Revoking a key

By revoking a key, you are telling people that the key should no longer be used. You should revoke a public key if you think that its corresponding private key has been compromised.

To revoke a key, you need the private portion of the key and the passphrase. As a safeguard against the possibility of forgetting your passphrase or losing your private key, you might want to specify a designated revoker. For more information on specifying a designated revoker, see *Adding a designated revoker to your key* on page 47.

You cannot restore a revoked key.

To revoke a key:

1 Ensure that the public portion of the key being revoked is on your keyring.

2 Enter the following on the command line:

```
ebs --key-edit <userID of key being revoked> --revoke
```

3 Enter your current passphrase to gain access to the key.

E-Business Server revokes the key. The best way to circulate a revoked key is to place it on a public key server.

Disabling and enabling a key

Sometimes you may want to temporarily disable a key. The ability to disable keys is useful when you want to retain a public key for future use, but you don't want it in your way when you perform encryption operations. Use the `--key-details` option to see if your key is disabled (see *Getting more information about keys* on page 40 for more information).

To disable a key:

1 Enter the following on the command line:

```
ebs --key-edit <userID of key to disable> --disable
```

E-Business Server temporarily disables your key.

To enable a key:

1 Enter the following on the command line:

```
ebs --key-edit <userID of key to re-enable> --enable
```

E-Business Server re-enables your key.

Splitting and rejoining a key

Any private key can be split into shares among multiple "shareholders" using a cryptographic process known as Blakely-Shamir key splitting. This technique is recommended for extremely high security keys.

Creating a split key

To create a split key, you are asked to specify the minimum number of people required to rejoin the key and the number of shares to make.

The resulting shares are saved as files either encrypted to the public key of a shareholder or encrypted conventionally if the shareholder has no public key. After the key has been split, the share files must be sent to the shareholders via ftp or email.

Attempts to sign or decrypt with a split key will automatically cause E-Business Server to temporarily rejoin the key.

To create a split key:

1 Enter the following on the command line:

```
ebs --key-split <userID of key to split>
```

2 Enter the user ID of the first shareholder.

3 Enter the number of shares for this user, or accept the default (one share).

4 Repeat steps 2 and 3 until you have specified all the shareholders, then hit `Return`.

5 Enter the minimum number of shares needed to rejoin the key, known as the *threshold*, or accept the default value.

6 Enter the passphrase of the key you are splitting.

E-Business Server splits the key into the number of shares specified, then encrypts each portion of the key to the specified shareholder. Each shareholder receives a share file (in `.shf` format) for every share he/she owns.

For example, if you were to split a company signing key between 3 shareholders (`Amy`, `Peter Wallings`, and `Jamal@mcafee.com`) where `Amy` gets 2 shares, and the other recipients each get 1 share, the resulting share filenames would be:

```
Amy2 Shares.shf

Peter Wallings1 Share.shf

Jamal@mcafee.com1 Share.shf
```

In this example, the following format is used:

```
<userID><number_of_shares> <Share.shf or Shares.shf>
```

When you want to rejoin the split key, you must enter the complete share filename, including the user ID and number of shares. Please note that there is no space between the user ID and the number of shares, but there is a space before the .shf file. Also, if a user has more than one share file, then the filename is plural—Shares.shf instead of Share.shf.

To verify the key has been split, use the `--key-list` option. The key displays `"Splt"` in the `Type` column.

Reconstituting a split key

Once a key is split among multiple shareholders, attempting to sign or decrypt with it will cause E-Business Server to automatically attempt to rejoin the key. There are two ways to rejoin the key, *locally* and *remotely*.

To rejoin key shares locally requires the shareholders presence at the rejoining computer. Each shareholder is required to enter the passphrase for his/her key share.

To rejoin key shares remotely requires the remote shareholders to authenticate and decrypt their keys before sending them over the network. E-Business Server's Transport Layer Security (TLS) provides a secure link to transmit key shares which allows multiple individuals in distant locations to securely sign or decrypt with his/her key share.

Before receiving key shares over the network, you should verify each shareholder's fingerprint and sign his/her public key to ensure that the authenticating key is legitimate. To learn how to verify a key pair, see *Validity and trust* on page 61.

Reconstituting a split key locally

To reconstitute a split key locally requires the shareholder's presence at the rejoining computer. Each shareholder must enter his/her own passphrase to decrypt the share file encrypted to his/her key.

To join a key locally:

1. Enter the following on the command line:

   ```
   ebs --key-join <userID of key to join>
   ```

2. Enter the complete share filename belonging to the first shareholder.

   For example, you might enter `Amy2 Shares.shf` from the previous example in *Creating a split key* on page 50.

3. Enter the passphrase belonging to this shareholder.

   The portion of the split key encrypted to this shareholder decrypts.

   E-Business Server displays the number of valid shares from each shareholder, as well as the minimum number of shares needed to rejoin the key.

4. Repeat steps 2 and 3 until the minimum number of shares needed to rejoin the key is decrypted.

   The key is rejoined.

5. Enter a new passphrase for the key.

6. Enter the new passphrase again for confirmation.

Reconstituting a split key over the network

To reconstitute a split key over the network, you use the `--key-join` option. Once you have created a split key, you must send the shares to the shareholders in other locations. You can do this by ftp or via email.

You must have a signing key on your keyring to set up a TLS connection, which provides a secure link to transmit the key shares securely to individuals in other locations. This key is authenticated by the remote machine to establish its trust of your identity. Likewise, the remote machine presents its key so that you can authenticate the identity of the remote user. You must establish mutual validity for these keys.

You can specify the signing key you want to use for the TLS connection using the `--auth-user` option, or you can let E-Business Server choose a signing key on your keyring for you.

Joining a key over the network

To join a key over the network, you must perform actions on the system that contains the split key *and* on each remote system.

On the system that contains the split key

1. Enter the following at the command line:

   ```
   ebs --key-join <userID of key to join> [--auth-user <userID>]
   ```

2. Press Enter.
3. E-Business Server chooses a signing key on your keyring to set up a TLS connection (unless you specified a key with `--auth-user`).
4. Enter the passphrase for this key.
5. The system opens a TLS connection and waits to receive the shares.

   The system displays, "`Listening...`"

At each remote site:

1. Enter the following on the command line:

   ```
   ebs --send-shares <quoted_share_filename> [--auth-user <userID>]
   ```

2. Enter the IP address for the remote system.
3. Enter your passphrase to decrypt the share.

   The system displays, "`Preparing to send the key share.`"

   E-Business Server chooses a signing key on your keyring to authenticate the TLS connection (unless you specified a key with `--auth-user`).
4. Enter the passphrase for this key.
5. When prompted, type `Y` to confirm the connection.

On the system that contains the split key

1. When prompted, type `Y` to confirm the connection.

Once the connection is confirmed on both ends, the share is sent securely over the network and received by the system containing the split key. When the minimum number of shares needed to rejoin the key is received, then the key is rejoined.

2. Enter a new passphrase for the key.

Additional Decryption Keys

Suppose your chief scientist is hit by a bus and is hospitalized for months. Or that your lead engineer, in a rage, encrypts his entire hard drive and leaves the company. What happens to all that data, which is so securely encrypted? Can you retrieve it, or is it gone forever?

An *Additional Decryption Key* (ADK) is a data recovery tool. In an environment that enforces use of an ADK, any information encrypted to a user's key is also encrypted to the Additional Decryption Key. When someone inside or outside the organization encrypts information to a user, the information is also encrypted to the Additional Decryption Key. This allows the owner of the Additional Decryption Key to decrypt any information sent to the user. This process happens automatically and is fully integrated into the encryption process.

Recover data in an emergency

An ADK is a powerful security tool in situations where an employee is injured, incapacitated, or terminated, leaving valuable information encrypted. Because E-Business Server has no "back door," recovery of this information would be otherwise infeasible.

While you may not ordinarily use your ADKs, there may be circumstances when it is necessary to recover someone's data, for example, if someone is out of work for some time or if you are subpoenaed by a law enforcement agency and must decrypt messages or files for a court case.

Data recovery versus key recovery

Do not confuse data recovery with key recovery. An Additional Decryption Key lets you recover information that has been encrypted to a particular key, not the key itself. The difference is crucial. If a mechanism exists to obtain a copy of a user's key, one major feature of a public-key cryptosystem—non-repudiation—is lost. If more than one copy of a key exists, then a user can deny having signed information with the key.

Retaining copies of users' keys has an added security risk: the machine storing the keys is an obvious target for attack, as is the administrator of the machine.

An Additional Decryption Key is far easier to protect, and it enables you to retain non-repudiation, which is a major advantage inherent to public-key cryptography.

Types of ADKs

E-Business Server offers two types of ADKs: Incoming ADKS, and Outgoing ADKs.

- An *incoming* ADK is used by E-Business Server during key generation. An incoming ADK's key ID is associated with new keys during key generation, and henceforth when someone attempts to encrypt to the new key, E-Business Server also attempts to encrypt to the ADK. Incoming ADKs may be either Diffie-Hellman/DSS or RSA keys. You cannot use an RSA Legacy key as an incoming ADK.
- An *outgoing* ADK is associated with an installation of E-Business Server. Outgoing ADKs are automatically added to users' keyrings and are always part of a recipient list. Outgoing ADKs can be of any key type.

Additional Decryption Key policy

Your Security Officer must decide whether your company enforces the use of ADKs. You should have a policy that governs how and when they will be used and should communicate this policy to everyone who will be affected by it. This policy should consider employee privacy as well as the security of the ADK against attack and accidental loss.

Protecting your Additional Decryption Key

Additional Decryption Keys must be secured both physically and electronically in order to prevent a security breach. If either the incoming or outgoing ADK is ever compromised, all encrypted messages sent to users with additional decryption enabled could be decrypted by the attacker.

To prevent unauthorized additional decryption and problems with liability, your organization should enforce a policy that the key should be split and shared by two or more individuals. Consider employee turnover and ensure that ADK's are properly rejoined and re-split as necessary to retain the ADK's integrity.

Do *not* use ADKs unless you can ensure their security. In an environment that enforces use of an ADK, security of these keys determines the security of all encrypted messages in your entire organization.

Implementing your Additional Decryption Keys

To implement ADKs in your environment, you must first create the ADK(s).

If you want separate keys for the incoming ADK and the outgoing ADK, your key generation process must explicitly specify a different ADK key (incoming) than the one specified in the configuration file (outgoing).

1 Do one of the following:

- Set the incoming/outgoing ADK by specifying the `ADK-KEY` parameter in the configuration file, and then generate a key (the `--key-gen` option) that meets your needs in terms of key type and key size.

or

- Enter the following on the command line to specify an incoming ADK:

```
ebs --key-gen --adk-key <keyID>
```

2 If you want to enforce use of the ADK, set the `ENFORCE-ADK` parameter to `ON`.

For more information on setting the `ADK-KEY` configuration parameter, see *ADK-KEY* on page 111. For more information on setting the `ENFORCE-ADK` configuration parameter, see *ENFORCE-ADK* on page 122.

Deleting your key from a key server

You can remove your key from a key server so that it is no longer available to others. To remove your key from a server, you use the `--keyserver-delete` option.

In order to delete a key from a key server, you need to specify a signing key to digitally sign the deletion request. Depending on the key server, you may also need a signing key to set up a secure TLS connection between the client and the server.

By default, E-Business Server uses the key specified by the `DEFAULT-KEY` parameter in the E-Business Server configuration file (see *DEFAULT-KEY* on page 120). If you do not want to use the default signing key, then you can specify the signing key by adding the `--sign-with` modifier on the command line.

Additionally, you can specify the passphrase for your signing key by including the `--passphrase` modifier.

To delete your key from a key server:

1. Enter the following command at the command line:

   ```
   ebs --keyserver-delete <keyID or userID> [--keyserver
   <keyserver_URL>]
   ```

   For example:

   ```
   ebs --keyserver-delete "John Lee"
   ```

   E-Business Server displays the key information and prompts you for confirmation on the deletion.

2. Enter `y` to delete the key.

3. Enter the passphrase for your secret key.

   E-Business Server includes your signature in a request that it sends to the key server to delete the key. If you are not authorized by the key server to delete the specified key or if you can't delete keys from the machine you are using, then an error message appears.

Reconstructing your key

If you ever lose your private key or you forget your passphrase, there is no way to recover from it unless you set up a key reconstruction policy, which includes setting up a key reconstruction server.

You would have provided recovery information—five questions and five secret answers—and would have sent your key to the key reconstruction server. To learn how to send your key to the reconstruction server, see *To send your key to your company's key reconstruction server:* on page 32.

If you sent your key to a reconstruction server, you can restore your key pair at any time as long as you have your public key and can answer at least three of the five questions you created.

To reconstruct your key from your company's reconstruction server

1 Enter the following on the command line:

```
ebs --key-reconstruct <userID or keyID>
```

Where `<userID>` or `<keyID>` is the ID belonging to the key you want to reconstruct.

2 Enter the name of the server that stores the reconstruction data using the following format:

```
ldap://<IP address or DNS name of server>
```

For example:

```
ldap://keyserver.mycorp.com
```

3 As each of the prompts (questions) appear, type the corresponding answer. Keep in mind that your answers are case sensitive. You must be able to answer at least three questions to restore your key.

Once you've answered all of the questions, E-Business Server prompts for a new passphrase.

4 Enter a new string of characters or words you want to use as the new passphrase for your key pair, then hit Enter.

Your passphrase should contain multiple words and may include spaces, numbers, and punctuation characters. Choose something that you can remember easily but that others won't be able to guess. The passphrase is case sensitive, meaning that it distinguishes between uppercase and lowercase letters. The longer your passphrase, and the greater the variety of characters it contains, the more secure it is. Strong passphrases include upper and lowercase letters, numbers, punctuation, and spaces but are more likely to be forgotten. See *Creating a passphrase that you will remember* on page 30, for more information about choosing a passphrase.

5 To confirm your entry, type the same passphrase again.

Your key pair is reconstructed. For information on how to view your keys, see *Viewing your keys* on page 39.

6 Working with Digital Signatures

For an overview of digital signatures, validation, trust, and the other concepts in this chapter, as well as a description of how E-Business Server performs such tasks, see *An Introduction to Cryptography.*

Signing information

To sign a plaintext file using your default private key, use the `--sign` option. If you do not specify another key (using the `--sign-with` modifier), E-Business Server uses your default key. (Your default key is specified using the `DEFAULT-KEY` parameter in the E-Business Server configuration file. For more information, see *DEFAULT-KEY* on page 120.)

```
ebs --sign <plaintext_filename>
```

You must supply the passphrase for the private key.

Unencrypted signed messages have a signature certificate prepended in binary form. The signed message is compressed, rendering the message unreadable to human eyes, even though the message is not encrypted. The following is an example of an unencrypted signed message:

```
-----BEGIN EBS MESSAGE-----
Version: EBS 8.x

owHrZLBnZmWwLJntk/hadk01T+xqQSahWwzzY67c+23aMIvPrqNLedIezbfJDPNr2
H8dcjW5FPnMeKXn4+063rt2JpqvZZRLYilJLS6RYGBgCMlIVSgszUzOyi/PU0jLr1
DIKs0tSE1RyC9LLVIoAcrnJFZVKqTkp+txjQzVAAkQKf

-----END EBS MESSAGE-----
```

Producing a clear-signed message

To produce a clear-signed message, one that can be read with human eyes, and without the aid of E-Business Server, the `CLEARSIG` parameter must be set to `on` (the default) in the E-Business Server configuration file, and it must be used in conjunction with the `ARMOR` and `TEXTMODE` parameters. Set `ARMOR=ON` (or use the `--armor` modifier), and set `TEXTMODE=ON` (or use the `--text` modifier).

For example, you would enter the following on the command line (assuming that `CLEARSIG=off` in the configuration file):

```
ebs --sign <plaintext_filename> --clearsig
```

The following is an example of a clear-signed message:

```
-----BEGIN EBS SIGNED MESSAGE-----
Hash: SHA1

The quick brown fox jumped over the lazy dog.
The quick brown fox jumped over the lazy dog.
The quick brown fox jumped over the lazy dog.
The quick brown fox jumped over the lazy dog.
The quick brown fox jumped over the lazy dog.

-----BEGIN EBS SIGNATURE-----
Version: EBS 8.x

owHrZLBnZmWwLJntk/hadk01T+xqQSahWwzzY67c+23aMIvPrqNLedIezbfJDPNr2
H8dcjW5FPnM
=vZZRL

-----END EBS SIGNATURE-----
```

Note that the recipient must still use E-Business Server to verify the signature. For more information on verifying signatures, see *Verifying a digital signature* on page 60. For more information on using the `CLEARSIG` parameter, see *CLEARSIG* on page 118.

Signing with a specific private key

If you have more than one private key on your private keyring, E-Business Server automatically uses the default key (specified using the `DEFAULT-KEY` parameter in `ebs.cfg`) to sign your messages. To sign using a private key that is not your default private key, you must specify a different key using the `--sign-with` modifier.

```
ebs --sign <textfile> --sign-with <userID>
```

You must supply the passphrase for the private key.

Signing and encrypting

To sign a plaintext file with your secret key and encrypt it with the recipient's public key in a single operation, you combine the `--encrypt` option with the `--sign` option. You can optionally specify which private key to use to sign the file.

```
ebs --encrypt <plaintext filename> --user <recipient's_userID>
--sign [--sign-with <your_userID>]
```

Signing and encrypting a plaintext ASCII text file

Do not use `--text` with binary data, such as a spreadsheet or word processing file, because the binary file format will be altered, making the file unusable.

To sign a plaintext ASCII text file with your secret key, producing a signed plaintext message suitable for distribution through channels such as email, use the `--text` modifier. To encrypt and sign a plaintext ASCII text file, producing a message suitable for sending through email, use the following syntax:

```
ebs --encrypt <plaintext ASCII text filename> --user
<recipient's_userID> --text --sign [--sign-with <your_userID>]
```

For example, if Cee Wong wants to encrypt `secretfile.txt` to Sean Adams and sign it with her private key, she would enter the following:

```
ebs --encrypt secretfile.txt --user "Sean Adams" --text --sign
[--sign-with "Cee Wong"]
```

The encrypted and signed file can then be sent through email. The following is an example of an encrypted and signed message:

```
-----BEGIN EBS MESSAGE-----
Version: EBS 8.x

aMIvPrqNLedIezbfJDPNr2H8dcjW5FPnMeKXn4+063rt2JpqvZZRLYilJLS6RYGBg
CMlIVSgszUzOyi/PU0jLr1DIKs0tSE1RyC9LLVIoAcrnJFZVKqTkp+txjQzVAAkQK
fowHrZLBnZmWwLJntk/hadk01T+xqQSahWwozHrZLBnZmWwLJntk/hadk01T+xqQS
ahWwzzY67c+23aMIvPrqNLedIezbfJDPNr2H8dcjW5FPnMeKXn4+063rt2JpqvZZR
LYilJLS6RYGBgCMlIVSgszUzOyi/PU0jLr1DIKs0tSE1RyC9LLVIoAcrnJFZVKqTk
p+txjQzVAAkQKfowHrZLBnZmWwLJntk/hadk01T+xqQSahWw+xqQSahWwozHrZLBn
ZmWwLJntk/hadk01T+xqQSahWwzzY67c+23aMIvPrqNLedIezbfJDPNr2H8dcjW5F
PnMeKXn4+063rt2JpqvZZRLYilJLS6RYGBgCMlIVSgszUzOyi/PU0jLr1DIKs0tSE
1RyC9LLVI==
=kggl

-----END EBS MESSAGE-----
```

Creating a detached signature

In most cases, signature certificates are physically attached to the text they sign. This makes it convenient to verify signatures. You can, however, create a separate, detached signature, and then send both files (the text file and the signature certificate file) to the recipient. This feature is useful when more than one party must sign a document such as a legal contract, without nesting signatures. Each person's signature is independent.

To create a separate, detached signature certificate file, combine the `--detached` modifier with the `--sign` option. You can optionally specify which private key to use to sign the file.

```
ebs --sign <plaintext_filename> --detached [--output <filename>]
[--sign-with <your_userid>]
```

For example:

```
ebs --sign letter.txt --detached
```

This instructs E-Business Server to produce a separate, detached signature certificate in a file named `letter.txt.sig`. The contents of `letter.txt.sig` are not appended to `letter.txt`.

Verifying a digital signature

To determine whether an attached digital signature is valid, you *verify* it. E-Business Server automatically verifies signatures as part of the decryption operation. If you want to verify a file, use the same syntax as that for decryption:

```
ebs <filename>
```

Verifying a detached signature

When you attempt to process a detached signature certificate file, E-Business Server asks you to identify the corresponding text file. Once the text file is identified, E-Business Server checks the signature integrity.

If you know that a signature is detached from a text file, you can specify both filenames on the command line:

```
ebs <signature_filename.sig> <textfile.txt>
```

For example:

```
ebs letter.txt.sig letter.txt
```

If the text file exists in the same directory as the detached signature certificate file, you can enter the following shortened command:

```
ebs letter.txt.sig
```

E-Business Server assumes that the signed text has the same name as the signature (`.sig`) file—if it does not, then you must specify the filename.

Storing signed files: signing a file without encrypting

If you sign a plaintext file without specifying encryption, E-Business Server compresses the file after you sign it. This makes the file unreadable to the casual human observer. This is a suitable way to store signed files in archival applications because it saves space. However, it is not an especially secure means for storing the data.

Validity and trust

Every user in a public key system is vulnerable to mistaking a phony key (certificate) for a real one. *Validity* is confidence that a public key certificate belongs to its purported owner. Validity is essential in a public key environment where you must constantly establish whether or not a particular certificate is authentic.

When you've assured yourself that a key belonging to someone else is valid, you can sign the copy on your keyring to attest to the fact that you've checked the key and that it's an authentic one. If you want others to know that you gave the key your stamp of approval, you can export the signature to a certificate server so that others can see it.

Checking a key's validity

Viewing signatures on a key

To view the signatures on a key use the `--key-list` option with the `--with-sigs` modifier. E-Business Server displays all the keys on your keyring and then, for each key, the signatures on the key. It also displays the level of trust you have in each key and its validity, and verifies the signatures.

```
ebs --key-list --with-sigs
```

E-Business Server lists the keys on your keyring with the signatures for each key. The signatures are represented with "`sig`" in the `Type` column.

For more information on the variations of the `--key-list` option, see *Viewing your keys* on page 39.

Getting more information about signatures on a key

You may want to display information about a signature on a key, such as the signature's creation date or expiration date. Use the `--sig-detail` option with the `--signer` modifier to list information about a signature on a key.

```
ebs --sig-detail <userID> --signer <userID of signing key>
```

For example, if there is a signature belonging to David Gibson on Odette Richards key, then I can view information about David's signature by entering the following command:

```
ebs --sig-detail "Odette Richards" --signer "David Gibson"
```

E-Business Server displays information about David's signature.

Viewing a key's fingerprint

You can check that a certificate is valid by calling the key's owner (so that you originate the transaction) and asking the owner to read his or her key's fingerprint to you and verifying that fingerprint against the one you believe to be the real one.

To do so, both you and the key's owner use the `--key-detail` option to view the key's fingerprint:

```
ebs --key-detail <userID> [--fingerprint-view hex|words]
```

This command instructs E-Business Server to display the key with the 40 character digest of the public key components (RSA Legacy keys have 32 character fingerprints). Read the fingerprint to the key's owner to see if the fingerprints match.

Using this procedure, you can verify and sign each other's keys with confidence. This is a safe and convenient way to get the key trust network started for your circle of friends.

Note that sending a key fingerprint via email is not the best way to verify the key because email can be intercepted and modified. It is best to use a different channel than the one that was used to send the key itself. A good combination is to send the key via email, and verify the key fingerprint via a voice telephone conversation. Some people even distribute their key fingerprint on their business cards.

The default format of a fingerprint view is a hexadecimal display. If you would prefer to display the fingerprint as a word list, set the `--fingerprint-view` option to `words`. This can also be set in the E-Business Server configuration file.

The word list is made up of special authentication words that E-Business Server uses and are carefully selected to be phonetically distinct and easy to understand without phonetic ambiguity.

The word list serves a similar purpose as the military alphabet, which allows pilots to convey information distinctly over a noisy radio channel. If you'd like to know more about the word hash technique and view the word list, see *Biometric Word Lists*

Granting trust for key validations

Trust is confidence in another person's ability to validate a key. If you designate someone a *trusted introducer*, then all keys validated by the trusted introducer are considered to be valid to you.

This means that if you ever get a key from someone that has been signed by an individual whom you have designated as trustworthy, the key is considered valid even though you have not done the check yourself.

Changing your trust settings on a key

To edit the trust parameters for a particular key on your keyring (that is, to designate someone a trusted introducer), you use the `--key-edit` option.

```
ebs --key-edit <userID> --trust <level>
```

Your trust options are:

- Enter `none`, if you do not know if you trust the owner of this key to act as a trusted introducer, or if you do not trust the owner of this key.
- Enter `marginal`, if you usually trust the owner of this key to act as a trusted introducer.
- Enter `complete`, if you always trust the owner of this key to act as a trusted introducer.
- Enter `implicit`, if the key is your own key.

For more information on editing your key and key management, see *Editing your key* on page 44.

Signing a key

For information on creating an X.509 certificate signature, see *General X.509 certificate attributes* on page 212.

To sign and validate someone else's public key on your public keyring, use the `--key-sign` option. When you sign a key it is automatically considered valid to you.

Be absolutely certain that the key belongs to its purported owner before you sign it!

Your default signing key is used to sign the key, unless you add the `--sign-with` modifier to the command line. You may also specify the passphrase for your signing key by using the `--passphrase` modifier.

E-Business Server uses the signature type specified by the `SIG-TYPE` parameter in the E-Business Server configuration file. If you want to specify a different signature type when signing a key, see the instructions outlined in *Specifying the type of signature you want to add to a key* on page 64.

During the key signing process, you are given the option to attach a regular expression to your signature. The purpose of which is to restrict the scope of the target key's signature power. For more information about regular expressions, see *Attaching Regular Expressions to Signatures*.

The key you are signing must be present on your keyring.

To sign a key:

1 Enter the following at the command line:

```
ebs --key-sign <recipient's_userID> [--sign-with <your_userID>]
[--passphrase <quoted-passphrase>]
```

For example,

```
ebs --key-sign "Earle Rice" --sign-with "Tim Ryans"
```

2 Enter the passphrase for your secret key.

E-Business Server prompts for a confirmation that you are certain that the key belongs to the user specified.

3 Enter `y` if you are certain of the key's owner, and want to sign the key.

E-Business Server asks if you want to attach a regular expression to your signature.

4 Enter the regular expression you want to attach to the signature, or press `Enter` if you do not want to attach a regular expression.

E-Business Server signs the specified key with your signing key.

Specifying the type of signature you want to add to a key

Use the `--sig-type` modifier with the `--key-sign` option to specify the type of signature you want to attach to the key you are signing.

```
ebs --key-sign <recipient's_userID> --sig-type <type>
```

Where `<type>` is one of the following:

- `Local`. Local signatures are non-exportable. This signature type remains on your local keyring.
- `Exportable`. Exportable signatures, the default option, are attached to the key certifying that you verify the key owner's identity. This signature type can be exported to a file or key server so other users can view them.
- `Meta`. Meta signatures (always non-exportable) bestow meta-introducer status on the key. Any key considered trusted by the meta-introducer is considered a trusted introducer by you, and any key considered valid by the trusted introducer is considered valid to you.
- `Introducer`. Introducer signatures bestow trusted introducer status on the key. Any key considered valid by a trusted introducer is considered valid to you.

You can add a regular expression to an 'introducer' type signature by including `--regexp <expression>` on the command line. This restricts the scope of the target key's signature power. For more information about regular expressions, see *Attaching Regular Expressions to Signatures*.

Adding an expiration date to your signature

Use the `--expires-after` modifier with the `--key-sign` option to append an expiration date to your signature on the recipient's key. Set `<expiration>` equal to the number of days until your signature expires or a future date you want your signature to expire in YYYY-MM-DD format. To set the signature to never expire, set `<expiration>` equal to zero. To add an expiration date to your signature, use the following syntax:

```
ebs --key-sign <recipient's_userID> --expires-after <expiration>
```

For example, if you want to sign Gilbert Sampson's key and have your signature expire in two weeks (or 14 days), you would enter:

```
ebs --key-sign "gilbert sampson" --expires-after 14
```

If you do not want your signature to expire, then you would enter:

```
ebs --key-sign "gilbert sampson" --expires-after 0
```

If you want your signature to expire on July 16, 2003, then you would enter:

```
ebs --key-sign "gilbert sampson" --expires-after 2003-07-16
```

Removing signatures from your key

Use the `--remove-sig` modifier with the `--key-edit` option to remove signatures from a local copy of your key. Bear in mind, however, that if others have signed a copy of your key that is residing on a public key server, the signatures will reappear on your key when you synchronize your key with the one on the key server. For more information on the `--key-edit` option, see *Editing your key* on page 44.

To remove selected signatures from a user ID on a key:

```
ebs --key-edit <userID of key being edited> --remove-sig <userID or
keyID of signature to remove>
```

For example:

```
ebs --key-edit "Batman" --remove-sig "The Joker"
```

In the above example, E-Business Server is instructed to delete The Joker's signature from Batman's key.

7 Working with X.509 Certificates

This chapter describes how to add, export, and create X.509 digital certificates, electronic documents used to prove identity and public key ownership over a communication network.

To:	See:
Understand common X.509 options	*Common X.509 options* on page 66
Add an X.509 certificate to your key	*Adding an X.509 certificate to your key or keyring* on page 68
Request and add an X.509 certificate from a CA	*Getting an X.509 certificate from a CA* on page 69
Export an X.509 certificate from your key	*Exporting an X.509 certificate from your key* on page 73
Issue X.509 certificates	*Issuing X.509 certificates* on page 73
Update X.509 certificates on your keyring	*Updating X.509 certificates on your keyring* on page 76

Common X.509 options

When requesting, exporting, or creating X.509 certificates, you may need to specify information such as the issuer's distinguished name, the issuer assigned serial number or certificate attributes. The following sections explain how to specify this information.

Specifying a certificate with the issuer's name and serial number

One key can contain several certificates, and the certificates can all be from the same issuer; therefore, in order to uniquely identify a certificate, you may need to specify two pieces of information: the issuer's distinguished name (DN) and the issuer assigned serial number on the certificate. This combination is always unique. Specify the issuer's DN using the `--issuer-dn` modifier. Specify the serial number assigned by the issuer using the `--issuer-serial` modifier.

You can find the issuer's DN and the certificate's serial number using the `--sig-detail` option.

```
ebs --sig-detail <keyID>
```

For example, if there's an X.509 certificate attached to a key belonging to Scott Tibson, key ID 0x196DE730, then you would enter the following to get more information about the certificate:

```
ebs --sig-detail 0x196DE730
```

The following information appears:

```
   Signed Key    : Scott Tibson <scott_tibson@mcafee.com>

   Signed User ID: Scott Tibson <scott_tibson@mcafee.com>

   Signed Key ID : 0x196DE730 (0xFBC4D3B5196DE730)

Name:              CN=Scott Tibson, EMAIL=scott_tibson@mcafee.com,
                   O=McAfee, OU=EBS

Issuer:            CN=Root CA, EMAIL=admin@mcafee.com

Signer Key ID:     0xD7C74275 (0x03534DC9D7C74275)

 Type:             X.509

Exportable:        Yes

Created:           2001-06-01

Expires:           2002-06-01

Last CRL:          N/A

Next CRL:          N/A

Trust Depth:       0

   Serial Number:

     9170E2A076CF0C8B4938
```

The issuer's DN

The issuer assigned serial number

Specifying certificate attributes

When you request an X.509 certificate from a public Certificate Authority (CA), or when you issue an X.509 certificate using E-Business Server, you can include certificate attributes, additional bits of information about the certificate that may be added to the certificate as per the CA's certification policies.

To add certificate attributes to the certificate you are requesting or creating, include the `--cert-attribute` modifier. Valid X.509 attributes include—but are not limited to—the email address of the certificate holder (E), the name of the company to which the certificate belongs (O), the unit or group within the company to which the certificate belongs (OU), and the location of the company to which the certificate belongs (L).

Certificate attributes are entered in *name=value* format. *Name* represents the type of attribute you want to define such as E, O, OU, or L. You can enter the complete attribute name (as one word, without any spaces) or the abbreviated version of the attribute name. *Value* represents your definition for the corresponding attribute. If the value contains spaces, then you must enclose it in quotes. For example, O="McAfee" indicates that the organization that owns the certificate is McAfee. You can list several certificate attributes when requesting or creating X.509 certificates. Simply precede each name=value pair with `--cert-attribute`.

The attributes used on certificates is a policy decision of the CA. Typically, the following attributes are used for X.509 certificates.

Attribute Name:	Description:
CN (Common Name)	Often a description of the type of certificate (e.g., "Root").
E (EMAIL)	The email address for the certificate holder.
O (Organization)	Typically the name of the company to which the certificate belongs (e.g.,"Secure Company").
OU (Organizational Unit)	The department or group within the company to which the certificate belongs (e.g.,"Accounting").
L (Locality)	The location of the holder of the certificate (e.g., "Santa Clara").
STREET	The street address of the holder of the certificate.
ST (State)	The state of the holder of the certificate (e.g., "CA").
POBOX	The PO box or postal code of the holder of the certificate.
C (Country)	The country of the holder of the certificate (e.g., "USA").
DN (Distinguished Name)	Typically the distinguished name of the company to which the certificate belongs.

For a complete list of the certificate attributes that E-Business Server supports, including a list of Verisign-specific attributes, see *Supported Certificate Attributes*

Adding an X.509 certificate to your key or keyring

You may need to add an X.509 certificate to your keyring, such as a Root CA's certificate, or manually add an X.509 certificate to your key pair from a file. To do either, use the `--key-add` option with the `--x509` modifier.

```
ebs --key-add <filename> --x509
```

Where `<filename>` is the name of the file containing the certificate you want to add.

E-Business Server supports importing of PEM, DER (PKCS #7) and PKCS #12 formatted certificates. If the file extension is `.pem`, E-Business Server assumes the certificate is PEM-encoded. If the file extension is `.p12` or `.pfx`, E-Business Server assumes the certificate is PKCS #12 formatted.

For example, if you enter the following command, E-Business Server automatically knows to add a PEM-encoded certificate:

```
ebs --key-add cert.pem --x509
```

If the binary file is PKCS #12, then you must include the `--with-private` modifier (as shown below).

```
ebs --key-add <filename> --x509 --with-private
```

This forces the PKCS #12 import format of the X.509 certificate and includes the private portion of your key pair.

When you add or change information in your key pair, always update it on the key server so that your most current key can be available to anyone. See *Adding your key to a key server* on page 36 for instructions.

Getting an X.509 certificate from a CA

You can request an X.509 digital certificate and add it to your key pair using E-Business Server options and your company's *Certificate Authority* (CA) or a public CA (for example, VeriSign). There are two main methods for requesting and adding X.509 certificates to your keys—automatically and manually. Both methods are described in the following sections.

For either method you must first obtain and add the Root CA certificate from the Certificate Authority and add it to your keyring. For instructions, see *Retrieving and adding the Root CA certificate to your keyring* on page 69.

Automatically requesting and adding an X.509 certificate to your key

The instructions in this section describe how to add an X.509 certificate to your key pair if you are using the Net Tools PKI Server. The process and terminology may vary between Certificate Authorities and some of the certificate attributes and certification procedures (identity-checks) you must use when interacting with your CA is a policy decision. You may need to consult the administrator of your Certificate Authority for instructions.

There are four main steps to automatically requesting and adding an X.509 certificate to your key pair:

1. Retrieve the Root CA certificate from the CA and add it to your keyring (see *Retrieving and adding the Root CA certificate to your keyring*).
2. Enter information about the CA in the E-Business Server configuration file (see *Specifying CA parameters in the E-Business Server configuration file* on page 70).
3. Request a certificate from the CA. Your X.509 certificate request is verified and signed by the CA (see *Automatically requesting a certificate from the CA* on page 70). (The CA's signature on the certificate makes it possible to detect any subsequent tampering with the identifying information or the public key, and it implies that the CA considers the information in the certificate valid.)
4. Retrieve the certificate issued by the CA and add it to your key pair (see *Retrieve your certificate and add it to your key pair* on page 71).

Retrieving and adding the Root CA certificate to your keyring

Whether you are automatically requesting or manually requesting X.509 certificates to add to your keys, you must first obtain and add the Root CA certificate to your keyring. The only exception is if you are requesting a PKCS #10 certificate.

When you add or change information in your key pair, always update it on the key server so that your most current key can be available to anyone. See *Adding your key to a key server* on page 36 for instructions.

To retrieve and add the Root CA certificate to your keyring:

1 Open your Web browser and connect to the CA's enrollment site.

2 Locate and examine the Root CA certificate. This process varies between Certificate Authorities. For example, if your company were using the Net Tools PKI Server, you would click the **Download a CA Certificate link**, and then click the **Examine this Certificate** button.

3 Copy the key block for the Root CA certificate and paste it into a file.

4 Add the Root CA certificate to your keyring. See *Adding an X.509 certificate to your key or keyring* on page 68 for instructions.

Specifying CA parameters in the E-Business Server configuration file

Specify the CA's URL using the `CA-URL` parameter. This URL must be fully qualified. For example, you might enter something like https://myca.ebs.com:444 (this is the same URL you used to retrieve the Root CA).

If there is a separate URL for retrieving certificate revocation lists (CRLs), specify it using the `CA-REVOCATION-URL` parameter. If you do not know the URL for revocation, leave this option blank.

Specify the name of certificate authority you are using with the `CA-TYPE` option. Your choices are:

- `nettools` (Net Tools PKI)
- `verisign` (VeriSign OnSite)
- `entrust` (Entrust)
- `iplanet` (iPlanet CMS)
- `win2k` (Windows 2000)

Specify the Root CA certificate you retrieved earlier using the `CA-ROOT-CERT` parameter. For more information on retrieving the Root CA certificate, see *Retrieving and adding the Root CA certificate to your keyring* on page 69.

Automatically requesting a certificate from the CA

After specifying CA parameters in the configuration file (see *Specifying CA parameters in the E-Business Server configuration file* on page 70), use the `--cert-request` option to automatically request a certificate from the CA.

You can request that specific attributes be added to your new certificate using the `--cert-attribute` option. It is up to the CA's discretion whether or not they include these attributes. For more information on adding certificate attributes, see *Specifying certificate attributes on page 67*.

To request a certificate:

1 Enter the following on the command line:

```
ebs --cert-request <keyID> [--cert-attribute <name=value>]
```

2 Enter the passphrase for your key pair.

The certificate request is sent to the CA server. The server authenticates itself to your computer and accepts your request.

In a corporate setting, your company's administrator verifies your information in the request. The identifying information and public key are assembled and then digitally signed with the CA's own certificate to create your new certificate.

The administrator sends you an email message stating that your certificate is ready for retrieval.

Retrieve your certificate and add it to your key pair

Use the `--cert-retrieve` option to get your certificate from the CA and automatically add it to your key pair.

To retrieve your certificate:

1 Enter the following on the command line:

```
ebs --cert-retrieve <keyID>
```

E-Business Server contacts the CA server and automatically retrieves your new X.509 certificate and adds it to your key.

You can verify that the certificate has been added by using the `--sig-details` option.

Manually requesting and adding an X.509 certificate to your key

The process for manually requesting and adding an X.509 certificate to your key is similar to the automated process. The difference being that the certificate you request must be in PKCS #10 format, and when the certificate is ready for retrieval, you manually copy and import the key block into your key.

There are four main steps to manually requesting and adding an X.509 certificate to your key pair:

1 Retrieve the Root CA certificate from the CA and add it to your keyring (see *Retrieving and adding the Root CA certificate to your keyring* on page 69).

2 Create a PKCS #10 formatted certificate request (see *Creating a PKCS #10 certificate request* on page 71).

3 Deliver your certificate request to the CA (see *Sending your certificate request to the CA* on page 72).

4 Manually retrieve the certificate issued by the CA and add the key block to your key (see *Manually retrieve your certificate and add it to your key pair* on page 72).

Creating a PKCS #10 certificate request

Use the `--cert-request` option with the `--pkcs10` modifier to create a certificate-request file in PKCS #10 format.

Optionally, you can add certificate attributes using the `--cert-attribute` modifier to your certificate request. For more information on how to specify a certificate attribute, see *Specifying certificate attributes on page 67*.

The Root CA key is not required for a PKCS #10 certificate request.

To create a PKCS #10 certificate request:

1 Enter the following on the command line:

```
ebs --cert-request <keyID> --pkcs10 --output <filename>
[--cert-attribute <name=value>]
```

2 Enter the passphrase for your key pair.

E-Business Server creates a PEM-encoded block of text representing your certificate request. Once you have created a PKCS #10 certificate request, you must send your request to the CA. For instructions, see *Sending your certificate request to the CA* on page 72.

Sending your certificate request to the CA

When manually requesting an X.509 certificate, you must deliver your certificate request to the Certificate Authority.

Copy your certificate request, the PEM-encoded block of text, and send it to your Certificate Authority. Typically, you can send this via email or copy it directly to the CA's web-site. This process varies between Certificate Authorities.

Manually retrieve your certificate and add it to your key pair

You need to manually copy the key block representing your new certificate and add it to your key pair.

To retrieve your certificate and add it to your key pair:

1 Go to the CA's web-site and copy the key block for your X.509 certificate and paste it into a file.

2 Add the X.509 certificate to your key by entering the following on the command line:

```
ebs --key-add <filename> --x509
```

Where `<filename>` is the name of the file where you copied the key block.

The X.509 digital certificate is added to the key you specified when you created the certificate request.

You can verify that the certificate has been added by using the `--sig-details` option.

Exporting an X.509 certificate from your key

You can export (copy) an X.509 certificate associated with your key to a file. The certificate you want to export must be uniquely identified using the `--issuer-dn` and `--issuer-serial` modifiers on the command line. For more information on identifying the certificate you want to use, see *Specifying a certificate with the issuer's name and serial number* on page 66.

EBS supports exporting PEM, DER (PKCS #7) and PKCS #12 formatted certificates.

By default, the key is exported in binary (DER) format. If you add the `--armor` modifier, PEM-encoded format is used instead. A `.crt` extension is added to the filename (both with and without `--armor` specified).

Include the `--with-private` modifier to include the private portion of the key pair you are exporting, so that you can use your key to certify your web browser to remote servers. Both the key pair and the certificate are exported in PKCS #12 format. E-Business Server uses a `.pfx` extension on the output filename making it easy to import the certificate into Internet Explorer.

The PKCS #12 certificate format encodes a decrypted version of a private key. We recommend using extreme caution when exporting or using PKCS #12 export format.

To export an X.509 certificate:

Use the following syntax:

```
ebs --key-export <userID> --x509 [--with-private] [--issuer-dn <DN>
[--issuer-serial <number>] [--output <filename>]
```

For example:

```
ebs --key-export "Cheri Walton" --x509 --with-private --issuer-dn
"CN=Root CA, EMAIL=cheri@mcafee.com, OU=EBS, O=MCAFEE"
--issuer-serial 2840D4A097CF3E1B4016 --output X509cert
```

E-Business Server copies the certificate that matches both the issuer's DN and the serial number specified to the file `X509cert.pfx`.

Issuing X.509 certificates

You can use E-Business Server to issue X.509 signature certificates to other users—acting as your own mini-Certificate Authority (CA).

You may want to do the following:

1. Create a key to use for the sole purpose of issuing X.509 certificates (see *Create a new key for issuing X.509 certificates* on page 74).
2. Create a Root CA certificate (see *Create a Root CA certificate* on page 74).
3. Issue X.509 certificates to others by signing with the root certificate (see *Sign public keys with the root certificate* on page 75).

Create a new key for issuing X.509 certificates

Generate a new RSA or RSA Legacy key to use for the purpose of issuing X.509 certificates. For more information, see *Creating a key pair* on page 26.

You cannot create X.509 certificate signatures with a Diffie-Hellman/DSS key. Using an RSA key, you can certify any type of key.

Create a Root CA certificate

When creating a Root CA certificate, the key you are signing must be the same key you are signing with. This results in a self-signed X.509 certificate (root certificate) where the user-dn is the same as the issuer-dn.

Once the Root CA certificate is created, it can be used to create other signing certificates. This is especially useful for corporations. For example, you can create a single root certificate (called "Root CA Certificate") for the company, and then use it to sign keys with certificates for departmental CA's (such as HR CA or Engineering CA). Finally, the departmental CA's could create certificates for end-users.

Use the `--key-sign --x509` option to create an X.509 certificate instead of a regular signature.

Optionally, you can add certificate attributes using the `--cert-attribute` modifier to the new certificate you are creating. For more information on how to specify a certificate attribute, see *Specifying certificate attributes on page 67*.

By default, an X.509 certificate is valid for one year from the certificate's creation date. When issuing an X.509 certificate, use the `--start-date` modifier to specify a future date as the beginning of the validity period and use the `--expires-after` modifier to specify the number of days you want the certificate to remain valid or a future date when the certificate will expire. When specifying a start or end date, enter the date in YYYY-MM-DD format.

To create a root certificate:

Enter the following:

```
ebs --key-sign <keyID> --x509 [--sign-with <keyID>]
```

For example, if you create a key with the user ID "Root CA <admin@mcafee.com>" and its key ID is "0xD7C74275", then you would enter the following:

```
ebs --key-sign 0xD7C74275 --x509 --sign-with 0xD7C74275
```

You can find out more about the X.509 signature using the `--sig-detail` option. For the above example, you would enter the following:

```
ebs --sig-detail 0xD7C74275
```

The following information appears:

```
Signed Key    : Root CA <admin@mcafee.com>

Signed User ID: Root CA <admin@mcafee.com>
```

```
  Signed Key ID : 0xD7C74275 (0x03534DC9D7C74275)

Name:              CN=Root CA, EMAIL=admin@mcafee.com

Issuer:            CN=Root CA, EMAIL=admin@mcafee.com

Signer Key ID:     0xD7C74275 (0x03534DC9D7C74275)

 Type:             X.509

Exportable:        Yes

Created:           2001-06-01

Expires:           2002-06-01

Last CRL:          N/A

Next CRL:          N/A

Trust Depth:       0

  Serial Number:

    B1B869D7A9A5F08E4EA8
```

Sign public keys with the root certificate

Now that you have a Root CA certificate, you can use it to sign other keys and issue X.509 certificates.

By default, E-Business Server uses the key specified by the `DEFAULT-KEY` parameter in the E-Business Server configuration file as the signing key. If you want to specify a different key for signing the new certificate, use the `--sign-with` option.

If the key you are signing the new certificate with has multiple certificates attached to it, then you must also include the `--issuer-dn` and `--issuer-serial` options to uniquely identify the issuing certificate. You do not need to supply these options if there is only one certificate on the signing key. See *Specifying a certificate with the issuer's name and serial number* on page 66 for more information.

Optionally, you can add certificate attributes using the `--cert-attribute` modifier to the new certificate you are creating. For more information on how to specify a certificate attribute, see *Specifying certificate attributes on page 67*.

By default, an X.509 certificate is valid for one year from the certificate's creation date. When issuing an X.509 certificate, use the `--start-date` modifier to specify a future date as the beginning of the validity period and use the `--expires-after` modifier to specify the number of days you want the certificate to remain valid or a future date when the certificate will expire. Enter the start and end dates in YYYY-MM-DD format.

To create an X.509 certificate:

Use the following syntax:

```
ebs --key-sign <keyID_of_key_to_sign> --x509 [--sign-with <keyID>]
[--issuer-dn <DN> [--issuer-serial <number>]] [--cert-attribute
<name=value>] [--start-date <date>] [--expires-after <expiration>]
```

For example, if the Root CA certificate (OxD7C74275) created in the previous section (see *Create a Root CA certificate* on page 74) is used to add an X.509 signature to the key belonging to Scott Tibson (0x196DE730), then we might enter the following:

```
ebs --key-sign 0x196DE730 --x509 --sign-with 0xD7C74275 --issuer-dn
"CN=Root CA, EMAIL=admin@mcafee.com, OU=EBS, O=MCAFEE"
--issuer-serial B1B869D7A9A5F08E4EA8 --cert-attribute
E=scott_tibson@mcafee.com --cert-attribute O="McAfee"
--cert-attribute OU=EBS
```

An X.509 certificate is created and added to Scott Tibson's key.

You can view the X.509 signature by entering the following:

```
ebs --sig-detail 0x196DE730
```

The following information appears:

```
Signed Key    : Scott Tibson <scott_tibson@mcafee.com>

Signed User ID: Scott Tibson <scott_tibson@mcafee.com>

Signed Key ID : 0x196DE730 (0xFBC4D3B5196DE730)

Name:              CN=Scott Tibson, EMAIL=scott_tibson@mcafee.com,
                   O=McAfee, OU=EBS

Issuer:            CN=Root CA, EMAIL=admin@mcafee.com

Signer Key ID:     0xD7C74275 (0x03534DC9D7C74275)

 Type:             X.509

Exportable:        Yes

Created:           2001-06-01

Expires:           2002-06-01

Last CRL:          N/A

Next CRL:          N/A

Serial Number:     9170E2A076CF0C8B4938

Trust Depth:       0
```

Updating X.509 certificates on your keyring

To ensure that you are always using the most current keys belonging to other users, you should periodically update your local keyring with the keys on a key server. You can specify that all keys with X.509 signature certificates associated with them are also updated on your keyring.

The `CA-URL`, `CA-ROOT-CERT` and `CA-TYPE` parameters must be set in the E-Business Server configuration file, or on the command line, in order to update X.509 certificates on your keyring. For instructions on how to set these parameters, see *Configuration parameters* on page 111.

Once the above parameters are set, enter the following at the command line:

```
ebs --key-update --x509
```

E-Business Server searches the key server for all keys on your local keyring and merges the matching keys back into your keyring. This ensures that any revocations from key servers are merged into the key.

For details on how to update other key information, see *Updating keys on your keyring* on page 43.

8 Encrypting and Decrypting

Exchanging encrypted information

This chapter describes the various E-Business Server options that let you encrypt and decrypt your data.

For an overview of encryption and decryption and a description of how E-Business Server performs the two operations, see *An Introduction to Cryptography.*

Getting the recipient's public key

Before you encrypt, you need to be sure that you are encrypting with the correct public key. This means you need to check the public key to ensure that it truly belongs to the person to whom you think it belongs. Encrypting the message with the wrong key basically makes it:

- Closed to your intended recipient
- Open to whomever's key you encrypted it to (possibly an interloper)

Verifying that a key belongs to its purported owner is discussed in the section "Validity and trust" on page 61.

Encrypting information

Encryption is one of the most common operations you will perform with E-Business Serfsver. You can encrypt using any of the following methods: conventional encryption, public key encryption, create a self-decrypting archive (SDA), or create a PGParchive.

Encrypting with conventional encryption

Encrypting with *conventional* encryption means encrypting to a particular passphrase instead of to a public key. Conventional encryption is useful in certain situations, like when you're encrypting to yourself; however, the typical problem one encounters with conventional encryption is the difficulty in securely communicating the passphrase to the recipient.

For more information on conventional encryption, see *An Introduction to Cryptography.*

```
ebs --encrypt --conventional <filename>
```

To specify a passphrase for conventional encryption, use the `--conventional-passphrase` modifier. If the passphrase contains spaces, you must enclose the entire string in quotes.

```
ebs --encrypt --conventional <filename> [--conventional-passphrase
<quoted-passphrase>]
```

If the file you want to encrypt is not in the current directory, then you must also specify the path to the file.

The following command encrypts the file `secretdocument.txt` using the passphrase `quick, get a mango`. To decrypt the file, the recipient will have to type in the same passphrase.

```
ebs --encrypt --conventional secretdocument.txt
--conventional-passphrase "quick, get a mango"
```

This results in an encrypted file named `secretdocument.txt.pgp`.

Exercise caution when using the `--passphrase` or `--conventional-passphrase` modifiers. Whenever you enter a passphrase as cleartext on the command line (as in the example above), you risk its interception. For alternative ways to supply E-Business Server with your passphrase, see *Alternative ways to work with passphrases* on page 93.

Encrypting with public key encryption

Encrypting with *public key* encryption means encrypting to a user's public key. To encrypt information using public key encryption, you use the `--encrypt --user` option. (The key to which you want to encrypt must be on your keyring.)

It is recommended that you reference the key ID instead of the user ID during encryption if you have very similar user IDs or subkeys on your keyring. For example, if you want to encrypt to user ID "smith@mcafee.com" and you also have a user ID of "jsmith@mcafee.com" on your keyring, then E-Business Server will encrypt to both users. To ensure that you only encrypt to the intended recipient, reference the key ID of the key to which you want to encrypt.

```
ebs --encrypt <filename> --user <recipient's_userID>
```

For example, to encrypt the file `testresults.doc` to Jennifer Quino's key, you would use the following syntax:

```
ebs --encrypt testresults.doc --user "Jennifer Quino"
```

Encrypting into ASCII-armored format

Typically, you use the `--encrypt` option in conjunction with other options; the `--encrypt --armor` option encrypts information into ASCII-armored format, which is suitable for sending through email channels. The following example would put the file `testresults.doc` in a format appropriate for sending via email:

```
ebs --encrypt --armor testresults.doc --user "Jennifer Quino"
```

This results in an encrypted file called `testresults.doc.asc`.

For more information on working with ASCII and binary data, see *Working with ASCII and binary data* on page 90.

Encrypting a text file

The `--encrypt --text` option tells E-Business Server that you are encrypting a text file and preserves its text format for decryption on other platforms.

Do not use `--text` with binary data, such as a spreadsheet or word processing file. The binary file format will change making it unusable.

The following example would put the file `testresults.txt` in a format appropriate for sending via email:

```
ebs --encrypt --armor --text testresults.txt --user "Jennifer
Quino"
```

This results in an encrypted file named `testresults.txt.asc`.

Encrypting and specifying the output file

To specify an output filename use the `--encrypt --output` option.

Please note the following:

- If *standard input* is used for reading data to encrypt, then the output is written to *standard output*.
- You can also specify '`--output -`' to encrypt to standard output.
- If no output file is specified for a single input file, E-Business Server creates a file with a `.pgp` or `.asc` extension in the current directory.
- If more than one file is used as input, then the output files are written to the current directory or the directory specified by `--output`.
- If more than one file is used as input, and no output file is specified, then an appropriate filename is used for each file and they are all written to the current directory.

If any output filenames already exist in the output directory, you are prompted for confirmation that you want to replace the existing file. If you do not want to be prompted, and instead want E-Business Server to automatically overwrite the existing file, use the `--overwrite` option.

The following example would encrypt the file `testresults.doc` to the key belonging to Jennifer Quino and filter the encrypted file to an application that reads *standard input*.

```
ebs --encrypt testresults.doc --user "Jennifer Quino" --output -
```

Encrypting to multiple recipients

To encrypt to several recipients at once, you can use multiple `--user` modifiers and specify the user IDs, as shown in the following syntax:

```
ebs --encrypt <filename> --user <userID1> --user <userID2> --user
<userID3>...
```

For example, suppose you want to encrypt meeting minutes to three coworkers in a format you can send in a text email.

```
ebs --encrypt --armor --text mtgminutes.txt --user "Carol Wong"
--user "Angie Vicari" --user "Kevin Sprole"
```

You can also create a *group*, which functions much like a mailing or distribution list functions in most email programs. For information on working with groups, see the section, "Working with groups" on page 96.

Encrypting multiple files to one recipient

To encrypt several files to a single recipient in one operation, you can manually specify the filenames as shown in the following syntax.

```
ebs --encrypt <filename1> <filename2> <filename3>... --user
<userID>
```

For example, suppose you want to encrypt the following status reports to your manager in a format you can send via email.

```
ebs --encrypt --armor status040601.doc status041301.doc
status042001.doc --user "Carol Wong"
```

All of the files are encrypted to Carol Wong's key.

Encrypting information to a group

To encrypt information to a predefined group of recipients, you specify the group name as you would a single recipient's name.

```
ebs --encrypt <filename> --user <groupname>
```

For information on managing groups, see "Working with groups" on page 96.

Automatically encrypting to your own key

The configuration parameter `ENCRYPT-TO-SELF` enables you to automatically encrypt everything to your own key or some other predefined key in addition to any specified recipients.

To set this up, you must set parameters in E-Business Server's configuration file, `pgp.cfg`:

- Set the `DEFAULT-KEY` parameter to the key ID of the desired key (see *DEFAULT-KEY* on page 120)
- Set `ENRYPT-TO-SELF=on` in the configuration file (see *ENCRYPT-TO-SELF* on page 122) or use `--encrypt-to-self` on the command line

Encrypting for viewing by recipient only

To specify that the recipient's decrypted plaintext be shown only on the recipient's screen and not saved to disk, add the `--secure-viewer` modifier to the `--encrypt` option:

```
ebs --encrypt --text <message.txt> --secure-viewer --user
<recipient's_userID>
```

The `--secure-viewer` option is only supported on files with a size of 500K or less.

The recipient decrypts the ciphertext with their secret key and passphrase; the plaintext is displayed on the recipient's screen but is not saved to disk. The text is displayed as it would if the recipient used the UNIX "more" command, one screen at a time. If the recipient wants to read the message again, he or she must decrypt the ciphertext a second time.

This feature is the safest way for you to prevent your sensitive message from being inadvertently left on the recipient's disk.

This feature does not prevent a clever and determined person from finding a way to save the decrypted plaintext to disk—it is designed to help prevent a casual user from doing it inadvertently.

Encrypting and signing

To digitally sign a file and encrypt it in the same operation, add the `--sign` option to the command line. E-Business Server signs the message before encrypting it. The signing key specified by the `DEFAULT-KEY` parameter in the E-Business Server configuration file is used unless you specify a different signing key with the `--sign-with` modifier.

```
ebs --encrypt --sign <plaintext_filename> --user
<recipient's_userID> [--sign-with <your_userID>]
```

For example:

```
ebs --encrypt --sign testresults.doc --user "Jennifer Quino"
--sign-with "Anita Brown"
```

Encrypting and wiping the original plaintext file

To wipe the original plaintext file, overwriting and deleting it completely, then add the `--wipe` option to the encryption operation.

```
ebs --encrypt --wipe <plaintext_filename> --user
<recipient's_userID>
```

For example:

```
ebs --encrypt --wipe confidential.txt --user mjohnson
```

This instructs E-Business Server to create a ciphertext file `confidential.pgp` and to destroy the plaintext file `confidential.txt`.

Note that this option will not wipe out any fragments of plaintext that your word processor might have created on the disk while you were editing the message before running E-Business Server. Most word processors create backup files, scratch files, or both.

By default, E-Business Server overwrites the file three times. If you want to specify the number of times E-Business Server overwrites the file, and not use the default, add the `--wipe-passes` modifier.

```
ebs --encrypt --wipe <plaintext_filename> --user
<recipient's_userID> --wipe-passes <number of times>
```

Creating Self-Decrypting Archives (SDAs)

You can use E-Business Server to create a self-decrypting executable file, which is conventionally encrypted using a passphrase that you are asked to choose. The resulting executable file can be decrypted by simply double-clicking on it and entering the appropriate passphrase. This option is especially convenient for users who are sending encrypted files to people who do not have E-Business Server software installed.

The final SDA cannot be larger than 4 GB in size. The input may be larger than 4 GB if E-Business Server can compress it to a size smaller than 4 GB.

While you can create SDAs on any platform E-Business Server supports, SDAs will execute only on Windows platforms—Windows 95, 98, NT, and 2000. See *Creating PGParchives* on page 84 for a cross-platform alternative.

To create an SDA, you simply add the `--sda` (self-decrypting archive) modifier to the `--encrypt` option. You must also specify the name of the input file.

```
ebs --encrypt --sda <filename>
```

Optionally, you can specify more than one input file, the name of the output file, and the passphrase you want to use for the conventional encryption.

```
ebs --encrypt --sda <filename1> <filename2>... [--output
<output_filename>] [--conventional-passphrase <quoted-passphrase>]
```

You can specify directories as input instead of filenames—the syntax would be `ebs --encrypt --sda <directory1> directory2>...`

The default name for the SDA is the name of the first file in the SDA. E-Business Server automatically appends a "`.sda.exe`" extension.

For example, to generate a SDA containing the files `c:\data.txt` and `c:\results.txt` you would enter:

```
ebs --encrypt --sda c:\data.txt c:\results.txt --output mysda.exe
```

The resulting SDA file is `mysda.exe`. (If you created the file without specifying the output filename, the resulting file would be `data.txt.sda.exe`.)

You can add the `--discard-paths` option to instruct E-Business Server to strip any relative path information from the list of files you want to include in the SDA. During decryption of the SDA, the files are placed in the current directory instead of in subdirectories of the current directory.

For example:

```
ebs --encrypt --sda --discard-paths foo/bar.txt abc/xyz.txt
```

In this example, E-Business Server includes the files `bar.txt` and `xyz.txt` in the archive, but the file's relative paths are not included. When the archive is decrypted, both files are placed in the current directory and not in `foo` and `abc` subdirectories.

Creating PGParchives

You can use E-Business Server to create a PGParchive. PGParchives are similar to self-decrypting archives (SDAs) except that they are not Windows executables, which means that there is no practical limit to the size of a PGParchive. (SDAs cannot be larger than 4 GB.) PGParchives are usable on all platforms that E-Business Server supports.

If you select this option, the file is conventionally encrypted using a passphrase that you are asked to choose. The resulting output file can be decrypted using the archive reader, which you can re-distribute freely. This option is especially convenient for users who are sending encrypted files to people who do not have E-Business Server software installed.

To create a PGParchive, you simply add the `--archive` modifier to the `--encrypt` option. You must also specify the name of the input file.

```
ebs --encrypt --archive <filename>
```

Optionally, you can specify multiple input files, the name of the output file, and the passphrase you want to use for the conventional encryption.

```
ebs --encrypt --archive <filename1> <filename2>... [--output
<output_filename>] [--conventional-passphrase <quoted-passphrase>]
```

The default name for the PGParchive is the name of the first input file in the archive. E-Business Server automatically appends a "`.pga`" extension.

For example, to generate a PGParchive containing the files `c:\data.txt` and `c:\results.txt` you would enter:

```
ebs --encrypt --archive c:\data.txt c:\results.txt --output
archive.txt
```

The resulting archive file is `archive.txt`. (If you created the file without specifying the output filename, the resulting file would be `data.txt.pga`.)

You can add the `--discard-paths` option to instruct E-Business Server to strip any path information from the list of files you want to include in the PGParchive. During decryption of the PGParchive, the files are placed in the current directory instead of in subdirectories of the current directory.

For example:

```
ebs --encrypt --archive --discard-paths foo/bar.txt abc/xyz.txt
```

In this example, E-Business Server includes the files `bar.txt` and `xyz.txt` in the archive, but the file's relative paths are not included. When the archive is decrypted, both files are placed in the current directory and not in `foo` and `abc` subdirectories.

Decrypting information

You decrypt information using the private portion of your key pair (unless you've encrypted using conventional encryption, in which case you decrypt using the correct passphrase). You can decrypt only that information which is encrypted to the corresponding public portion of your key pair.

Decrypting with E-Business Server is a matter of specifying the encrypted file's name, as shown in the following syntax.

```
ebs --decrypt <ciphertext_filename>
```

You are required to enter a passphrase for your private key or the passphrase used to conventionally encrypt the file.

To specify the passphrase for your private key as part of the operation, use `--passphrase` option as shown in the following syntax:

```
ebs --decrypt <ciphertext_filename> --passphrase
<quoted-passphrase>
```

For example:

```
ebs --decrypt secretdocument.asc --passphrase "quick, get a mango"
```

To specify the passphrase used to conventionally encrypt the file, use the `--conventional-passphrase` option as shown in the following syntax:

```
ebs --decrypt <ciphertext_filename> --conventional-passphrase
<quoted-passphrase>
```

Viewing the decrypted file

When E-Business Server encrypts a plaintext file, it saves the original filename and attaches it to the plaintext before it is compressed and encrypted. When E-Business Server decrypts the ciphertext file, it names the plaintext output file with a name similar to the input ciphertext filename, but drops the extension.

You can specify other output results for the decrypted information as described below.

Viewing decrypted plaintext output on your screen

To view decrypted plaintext output on your screen (similar to the UNIX-style "more" command), without writing the output to a file, use the `--secure-viewer` modifier when you decrypt:

```
ebs --decrypt --secure-viewer <ciphertext_filename>
```

This command instructs E-Business Server to display the decrypted plaintext on your screen, one screen at a time, regardless of whether `--secure-viewer` was used during encryption.

Renaming the decrypted plaintext output file

When E-Business Server decrypts a ciphertext file, it names the plaintext output file with a name similar to the input ciphertext filename, but drops the extension. For example, if you decrypt a file named `foo.txt.pgp`, E-Business Server creates a file named `foo.txt`.

Use the `--output` option on the command line to specify a more meaningful plaintext filename for the output:

```
ebs --decrypt <original_ciphertext_filename> --output
<new_plaintext_filename>
```

If there are multiple input files, the output files are written to the current directory or to the directory name specified by `--output`.

If any output filenames already exist in the output directory, you are prompted for confirmation that you want to replace the existing file. If you do not want to be prompted, and instead want E-Business Server to automatically overwrite the existing file, use the `--overwrite` option.

Recovering the original plaintext filename

As stated in the previous section, when E-Business Server encrypts a plaintext file, it saves the original filename and attaches it to the plaintext before it is compressed and encrypted. Use the `--preserve-name` modifier to instruct E-Business Server to preserve the original plaintext filename and use it as the name of the decrypted plaintext output file.

```
ebs --decrypt --preserve-name <ciphertext_filename>
```

Decrypting SDAs and PGParchives

Self-decrypting archives (SDAs) and PGParchives can be decrypted on Windows and Unix platforms using the archive reader. The PGParchive reader is freely distributable and installed automatically with the product. This is especially useful if you want to send encrypted data to someone not using E-Business Server. Simply send that person the archive reader (`EBSreader.exe`) along with the SDA (`.sda.exe` extension) or the PGParchive (`.pga` extension).

With conventional encryption, you must securely communicate the passphrase to the recipient.

On Windows systems, you can double-click the SDA or the PGParchive and enter the passphrase when prompted. The file is decrypted and placed in the current directory. You can also use the command line syntax as described below.

Optionally, when working on a Windows system, you can add the `--no-gui` modifier to the command line if you do not want E-Business Server to display information in helpful dialog boxes. (This option is simply ignored on Unix.)

On Unix and Windows systems, you can decrypt SDAs and PGParchives using the following syntax:

```
ebsreader <filename> [<target_directory>] [--overwrite] [--no-gui]
```

Where `<filename>` is the SDA or PGParchive created during encryption and `<target_directory>` is the path to the location where you want to save the output.

When you use the archive reader for the first time on a Windows system, E-Business Server searches for the file extension .pga in its list of registered extensions. If the file extension does not exist, or if it is set to something other than the archive reader, E-Business Server prompts you to set a file association with the reader. To do so, select the **Always check for the .pga file association?** check box, and then click **OK**.

When E-Business Server prompts you, enter the passphrase needed to decrypt the file. On Windows, you can browse to the directory where you want to save the output (unless you've specified a different target directory on the command line). On Unix, E-Business Server uses the current directory (unless you've specified a different target directory).

If `--discard-paths` was used during the creation of the PGParchive, then all extracted files are saved to the target directory. If this option was not specified, then the files are still saved to the target directory, but may be in sub-directories corresponding to the original input files.

You must specify `--overwrite` if an output file (typically the same name as the encrypted file minus the .sda.exe or .pga extension) already exists in the directory and you do not want E-Business Server to prompt you for confirmation when overwriting it.

To display the version number for the archive reader you have installed, enter the following on the command line:

```
ebsreader --version
```

The version information appears.

9 Advanced Topics

This chapter describes several advanced E-Business Server topics and commands.

To:	See:
Use scripts with E-Business Server	*Using scripts with E-Business Server* on page 88
Using E-Business Server without interaction	*Using E-Business Server without interaction* on page 88
Encrypt and transmit binary data	*Encrypting and transmitting binary data* on page 90
Send ASCII files to different machine environments	*Working with ASCII and binary data* on page 90
Wipe your disk	*Wiping your disk* on page 92
Wipe your smart card	*Wiping your smart card* on page 93
Work with passphrases	*Alternative ways to work with passphrases* on page 93
Work with groups	*Working with groups* on page 96
Manually start the ebssdkd	*Starting the ebssdkd* on page 98
Keep your keyring files open	*Keeping your keyring files open with EBScache* on page 98

Using scripts with E-Business Server

You can run E-Business Server in "batch" mode (for example, from a Windows "`.bat`" file or from a UNIX shell script).

For example, if you wanted to create a script to encrypt and sign several files inside a loop, then you might include the following syntax:

```
ebs --encrypt --sign $(FILE) --user $(USER) --passphrase
$(QUOTED-PASSPHRASE) --output $(FILE)-$(USER).pgp
```

Using E-Business Server without interaction

To use E-Business Server without interaction you must include all information normally prompted for by E-Business Server using command line switches.

For example, to sign a key and not have E-Business Server prompt for additional information, simply provide all of the necessary options on the command line using the following syntax:

```
ebs --key-sign <keyID> --sign-with <userID> --passphrase
<quoted-passphrase> --expires-after <expiration> --sig-type <type>
--force
```

Therefore, if Andy Tobbs wants to sign Willy Kampton's key with a specific signature type and expiration date, he would enter:

```
ebs --key-sign wkampton@mcafee.com --sign-with "Andy Tobbs"
--passphrase "2BeeRnotTobe" --expires-after 10 --sig-type local
--force
```

E-Business Server has all the information needed to perform the key signing operation, and because `--force` is also included, E-Business Server does not ask for any confirmations.

Using E-Business Server without interaction is especially useful when working with the E-Business Server API. For more information, see *Using the E-Business Server API* on page 100.

Understanding E-Business Server exit status codes

When you run E-Business Server in batch mode, E-Business Server returns an error exit status to the shell.

- A zero exit status code signifies a normal exit.
- A non-zero exit status code tells you that an error occurred. Different error exit conditions return different exit status codes to the shell.

For a list of E-Business Server exit codes, see *Exit and Error Codes*.

Using E-Business Server as a UNIX-style filter

UNIX uses pipes to make two applications work together. The output of one application can be directly fed through a pipe to be read as input to another application. For this to work, the applications must be able to read the raw material from "standard input" and write the output to "standard output."

E-Business Server uses standard input if no input files are specified. E-Business Server uses standard output if standard input was used for input and no output files were specified. You can force standard output by adding '`--output -`' to the command line.

To use E-Business Server's UNIX-style filter mode, reading from standard input and writing to standard output, add the '`--output -`' option and do not specify any input files:

```
ebs --encrypt --armor --user <recipients_userid> --output -
```

For example:

```
ebs --encrypt --armor --user ksmith --output -
```

This feature makes it easier to use E-Business Server with scripts and email applications. When you use E-Business Server's filter mode to decrypt a ciphertext file, you may find the `PGPPASS` environment variable useful. This variable holds the passphrase so that E-Business Server does not prompt you for this information. For information on various ways to specify your passphrase, see *Alternative ways to work with passphrases* on page 93.

Working with ASCII and binary data

ASCII-armored text is binary data that has been encoded using a standard, printable, 7-bit ASCII character set. This allows users to transport the information through many email systems that only allow messages that contain ASCII text.

The sections to follow describe how to:

- encrypt and transmit binary data
- send binary data files in ASCII-armored format without encryption or signature
- decrypt ASCII-armored messages
- send a public key in ASCII-armored format
- send ASCII text files to different machine environments

Encrypting and transmitting binary data

Many email systems only allow messages that contain ASCII text. As a result, E-Business Server supports an ASCII-armored format for ciphertext messages (similar to MIME).

This format, which represents binary data using only printable ASCII characters, enables you to transmit binary encrypted data through 7-bit channels, or to send binary encrypted data as normal email text. E-Business Server's ASCII-armored format acts as a form of "transport armor," protecting the message against corruption as it travels through intersystem gateways on the Internet. E-Business Server also appends a CRC to detect transmission errors.

ASCII-armored format converts the plaintext by expanding groups of 3 binary 8-bit bytes into 4 printable ASCII characters. As a result, the file expands by about 33%. However, this expansion is offset by the compression that occurs before encryption.

To produce an ASCII-armored formatted file, enter the following command:

```
ebs --encrypt --armor <plaintext_filename> --user
<recipient's_userID>
```

This command instructs E-Business Server to produce a ciphertext file in ASCII-armored format called `.asc`. This file contains data in a MIME-like ASCII-armored format. You can upload the file into a text editor through 7-bit channels and transmit as normal email.

Sending binary data files in ASCII-armored format without encryption or signature

Use E-Business Server's `--armor` option to convert a file into ASCII-armored format. No encryption or signing is involved, so neither sender nor recipient requires a key. When you use the `--armor` option, E-Business Server attempts to compress the data before converting it to ASCII-armored format. Use the command as follows:

```
ebs --armor <binary_filename>
```

This command instructs E-Business Server to produce an ASCII-armored file called `filename.asc`. The recipient uses the `--preserve-name` option to unwrap the message and restore the sender's original filename.

Decrypting ASCII-armored messages

To decrypt an ASCII-armored message, enter the following command:

```
ebs --decrypt <ASCII-armored_filename>
```

E-Business Server recognizes that the file is in ASCII-armored format, converts the file back to binary, and creates an output file in normal plaintext form.

When E-Business Server is decrypting the message, it ignores any extraneous text in mail headers that are not enclosed in the ASCII-armored message blocks.

Sending a public key in ASCII-armored format

To send a public key to someone else in ASCII-armored format, add the `--armor` option while extracting the key from your keyring.

```
ebs --key-export <userID> --armor
```

If you forgot to use the `--armor` option when you made a ciphertext file or extracted a key, you can convert the binary file into ASCII-armored format by using the `--armor` option (do not specify encryption). E-Business Server converts the file to a "`.asc`" file.

Sending ASCII text files to different machine environments

E-Business Server encrypts any plaintext file, binary 8-bit data, or ASCII text. The most common use of E-Business Server is for email, which is ASCII text.

ASCII text is represented differently on different machines. For example, on an MSDOS system, all lines of ASCII text are terminated with a carriage return followed by a linefeed. On a UNIX system, all lines end with just a linefeed. On a Macintosh, all lines end with just a carriage return.

Normal unencrypted ASCII text messages are often automatically translated to some common "canonical" form when they are transmitted from one machine to another. Canonical text has a carriage return and a linefeed at the end of each line of text.

Encrypted text cannot be automatically converted by a communication protocol because the plaintext is hidden by encipherment. To remedy this problem, E-Business Server's `--text` option lets you specify that the plaintext be treated as ASCII text and converted to canonical text before encryption. When the message is received, the decrypted plaintext is automatically converted to the appropriate text form for the local environment.

To use this feature, enter the `--text` option when encrypting or signing a message:

```
ebs --encrypt --text <plaintext_filename> --user
<recipient's_userID>
```

E-Business Server includes an environment variable that corresponds to the `--text` option, `TEXTMODE`. If you consistently receive plaintext files rather than binary data, set `TEXTMODE=on` in the E-Business Server configuration file. For more information on setting configuration parameters, see *Using the Configuration File* on page 110.

Wiping your disk

After E-Business Server produces a ciphertext file for you, you can request E-Business Server to automatically overwrite and delete the plaintext file, leaving no trace of plaintext on the disk. Use the `--wipe` option when a plaintext file contains sensitive information; it prevents someone from recovering the file with a disk block scanning utility.

```
ebs --encrypt --wipe <plaintext_filename> --user
<recipient's_userID>
```

For example:

```
ebs --encrypt --wipe confidential.txt mjohnson
```

This instructs E-Business Server to create a ciphertext file `confidential.pgp` and to destroy the plaintext file `confidential.txt`.

Note that this option will not wipe out any fragments of plaintext that your word processor might have created on the disk while you were editing the message before running E-Business Server. Most word processors create backup files, scratch files, or both.

By default, E-Business Server overwrites the file three times. If you want to specify the number of times E-Business Server overwrites the file, and not use the default, add the `--wipe-passes` modifier as shown below.

```
ebs --encrypt --wipe <plaintext_filename> --user
<recipient's_userID> --wipe-passes <number of times>
```

You can also set the `WIPE-PASSES` parameter in the E-Business Server configuration file. For more information, see *WIPE-PASSES* on page 145.

Wiping a sensitive data file

To wipe the contents of a data file without encrypting it, use the `--wipe` option.

By default, E-Business Server overwrites the file three times. If you want to specify the number of times E-Business Server overwrites the file, and not use the default, add the `--wipe-passes` modifier as shown below.

```
ebs --wipe <filename1> <filename2> <filename3>... [--wipe-passes
<number of times>]
```

Wiping your smart card

The smart card must be in your smart card reader.

Before you can wipe your smart card, you must specify the smart card type using the `SMARTCARD-TYPE` parameter in the E-Business Server configuration file or by setting it on the command line using `--smartcard-type`. For more information, see *SMARTCARD-TYPE* on page 137.

If you are using a smart card other than one that we have listed as being supported, then you must set the `SMARTCARD-TYPE` to `other`, as well as specify the path to the DLL to use with it by setting the `SMARTCARD-DLL` parameter. You can also set this on the command line using `--smartcard-dll`. For more information on specifying the DLL, see *SMARTCARD-DLL* on page 137.

To wipe the contents of your smart card, use the `--smartcard` modifier with the `--wipe` option. This deletes all keys and data from your smart card. Optionally, you can specify your smart card PIN on the command line.

```
ebs --wipe --smartcard [--pin <smart card PIN>]
```

For information on how to view the contents of your smart card, see *Viewing your keys* on page 39. For information on the various ways you can supply your PIN to E-Business Server, see *Alternative ways to work with passphrases* on page 93.

Alternative ways to work with passphrases

E-Business Server generally prompts you for your passphrase. If you want to streamline your interaction with E-Business Server, you can use one of the following methods for supplying E-Business Server with your passphrase.

- Specify a file descriptor number
- Set the `PGPPASS` environment variable
- Specify your passphrase on the command line

The recommended method for supplying E-Business Server with your passphrase is by setting the passphrase file descriptor options in the E-Business Server configuration file (or on the command line).

If working on a shared system, you should never store your passphrase with the `PGPPASS` environment variable or enter your passphrase directly on the command line using one of the `--passphrase` options. Your passphrase may be visible to others putting the security of your data at risk.

The `--passphrase` options are safe when using the E-Business Server API.

Specifying a file descriptor number

The most secure method for supplying E-Business Server with your passphrase is by supplying E-Business Server with the file descriptor number to which your passphrase will be passed. You can specify the file descriptor number using the environment variable `PGPPASSFD`, or one of the following configuration file parameters:

- `PASSPHRASE-FD`
- `CONVENTIONAL-PASSPHRASE-FD`
- `PIN-FD`
- `CHALLENGE-FD`

You can use the `PASSPHRASE-FD` option to supply a passphrase, conventional passphrase, or even a smartcard PIN number, but if you ever need to supply more than one type of passphrase on the command line for a single operation, then you must use the appropriate options.

PGPPASSFD

Use the `PGPPASSFD` (passphrase file descriptor) environment variable to supply E-Business Server with the file descriptor to which the passphrase will be passed. This is most useful when writing scripts. This parameter cannot be used if more than one passphrase must be supplied.

```
SET PGPPASSFD=<file_descriptor_number>
```

If this environment variable is set to zero (0), the passphrase is read from standard input (STDIN). E-Business Server uses the first text line from the specified filename as the password.

A `PASSPHRASE-FD` value specified in the configuration file supersedes a value set in the `PGPPASSFD` environment variable. For more information on setting the `PASSPHRASE-FD` value, see *Specifying configuration values* on page 110.

PASSPHRASE-FD

Set the `PASSPHRASE-FD` parameter in the E-Business Server configuration file, or use the `--passphrase-fd` option on the command line to supply E-Business Server with the file descriptor number to which the passphrase will be passed. This is most useful when writing scripts.

For example:

```
ebs --encrypt --passphrase-fd 4 --user joe foo.txt 4<
mypassphrase.txt
```

This instructs the bash shell to get your pgppassphrase for the encryption operation from the file mypassphrase.txt using file-handle number 4 and tells E-Business Server to find it at that location.

CONVENTIONAL-PASSPHRASE-FD

If you need to supply your E-Business Server passphrase, as well as a conventional passphrase, then set the `CONVENTIONAL-PASSPHRASE-FD` parameter in the E-Business Server configuration file, or use the `--conventional-passphrase-fd` option on the command line to supply E-Business Server with the file descriptor number to which the conventional passphrase will be passed. This is most useful when writing scripts.

```
ebs --encrypt --conventional --sign <filename> --passphrase-fd
<file_descriptor_number> --conventional-passphrase-fd
<file_descriptor_number>
```

PIN-FD

Use the `PIN-FD` option to specify a file descriptor for supplying E-Business Server with smartcard PIN number. For example, if you want to sign a file with a key that resides on a smartcard, you would use the following syntax:

```
ebs --sign <filename> --pin-fd <file_descriptor_number>
```

CHALLENGE-FD

Use the `CHALLENGE-FD` option to specify a file descriptor for supplying the challenge passphrase used by Verisign for its certificate revocation process. This can also be set with `--cert-attribute Challenge="..."`. However, specifying the Challenge attribute on the shell command-line could reveal the challenge passphrase to other users on the system. The `--challenge-fd` option provides a more secure method of delivering the passphrase to E-Business Server.

For shell scripts, use the redirection syntax described under `PASSPHRASE-FD`.

Storing your passphrase with PGPPASS

You should not use this feature if working on a shared system. The passphrase may be visible to others.

When E-Business Server needs a passphrase to unlock a secret key, E-Business Server prompts you to enter your passphrase. Use the `PGPPASS` environment variable to store your passphrase. When E-Business Server requires a passphrase, it attempts to use the stored passphrase. If the stored passphrase is incorrect, E-Business Server recovers by prompting you for the correct passphrase.

```
SET PGPPASS= <passphrase>
```

The following is an example of how you might set this variable in the environment.

```
SET PGPPASS="zaphod beeblebrox for president"
```

The above example would eliminate the prompt for the passphrase if the passphrase was "zaphod beeblebrox for president".

This feature is convenient if you regularly receive a large number of incoming messages addressed to your secret key, eliminating the need for you to repeatedly type in your passphrase. The recommended way to use this feature is to enter the command each time you boot your system, and erase it or turn off your computer when you are done.

Passing your passphrase from another application

You should not use this feature if working on a shared system. The passphrase may be visible to others.

E-Business Server includes a command line option, `--passphrase`, that you can use to pass your passphrase into E-Business Server from another application. This option is designed primarily to invoke E-Business Server from inside an email package.

The passphrase in quotes follows the `--passphrase` option on the command line. Use this feature with caution.

For example:

```
ebs --sign <filename> --passphrase <quoted-passphrase>
```

You can also use the `--passphrase` option to supply E-Business Server with a passphrase for conventionally encrypting a file, whether you are performing a regular conventional encryption, creating an SDA, or creating a PGParchive. However, if you need to supply your regular E-Business Server passphrase as well as a conventional passphrase in a single operation, then you must also use the `--conventional-passphrase` option.

For example, to conventionally encrypt and sign a file in a single operation, you would use the following syntax:

```
ebs --encrypt --conventional --sign <filename> --passphrase
<quoted-passphrase> --conventional-passphrase <quoted-passphrase>
```

Therefore, if I wanted to conventionally encrypt a file called *confidential.doc* to the passphrase "In a while crocodile" and digitally sign it with my key, which has a passphrase of "2BeeRnot2be", then I would enter the following:

```
ebs --encrypt --conventional --sign confidential.doc --passphrase
2BeeRnot2be --conventional-passphrase "In a while crocodile"
```

Use the `--pin` option (instead of `--passphrase`) when you need to supply E-Business Server with a PIN number for a smartcard when performing an operation on a smartcard.

For example, you would use the following syntax when wiping your smartcard:

```
ebs --wipe --smartcard --pin <smart card PIN>
```

Working with groups

You may find that you need to perform encryption operations to multiple people at one time. Specifying them individually is inefficient. Instead, you can create distribution lists, or groups, that include everyone to whom you want to encrypt.

For example, if you want to encrypt a file to 10 people at HRdepartment@mcafee.com, you would create a distribution list of that name. You would then add the keys for all 10 members of the HR department mailing list to the group. This enables you to encrypt a file to all 10 people in a single operation.

The `--group` option displays help on all group options.

Creating a group

To create a group, you use the `--group-add` option. This option adds a group definition to the groups file (`pgpgroup.pgr`). You will be asked to supply a description of the group you are creating.

```
ebs --group-add <groupname>
```

The following syntax creates a group with the name "engineers."

```
ebs --group-add engineers
```

Add recipients to a group

You can add users (or groups) to groups using the `--group-add` option.

```
ebs --group-add <groupname> <userID1> <userID2> <userID3>...
```

The following syntax adds the key's for Cal Pettit, Brandon Gillman, and Hal to the `engineers` group.

```
ebs --group-add engineers "Cal Pettit" bgillman Hal
```

Cal Pettit, Brandon Gillman, and Hal are all added to the Engineers group.

Viewing a group

The `--group-list` and `--group-detail` options enable you to obtain more information about your groups.

`ebs --group-list <groupname>` lists the name and description for each group.

`ebs --group-detail <groupname>` lists the name and description for each group as well as the members belonging to each group. (If you do not specify a group, `--group-detail` shows the contents of all of your groups.)

Remove recipients from a group

To remove members or groups from a group, use the `--group-remove` option.

```
ebs --group-remove <groupname> --user <userID>
```

The following syntax removes the key for Cal Pettit from the `engineers` group.

```
ebs --group-remove engineers --user "Cal Pettit"
```

Removing an entire group

To delete an entire group, use the `--group-remove` option.

```
ebs --group-remove <groupname>
```

If the group contains members, E-Business Server prompts for a confirmation that you want to delete the group anyway. Enter `y` to delete the group.

Starting the ebssdkd

This section does not apply to AIX or HPUX.

If you receive a warning message of "Unable to initialize the SDK service, executing in local mode", then you may need to manually re-start the `ebssdkd`.

To start the E-Business Server SDK service on a Solaris or Linux system:

1 Login to your system as root.

2 Run the system script by entering the following at any command prompt:

```
/etc/init.d/ebssdkd start
```

The SDK service starts.

To start the E-Business Server SDK service on a Windows NT or Windows 2000 system:

1 Double-click the **Services** icon from the Control Panel.

On Windows 2000, the Services icon is located in the Administrative Tools folder from the Control Panel.

2 Select the `ebssdkService`, and click `Start`.

The SDK service starts.

Keeping your keyring files open with EBScache

EBScache allows you to load a specified keyring and keep it open for as long as it is running. Previously, E-Business Server would close the keyring and exit the program when the operation was complete. With PGPcache, the keyring files stay open until you kill the process.

```
EBScache [--version|--help] [--filelist <filename>]
[<public_keyring> <secret_keyring>...]
```

Use the `--filelist` option to specify a text file that contains a list of keyring filenames you want PGPcache to load. Each filename must be on a separate line and the files must exist in pairs—a public keyring and a secret keyring.

The following is an example of three pairs of keyring files:

```
#Lines beginning with a '#' are considered comments.

pubring1.pkr
  secring1.skr

/home/joe/.pgp/public.pkr
  /home/joe/.pgp/secret.skr

.pgp/foo.bin
  .secrets/bar.bin
```

If filenames are supplied—in conjunction with the `--filelist` option or not—then they are also opened as keyrings.

If no parameters are supplied and a file called '`ebscache.lst`' exists in the same directory as the executable, then processing continues as if `--filelist ./ebscache.lst` was entered from the directory that contains the executable.

You can get more information about the version of PGPcache you are using by including the `--version` or `--help` options.

To display the version number of PGPcache you are using, enter the following:

```
ebscache --version
```

To display command usage information for PGPcache, enter the following:

```
ebscache --help
```

Under Unix, the process detaches from the terminal once it has successfully started. The process should be run at a privilege level sufficient to open all the keyrings read-only. Under UNIX, the process can also be sent a SIGHUP signal using the kill command to tell it to re-process its arguments and re-load the keyrings.

10 Using the E-Business Server API

To make it simpler to call E-Business Server commands from other languages and to improve performance when commands are called repeatedly, a simple *C* language API for entering E-Business Server commands is included in this version of the software. The central function in the API imitates the command line, allowing E-Business Server commands normally issued from the command line to be called through the API. Four supporting functions for freeing allocated memory, translating returned error codes, and initializing and cleaning up the API are also provided.

The E-Business Server API is supported on Windows and UNIX platforms.

Library and header file organization

The following header files are included in the E-Business Server API:

- `PGPeBiz.h`. Contains function prototypes for the E-Business Server API.
- `PGPBase.h`. Contains the basic data types used by the E-Business Server (and EBSsdk) APIs.
- `PGPPFLConfig.h`. Platform-dependent data type configuration file. Versions for each supported platform are shipped with the E-Business Server API. The file appropriate for the user's platform will be installed.

EBSsdk ships as shared libraries on all supported platforms. On all UNIX platforms except HP-UX, the E-Business Server API and EBSsdk shared libraries are named:

- `libEBSsdk.so`
- `libEBSsdkNetwork.so`
- `libEBSEngine.so`

The HP-UX versions of these files have ".sl" filename extensions. The Win32 version uses the `EBSEngine.dll` dynamic link library

The Windows version of the API includes an import library, `EBSEngine.lib`, which must be linked with your program to properly resolve functions exported from the EBSsdk DLLs.

E-Business Server API functions

The E-Business Server API provides these functions:

- `PGPeBizInit()`
- `PGPeBizCleanUp()`
- `PGPeBiz()`
- `PGPeBizFree()`
- `PGPeBizErrorString()`

In order to set up the proper environment for E-Business Server API calls, the `PGPeBizInit()` function must first be called before calling any other E-Business Server API functions. This creates a context of type `PGPeBizContextRef` that is used in subsequent E-Business Server API calls in the current process, including API calls from threads spawned from the current process.

A corresponding call to `PGPeBizCleanUp()` must be made at some point to release the context. You can call `PGPeBizInit()` more than once in a process, provided you call PGPeBizCleanUp() to release the current context before you create a new one.

The syntax and parameters for each of the E-Business Server API functions are described below.

PGPeBizInit

Creates a context for use in subsequent E-Business Server API calls. Only one such context should exist at a time in a process. The context is typically created near the beginning of an application in its main thread before spawning any other threads.

Syntax

```
PGPInt32 PGPeBizInit (

PGPeBizContextRef *context );
```

Parameters

`context`	API context parameter.

Notes

A call to `PGPeBizCleanUp()` must be made to free the context created by this function when a process finishes making E-Business Server API calls or before creating another `PGPeBizContextRef` context in the same process.

PGPeBizCleanUp

Releases a context created by `PGPeBizInit()`. Call this function when a process is finished making E-Business Server API calls or before creating a new `PGPeBizContextRef` context in the same process.

Syntax

```
PGPInt32 PGPeBizCleanUp (

PGPeBizContextRef context);
```

Parameters

`context`	The E-Business Server context to release.

Notes

After calling this function, you must create a new context with `PGPeBizInit()` before making further E-Business Server API calls, as the context released by this function will no longer be usable.

PGPeBiz

Specifies an E-Business Server command. Output buffers for stdout and stderr may be optionally specified.

Syntax

```
PGPInt32 PGPeBiz(

PGPeBizContextRef context,

const char *args,

const PGPByte *stdIn,

PGPUInt32 stdInLen,

PGPByte **stdOut,

PGPUInt32 *stdOutLen,

PGPByte **stdErr,

PGPUInt32 *stdErrLen);
```

Parameters

`context`	The context value from `PGPeBizInit()`.
`args`	A null-terminated string containing the command line arguments for the operation you want to perform. The string contains the arguments exactly as they would be entered on the command line, except for shell extensions, which should not be included since they will not be processed. For example, piping or redirecting input or output should not be included.
`stdIn`	The data for the E-Business Server API to use as stdin. This input parameter is optional. Since the data might be binary, the value is not null-terminated. If NULL (0) is specified, then no stdin data will be used.
`stdInLen`	Input parameter that specifies the number of bytes passed in via the `stdIn` parameter.
`stdOut`	A buffer, internally allocated by the program, that receives the data normally written to stdout (displayed on the screen). This output parameter is optional. If NULL is passed in, then the stdout data is not returned. If NULL is not passed in, then this buffer must be freed by calling `PGPeBizFree()`.
`stdOutLen`	Output parameter that specifies the length of data, in bytes, placed in the `stdOut` buffer.
`stdErr`	A buffer, allocated internally by the program, that receives the data normally written to stderr (displayed on the screen). This output parameter is optional. If NULL is passed in, then the stderr data is not returned. If NULL is not passed in, then this buffer must be freed by calling `PGPeBizFree()`.
`stdErrLen`	Output parameter that specifies the length of data, in bytes, placed in the `stdErr` buffer.

Notes

The return code for this API is the exit code that is normally set by the E-Business Server executable. Additionally, a few API-specific return codes may be returned that indicate input parameter errors. The output buffers may be useful in diagnosing these errors.

A return code of zero indicates success. A non-zero return code indicates an error of some kind. Note that even if the return code is non-zero, the `stdOut` and `stdErr` parameters may contain data. The contents of these buffers will usually assist in diagnosing the problem.

Unlike the command-line interface, the API does not provide a way to prompt the user for more data. If E-Business Server does not have all the information it needs to complete the call (that is, if it gets to a point where the executable would prompt for user input), then the call will fail and return a useful error code.

A possible use for the `stdIn` parameter would be to pass encrypted data to an API encrypt command without first storing it to disk.

PGPeBizFree

The output buffers used in the `PGPeBiz()` function are allocated internally by the API. `PGPeBizFree()` is provided to free these buffers. Buffers allocated by `PGPeBiz()` must be freed upon completion of the call before reusing the pointers for which memory was allocated.

Syntax

```
PGPInt32 PGPeBizFree(

PGPeBizContextRef context,

void *buffer );
```

Parameters

`context`	The context value from `PGPeBizInit()`.
`buffer`	The pointer to free. `buffer` must be a pointer for which memory was previously allocated by a call to `PGPeBiz()`.

Notes

A return value of zero indicates that the call was successful. A non-zero return code indicates an error occurred while processing the call.

PGPeBizErrorString

Converts an E-Business Server API return code into a text string that describes the error.

Syntax

```
char *PGPeBizErrorString(
PGPeBizContextRef context,
PGPError returnCode,
char *errorString,
size_t maxLen);
```

Parameters

`context`	The context value from `PGPeBizInit()`.
`returnCode`	The return code to convert to a descriptive string.
`errorString`	A buffer to hold the descriptive string.
`maxLen`	The maximum length of `*errorString`, plus a terminating NULL character.

Notes

The function returns a pointer to `errorString`. The caller must allocate and free the `errorString` buffer. If the error string is larger than the specified buffer, then the string will be truncated and null-terminated.

Programming with the E-Business Server API

The E-Business Server API is supported on Win32 and several UNIX platforms. The following sections provide information for using the API on these platforms.

Programming on Win32

The E-Business Server API functions are exported from the main E-Business Server DLL, `EBSEngine.dll`. This file is installed in the top-level E-Business Server installation directory. The other EBSsdk DLLs are installed to `\%SYSTEMROOT%\system32`.

The three API header files—`PGPBase.h`, `PGPeBiz.h` and `PGPPFLConfig.h`—are installed in a `.\include` directory under the E-Business Server installation directory. A `.\lib` directory containing the import library file `EBSEngine.lib` is installed at the same level as the `.\include` directory. Win32 applications using the API must include `PGPeBiz.h` and link to `EBSEngine.lib`.

The only header file you need to include in your source is `PGPeBiz.h`, which in turn loads the other E-Business Server header files. Ensure that the location of the supplied header files is included in your compiler's include path.

Programming on UNIX

On UNIX platforms, the three API header files are installed in the directory `/usr/local/ebs/include`. This directory needs to be included in the header search path. The only header file you need to include in your source is `PGPeBiz.h`, which in turn loads the other E-Business Server header files.

The shared object libraries are installed in the `/usr/local/ebs/lib` directory. Sample Linux `gcc` command lines with appropriate switches and paths are shown below:

Compilation example:

```
gcc -I/usr/local/ebs/include -c userprog.c -o userprog.o
```

Linking example:

```
gcc userprog.o -o userprog -L/usr/local/ebs/lib \ -lEBSEngine
-lEBSsdk -lEBSsdkNetwork -lm
```

Setting the run-time library path

The way that the run-time library path is set varies between the various UNIX platforms. The only platform on which no manual configuration need be performed is Linux. The following sections discuss the steps required to ensure your programs can find the EBSsdk shared libraries at run time.

You may need to consult your compiler and linker user manuals for more information on using shared libraries and the run-time linker.

Linux

The way Linux locates run-time libraries differs from other versions of UNIX. For Linux, the E-Business Server installation script adds the `/usr/local/ebs/lib` path to `/etc/ld.so.conf` and runs `ldconfig -v` to update `/etc/ld.so.cache`. This means that no special steps are required to ensure that applications can find the EBSsdk libraries, so long as the libraries, or links to the libraries, exist in their default installation path.

Solaris

For Solaris, you can set your `RPATH` properly in your executable. For example, you might specify a "`-R/usr/local/ebs/lib`" option to the Solaris Linker. Alternatively, you can add `/usr/local/ebs/lib` to the LD_LIBRARY_PATH environment variable or put links to the EBSsdk libraries in the default run-time library directory, `/usr/lib`.

AIX and HP-UX

By default, the AIX and HP-UX linkers will imbed any `-L` paths into the executable, and the run-time linker will use these paths to locate required shared libraries.

For example, the following AIX `gcc` command line specifies the default installation directory of the EBSsdk libraries:

```
gcc -wl,-brtl userprog.o -o userprog \
-L/usr/local/ebs/lib -lEBSEngine -lEBSsdk \
-lEBSsdkNetwork  -lm
```

This command will cause `-L/usr/local/ebs/lib` to be imbedded in the resulting executable. This requires that the libraries, or links to the libraries, exist in their default installation directory on any machine where the executable will be run.

11 Using the Configuration File

Learning about the configuration file

E-Business Server stores a number of user-defined parameters in the configuration file `pgp.cfg`. A configuration file enables you to define flags and parameters for E-Business Server, eliminating the need to define these parameters on the command line. (For more information on `pgp.cfg` and its location, see *Setting the location of E-Business Server files* on page 17.)

Use these configuration parameters to perform the following tasks as well as many others:

- Control where E-Business Server stores its temporary files.
- Adjust E-Business Server's level of skepticism when it evaluates a key's validity based on the number of the key's certifying signatures.
- Set the location and name of your keyrings.

Specifying configuration values

Configuration parameters may be assigned integer values, character string values, or on/off values; the type of values depends on the type of parameter. E-Business Server includes a sample configuration file for your review.

The following rules apply to the configuration file:

- E-Business Server ignores blank lines.
- E-Business Server also ignores characters that follow the comment character, #.
- Keywords are not case-sensitive.

The following is a short sample fragment of a typical configuration file, where the file's owner used comments in conjunction with the actual settings:

```
# TMP is the directory for E-Business Server scratch files.

TMP = "e:\"     # Can be overridden by environment variable TMP.

Armor = on      # Use --armor flag for ASCII armor when applicable.

# CERT-DEPTH sets how many levels deep you can nest trusted introducers.

cert-depth = 3
```

E-Business Server uses default values for the configuration parameters under the following conditions:

- When configuration parameters are not defined.
- If the configuration file does not exist.
- If E-Business Server cannot find the configuration file.

Setting configuration parameters from the command line

Typically, you set configuration parameters in the E-Business Server configuration file, but you can also set configuration parameters directly from the E-Business Server command line.

Unless you are working in legacy mode, you can set options on the command line by using the following syntax:

```
ebs --<option> <value>
```

For example, if the `ARMOR` parameter is set to `on` in the E-Business Server configuration file, you can override this setting by using the `--armor` option on the command line:

```
ebs --encrypt --armor off message.txt --user smith
```

If you are working in legacy mode, then you must precede the parameter setting with a plus (+) character. For example, if the `ENCRYPT-TO-SELF` parameter is turned off in the configuration file, but you want to use it in a single legacy operation, then enter the following on the command line:

```
ebs -e +ENCRYPT-TO-SELF=on message.txt smith
```

For the location of the `pgp.cfg` file, please refer to *Setting the location of E-Business Server files* on page 17.

The remainder of this chapter summarizes E-Business Server's configuration parameters in alphabetical order.

Configuration parameters

ADD-ALL

Specifies that `--keyserver-fetch` will always add all matching keys found on the keyserver to the local keyring. If not specified, then each matching key is displayed in turn, and E-Business Server prompts for confirmation that you want to import each key.

Syntax

```
ADD-ALL = off
```

ADK-KEY

Specifies an Additional Decryption Key (ADK) for messages encrypted and keys generated.

Encrypt to an Additional Decryption Key (ADK). When this parameter is used, all generated keys have an ADK equal to the value of ADK-KEY. Additionally, everything E-Business Server encrypts to a public key is also encrypted to the ADK key identified by this parameter. Note the difference between *incoming* ADKs and *outgoing* ADKs as described in *Managing Keys* on page 38.

If you choose to use two different keys for the incoming and outgoing ADKs, you can set ADK-KEY to specify the outgoing ADK, then use --adk-key <keyID> on the command line to override it during key generation to specify the incoming ADK attached to such keys.

Syntax

```
ADK-KEY = <keyID>
```

For example:

```
ADK-KEY = 0xAB12C34D
```

Default Value

```
ADK-KEY = ""
```

Notes

- You use ADK-KEY in conjunction with the parameter ENFORCE-ADK to determine whether E-Business Server enforces the use of ADKs. If ENFORCE-ADK is not set, then users can subvert use of the ADK.
- If ENFORCE-ADK is on and the encryption key was generated with ENFORCE-ADK set to on, data is always encrypted to the ADK if the ADK key is available. If the ADK key is not available, an error message appears and the encryption operation fails.
- If ENFORCE-ADK is set to off and the ADK key is not present on the user's keyring, E-Business Server displays a warning message and does not encrypt to the ADK key.
- You can also set this parameter on the command line with the -- prefix; for example, --ADK-KEY 0xAB12C34D.

We recommend you always specify the ADK-KEY in your configuration file using the key ID to prevent any potential security holes—if you were to specify the ADK-KEY using the key's user ID, an interloper might create another key with the same user ID and introduce a means for decrypting secret data.

ALIAS

Aliases are not supported in Legacy mode.

The ALIAS parameter allows you to create aliases (shortcuts) for command line options and their arguments. Aliases can only be defined in the configuration file and must start with at least one dash. If an alias contains spaces, then you must enclose it in quotes.

Syntax

```
ALIAS -<alias> --<long-option>

ALIAS --<alias> --<long-option>

ALIAS -<alias> "--<long-option> <value>"

ALIAS -<alias> "--<long-option> --<argument> %1 --<argument> %2
  --<argument> %3..."
```

Notes

Creating aliases for single command line options

To create an alias for a single E-Business Server command line option, use the following syntax:

```
ALIAS -<alias> --<long-option>
```

For example:

```
ALIAS -e --encrypt
```

Once this is set, you can enter `-e` on the command line instead of entering `--encrypt` when you want to encrypt data.

If you want to give the alias a long name, then you can put two dashes in front of it. For example, if you wanted to set `--search` as a shortcut for `--keyserver-search`, then you would enter the following in the configuration file:

```
ALIAS --search --keyserver-search
```

Creating aliases for command line options with values

To create an alias for a E-Business Server command line option and its value, you must enclose the option and value in quotes because there is a space between them:

```
ALIAS --<alias> "--<long-option> <value>"
```

For example, if you wanted to set `--bigkey` as a shortcut for `--key-size 4096`, then you would enter the following in the configuration file:

```
ALIAS --bigkey "--key-size 4096"
```

Once this is set, you can enter the following on the command line to create a new key with a key size of 4096 bits:

```
ebs --key-gen --bigkey
```

Creating aliases for command line options with multiple arguments

To create an alias for a E-Business Server option and multiple arguments, use the following syntax:

```
ALIAS --<alias> "--<long-option> <argument> %1 --<argument> %2
  --<argument> %3..."
```

You can list up to 9 arguments. For example, you could create an alias to represent creating a new key with a specific key size, key type, and user ID. To do so, you would enter the following in the configuration file:

```
ALIAS --newkey "--key-gen --key-size %1 --key-type %2 --userid %3"
```

When you want to create a new key, then you would enter the alias followed by the values for arguments 1, 2 and 3 on the command line. For example:

```
ebs --newkey 1024 RSA "jackson jones"
```

E-Business Server recognizes that 1024 corresponds to the key size, RSA corresponds to the key type, and "jackson jones" corresponds to the user ID because of the order in which they are listed.

To assist you in diagnosing problems using aliases, set `INFO` to `Verbose`. When set, E-Business Server displays the expanded form of all commands. For details, see *INFO* on page 127.

ALLOW-PASSPHRASE-RETRY

Tells E-Business Server to abort an encryption operation if the user does not specify a passphrase in the original encryption command, or if the supplied passphrase is not correct.

Default Value

```
ALLOW-PASSPHRASE-RETRY =
```

ARMOR

If enabled, this parameter causes E-Business Server to emit ciphertext or keys in ASCII-armored format suitable for transport through email channels.

Default Value

```
ARMOR = off
```

Notes

- Output files are named with the `.asc` extension.
- The configuration parameter `ARMOR` is equivalent to the `--armor` command line option.
- If you intend to use E-Business Server primarily for email purposes, you may wish to turn this parameter on (`ARMOR=on`). This can be overridden on the command-line by entering the following "`--armor off`".

AUTHENTICATE

Only decrypt a file if it has been signed.

Default Value

```
Off
```

Details

You can also set this on the command line by entering `--authenticate`.

AUTH-PASSPHRASE

Although not recommended, you can set the AUTH-PASSPHRASE configuration parameter equal to an E-Business Server passphrase (if AUTH-USER specifies a key ID) or password (if AUTH-USER specifies a user ID).

Putting a key passphrase in the configuration file effectively nullifies the protection the passphrase is designed to offer.

Default Value

```
AUTH-PASSPRASE = ""
```

AUTH-USER

Along with AUTH-PASSPHRASE, the AUTH-USER parameter specifies a user ID or a key ID to use for authenticating with a remote user, such as for reconstituting split keys (in this case, a key ID must be specified) or for key reconstruction on a generic LDAP server (in this case, a user ID must be specified).

Default Value

```
AUTH-USER = ""
```

BATCHMODE

The BATCHMODE parameter specifies that default answers are accepted for all prompts, instead of waiting for user interaction. This option is deprecated and will be removed from future versions of the product. It is supported only in legacy mode operation.

CA-REVOCATION-URL

Specifies the URL used to fetch the Certificate Revocation List (CRL) from the CA. The URL must be fully qualified. For example, you might enter something like https://myca.mcafee.com:444

Default Value

```
CA-REVOCATION-URL = ""
```

CA-ROOT-CERT

Specifies the key ID belonging to the root CA's X.509 certificate. It goes along with the other CA-* parameters.

Syntax

```
CA-ROOT-CERT = <keyID>
```

For example:

```
CA-ROOT-CERT = 0xAB12C34D
```

Default Value

```
CA-ROOT-CERT = ""
```

CA-TYPE

Specifies the type of CA server described by the other `CA-*` parameters.

Default Value

```
CA-TYPE = ""
```

Your options are:

- Entrust
- iPlanet
- NetTools
- Verisign
- Win2K

CA-URL

Specifies the default URL used to connect to the Certificate Authority (CA). The URL must be fully qualified. For example, you might enter something like https://myca.mcafee.com:444.

Default Value

```
CA-URL = ""
```

CERT-ATTRIBUTE

Use to specify a certificate attribute always bound to certificate requests and X.509 signatures. All `ebs --cert-request` or `ebs --sign --x509` operations will include these attributes as part of their operation.

Syntax

```
CERT-ATTRIBUTE "name=value"
```

For example:

```
CERT-ATTRIBUTE "O=McAfee"
```

Notes

- No certificate attributes are set by default.
- Multiple certificate attributes may be specified in the configuration file.
- For a complete list of the certificate attributes that E-Business Server supports, including a list of Verisign-specific attributes, see *Supported Certificate Attributes*

CERT-DEPTH

The configuration parameter `CERT-DEPTH` identifies how many levels deep you can nest trusted introducers. (Trusted introducers are those people who you trust to certify—or validate—others' keys. If a trusted introducer certifies a key, it will appear valid on your public keyring.)

Default Value

```
CERT-DEPTH = 4
```

Notes

For example, if `CERT-DEPTH` is set to `1`, there can only be one layer of introducers below your own ultimately-trusted key. If that is the case, you are required to directly certify the public keys of all trusted introducers on your keyring. If you set `CERT-DEPTH` to zero, you could have no introducers at all, and you would have to directly certify each and every key on your public keyring to use it.

The minimum `CERT-DEPTH` is `0`; the maximum is `8`.

CHALLENGE-FD

Use the `CHALLENGE-FD` option to specify a file descriptor for supplying the challenge passphrase used by Verisign for its certificate revocation process. This can also be set with `--cert-attribute Challenge="..."` However, specifying the Challenge attribute on the shell command-line could reveal the challenge passphrase to other users on the system. The `--challenge-fd` option provides a more secure method of delivering the passphrase to E-Business Server. Set the `CHALLENGE-FD` parameter equal to a file descriptor number.

For shell scripts, use the redirection syntax described under `PASSPHRASE-FD`.

Note that API programs can send passphrases via its "command line" argument with the same safety as storing the passphrase in the program's memory space.

CIPHER

Specifies which symmetric cipher E-Business Server should use to encrypt the session key—IDEA, Triple-DES, CAST, AES, or Twofish.

This parameter specifies the cipher preference when generating a new key pair, when changing the self-signature or passphrase on your private key, and when performing a conventional encryption operation, except when you are creating an SDA or PGParchive, which always use CAST5.

This setting is ignored when generating RSA Legacy keys. RSA Legacy keys always use the IDEA cipher.

Default Value

```
CIPHER = IDEA
```

Your options are:

- IDEA
- 3DES
- CAST5
- AES128
- AES192
- AES256
- Twofish

CIPHERNUM

The `CIPHERNUM` parameter is only supported for compatibility purposes. Unless you are running in legacy mode, a warning appears if your configuration file contains this setting. Use the `CIPHER` parameter instead. For more information, see *CIPHER* on page 117.

CLEARSIG

Use the `CLEARSIG` parameter to generate a signed message that can be read with human eyes, without the aid of E-Business Server. The recipient must still use E-Business Server to verify the signature. For an example of a clear-signed message, see *Producing a clear-signed message* on page 58.

E-Business Server messages that are signed and not encrypted include a signature certificate as well as the compressed message. For email, this message is then ASCII-armored, rendering the message unreadable to human eyes. E-Business Server would need to decode the message in order to make it readable.

If the original plaintext message is in text, not binary form, `CLEARSIG` can make the signed message readable by skipping the message compression and armoring just the signature portion of the message. Thus, the message can be read with human eyes, without the aid of E-Business Server (again, the recipient still needs E-Business Server to be able to verify the signature).

Default Value

```
CLEARSIG = on
```

Notes

- `CLEARSIG` is enabled by default, and applies to `--sign` operations only when `--armor` and `--text` are also enabled. Set `ARMOR=ON` (or use the `--armor` option), and set `TEXT=ON` (or use the `--text` option).
- `CLEARSIG` can be disabled on the command line with `--clearsig off`.

- Note that since this method only applies ASCII armor to the binary signature certificate, and not to the message text itself, there is some risk that the unarmored message may suffer some accidental corruption while en route. This can happen if it passes through an email gateway that performs character set conversions, or in some cases extra spaces may be added to or stripped from the ends of lines. If this occurs, the signature will fail to verify, which may give a false indication of possible tampering.
- When E-Business Server calculates the signature for text in `CLEARSIG` mode, trailing blanks are ignored on each line.

CMDLINE-FORMAT

Sets the compatibility mode for legacy or long options when entering command-line options. You can also set this option on the command line with the `--cmdline-format` option.

E-Business Server is mostly compatible with the legacy options and configuration values used in previous versions. However, the on-screen messages, error strings, and prompts have changed. By setting the configuration value `CMDLINE-FORMAT` (or the environment variable `PGP_CMDLINE_FORMAT`) to `legacy`, the product translates legacy options into equivalent new options, and recognizes most previous configuration file options. E-Business Server displays a warning for any unsupported options.

Syntax

```
CMDLINE-FORMAT = <legacy|long>
```

Default Value

```
CMDLINE-FORMAT = long
```

Notes

- If `INFO` is set to `Verbose` in the E-Business Server configuration file (same as `VERBOSE=2` in legacy mode), then E-Business Server displays the long-option equivalent for all legacy commands. This may assist you in porting legacy command line options to the new long-options. For more information on the `INFO` parameter, see *INFO* on page 127.
- When using legacy mode, long-options and aliases are not supported.

COMMENT

Displays a comment header in all armored output just beneath the E-Business Server Version header.

Default Value

```
COMMENT = ""
```

COMPATIBLE

The COMPATIBLE parameter, used for specifying E-Business Server 2.6.2 interface compatibility, is not supported in this version. It is allowed for compatibility purposes, but is ignored. In this release, use the EXPORT-FORMAT parameter to set compatibility with E-Business Server 2.6.2. For more information, see *EXPORT-FORMAT* on page 123.

COMPLETES-NEEDED

The configuration parameter COMPLETES-NEEDED identifies the minimum number of completely trusted introducers required to fully certify a public key on your public keyring. For more information on trusted introducers, see *An Introduction to Cryptography*.

Default Value

```
COMPLETES-NEEDED = 1
```

COMPRESS

The configuration parameter COMPRESS enables or disables data compression before encryption. It is used mainly to debug E-Business Server. Under normal circumstances, E-Business Server attempts to compress the plaintext before it encrypts it. Compression strengthens security. Therefore, turning COMPRESS off weakens your security. We recommend you do not change this setting.

Default Value

```
COMPRESS = on
```

CONVENTIONAL-PASSPHRASE-FD

If CONVENTIONAL-PASSPHRASE-FD is specified, E-Business Server reads the passphrase from the specified file descriptor. Use this parameter to transmit a conventional passphrase from one program to another in order to conventionally encrypt a file. Set the CONVENTIONAL-PASSPHRASE-FD parameter equal to a file descriptor number.

This option is only necessary if you must supply both your regular E-Business Server passphrase and a conventional passphrase in a single operation. Otherwise, you can use the PASSPHRASE-FD option to supply your passphrase, whether it's your key passphrase, a conventional passphrase, or even a smart card PIN number. For more information on this option, see *PASSPHRASE-FD* on page 131.

For information on the various ways you can supply your passphrase to E-Business Server, see *Alternative ways to work with passphrases* on page 93.

DEFAULT-KEY

The configuration parameter DEFAULT-KEY specifies the default key ID to use when selecting a private key for making signatures. If DEFAULT-KEY is not defined, E-Business Server uses the most recently generated private key found on your secret keyring (`secring.skr`). You can override this setting by using the `--default-key` option on the E-Business Server command line.

Default Value

```
DEFAULT-KEY = ""
```

Notes

- ENCRYPT-TO-SELF refers to DEFAULT-KEY.
- You must always specify DEFAULT-KEY using the key's key ID, not user ID, to prevent a potential security risk.

DEPTH

The configuration parameter DEPTH specifies how many levels deep you can set trust for a meta or trusted introducer signature. (Trusted introducers are those people who you trust to certify—or validate—others' keys. If a trusted introducer certifies a key, it will appear valid on your public keyring.)

Default Value

Meta Introducer signature:

```
DEPTH = 2
```

Trusted Introducer signature:

```
DEPTH = 1
```

Notes

- This setting is ignored if SIG-TYPE is set to local or exportable.
- The DEPTH setting applies to an introducer signature (meta or trusted), whereas the CERT-DEPTH configuration option limits the depth of the trust chain. E-Business Server will not validate any keys deeper in the chain of trust than the level specified by CERT-DEPTH.

DISCARD-PATHS

Instructs E-Business Server to strip any relative path information from the list of files you want to include in a Self-Decrypting Archive (SDA) or PGParchive. During decryption of the archive, the files are placed in the current directory instead of in subdirectories of the current directory.

Default Value

```
DISCARD-PATHS=
```

Notes

You can also add the --discard-paths option to the command line when creating an SDA or PGParchive.

For example:

```
ebs --encrypt --sda --discard-paths foo/bar.txt abc/xyz.txt
```

In this example, E-Business Server includes the files `bar.txt` and `xyz.txt` in the archive, but the file's relative paths are not included. When the archive is decrypted, both files are placed in the current directory and not in `foo` and `abc` subdirectories.

ENCRYPT-TO-SELF

Instructs E-Business Server to always add the recipient specified in the configuration parameter `DEFAULT-KEY` to its list of recipients and thus always encrypt to the predefined key as well as to any specified recipients.

Just because you originated the encryption does not mean you can decrypt the information. If you want to have access later to messages you encrypt to another person, you must enable `ENCRYPT-TO-SELF`.

Default Value

```
ENCRYPT-TO-SELF = off
```

ENFORCE-ADK

Forces encryption to any ADKs associated with a recipient's key and to the `ADK-KEY` configuration setting.

Default Value

```
ENFORCE-ADK = off
```

Notes

- With this setting enabled, if a user tries to encrypt to a key that is associated with an ADK (or to any key, with `ADK-KEY` enabled), E-Business Server attempts to encrypt to the ADK as well. If the ADK is not present on the keyring, E-Business Server generates an error message.
- If `ENFORCE-ADK` is set to `off` and the ADK is not present on the user's keyring, E-Business Server displays a warning. It then encrypts the message, but does not encrypt to the ADK key.
- You can override this parameter on the command line by specifying `--enforce-adk off`.
- On Unix platforms, you can prevent individual users from overriding this parameter by specifying `ENFORCE-ADK=on` in the system policy configuration file.
- For more information on using ADKs, see *Implementing your Additional Decryption Keys* on page 55.

EXPIRES-AFTER

Specifies the default validity period for a key signature or a newly-generated key.

In the configuration file, the `EXPIRES-AFTER` parameter should be set to the total number of days you want the key or signature to remain valid. If you never want it to expire, then set the expiration date to zero (0).

Default Value

```
EXPIRES-AFTER = 0
```

EXPORTABLE

This option has been deprecated and is only supported in Legacy mode. Use `SIG-TYPE=export` instead. For details, see *SIG-TYPE* on page 136.

EXPORT-FORMAT

Specifies whether you want E-Business Server to strip newer key features, such as photo IDs, from the key during a `--key-export` operation.

Normally when keys are exported (copied), the complete key is included. If this key is to be used by versions of E-Business Server prior to 6.0, or by other software that does not recognize the newer attributes (such as photo IDs), you may wish to set `EXPORT-FORMAT` to `COMPATIBLE`. If you do not want E-Business Server to strip key features during a `--key-export` operation, then keep the default setting, `COMPLETE`.

Default Value

```
EXPORT-FORMAT = COMPLETE
```

FASTKEYGEN

Use to specify fast key generation. With this setting enabled, DH/DSS keys are generated using "canned primes" for common key sizes (currently 1536, 2048, 3072, 4096) to speed key generation. Other key sizes have no canned primes, so this option is ignored in those cases.

Default Value

```
FASTKEYGEN = on
```

FINGERPRINT-VIEW

Specifies the format for displaying your fingerprint information in the `--key-detail` view, or when `--key-detail` is specified for keyserver searches.

Syntax

```
FINGERPRINT-VIEW = HEX|WORDS
```

Default Value

```
FINGERPRINT-VIEW = HEX
```

Your hexadecimal fingerprint is made up of a 40 character digest of the public key components (RSA Legacy keys have 32 character fingerprints).

The word list is made up of special authentication words that E-Business Server uses and are carefully selected to be phonetically distinct and easy to understand without phonetic ambiguity. If you'd like to know more about the word hash technique and view the word list, see *Biometric Word Lists*

FORCE

To run E-Business Server non-interactively from a UNIX shell script or MSDOS batch file, you can use the FORCE option to eliminate interaction with E-Business Server in the following situations. All warnings and confirmations are suppressed.

- When you delete, disable, or revoke a key on your keyring (either public or private).
- When you disable or delete a key from a key server.
- When encrypting to, or signing, an untrusted key.
- When removing a photo ID from your key.
- (Legacy only) When a filename to be output already exist, such as for decrypting or key-exporting. For non-legacy operation, use –overwrite for this purpose.

Default Value

```
FORCE = off
```

FTP-PASSWORD

Specifies the password that you want E-Business Server to use when connecting to an FTP server during encryption or decryption operations.

This parameter works with the FTP-USERNAME parameter. If you do not specify a value for FTP-USERNAME, E-Business Server assumes that you want to connect as an anonymous user, and you do not need to specify a value for FTP-PASSWORD. In this case, E-Business Server will automatically use anonymous@ as the password.

Syntax

```
FTP-PASSWORD = <password>
```

Default Value

```
FTP-PASSWORD=
```

Notes

You can also add the --ftp-password option to the command line when encrypting or decrypting files.

For example:

```
ebs --encrypt --ftp --ftp-password <password>
```

FTP-PATHNAME

Specifies which directory to which E-Business Server transfers encrypted data during encryption or decryption operations.

Syntax

```
FTP-PATHNAME = <directory path>
```

Default Value

```
FTP-PATHNAME =
```

FTP-PORT

Specifies the port that you want E-Business Server to use when connecting to an FTP server during encryption or decryption operations.

Syntax

```
FTP-PORT = <port number>
```

Default Value

```
FTP-PORT=
```

Notes

You can also add the `--ftp-port` option to the command line when encrypting or decrypting files.

For example:

```
ebs --encrypt --ftp --ftp-port <port number>
```

FTP-SECURE

Tells E-Business Server to do an FTP transfer securely, using the FTPS protocol. This protocol uses Transport Layer Security (TLS) to protect the user name, password, file name and data.

The remote FTP server's X.509 certificate must be signed and on your keyring for `--ftp-secure` to work.

E-Business Server only supports servers that use the FTPS protocol and are capable of exporting their X.509 certificates.

Syntax

```
FTP-SECURE = <Yes|No>
```

Default Value

```
FTP-SECURE =
```

FTP-SERVER

Specifies the FTP server that you want E-Business Server to connect to during encryption or decryption operations.

Syntax

```
FTP-SERVER = <host name|IP address>
```

Default Value

```
FTP-SERVER=
```

Notes

You can also add the `--ftp-server` option to the command line when encrypting or decrypting files.

For example:

```
ebs --encrypt --ftp --ftp-server <server information>
```

FTP-USERNAME

Specifies the user name that you want E-Business Server to use when connecting to an FTP server during encryption or decryption operations. If you do not specify a user name, E-Business Server connects as an anonymous user.

This parameter works with the `FTP-PASSWORD` parameter. If you do not specify a value for `FTP-USERNAME`, E-Business Server assumes that you want to connect as an anonymous user and you do not need to specify a value for `FTP-PASSWORD`. (E-Business Server will automatically use `anonymous@` as the password.) If you do enter a value for `FTP-USERNAME`, and do not specify a value for `FTP-PASSWORD`, then E-Business Server will automatically prompt you for a password when you perform encryption or decryption operations using FTP.

Syntax

```
FTP-USERNAME = <username>
```

Default Value

```
FTP-USERNAME=
```

Notes

You can also add the `--ftp-username` option to the command line when encrypting or decrypting files.

For example:

```
ebs --encrypt --ftp --ftp-username <user name>
```

GROUPSFILE

Specifies the location of the E-Business Server groups file, `pgpgroup.pgr`.

Default Value

Unix

```
GROUPSFILE = "<HOME>/.pgp/pgpgroup.pgr"
```

Windows NT

```
GROUPSFILE = "<USERPROFILE>\Personal\pgp\pgpgroup.pgr"
```

Windows 2000

```
GROUPSFILE = "<USERPROFILE>\My Documents\pgp\pgpgroup.pgr"
```

Notes

The <HOME> and <USERPROFILE> portions of the paths must be replaced with the value of the current environment variables of the same names.

HASH

Defines the hash algorithm preference used for signing and encrypt-and-sign operations using an RSA Legacy key.

This setting has no effect for RSAv4 or DH/DSS keys.

Default Value

```
HASH = MD5
```

Values for HASH are as follows:

- MD5
- SHA1
- RIPEMD160
- SHA256
- SHA384
- SHA512

HASHNUM

The HASHNUM parameter is only supported for compatibility purposes. A warning appears if your configuration file contains this setting. Use the HASH parameter instead. For more information, see *HASH* above.

INFO

The INFO parameter controls the amount of detail you receive from E-Business Server diagnostic messages.

Default Value

```
INFO = normal
```

Notes

The settings are as follows:

- Quiet. Only displays error messages.
- Normal. Displays warnings and error messages.

- `Verbose`. Displays verbose information—helpful messages along with errors and warnings. Use this setting to help diagnose any problems you may have using E-Business Server.
- `Debug`. Displays developer level output in addition to the output produced by the other levels. This level may include the display of internal data, statistics, trace information, and return codes from internal functions. Not recommended for normal use.

INTERACTIVE

The `INTERACTIVE` parameter is only supported for compatibility purposes. A warning appears if your configuration file contains this setting.

ISSUER-DN

Specifies the default root certificate to use when issuing an X.509 certificate. This certificate must be a self-signed X.509 certificate. The `ISSUER-DN` option is used for `--key-sign --x509` operations. This is NEVER used for `--cert-request` or `--cert-retrieve` operations.

The DN specifies the certificate that is used by the issuer of the new X.509 certificate and is placed in the new X.509 certificate.

A key may have more than one X.509 certificate attached to it; therefore, you must also specify the `ISSUER-SERIAL` to uniquely identify the certificate you want to use to issue new certs.

The certificate specified must be a self-signed X.509 certificate.

Default Value

```
ISSUER-DN = ""
```

ISSUER-SERIAL

Use in conjunction with the `ISSUER-DN` option to uniquely identify the default root certificate to use for key signing. For more information, see *ISSUER-DN*

Default Value

```
ISSUER-SERIAL = ""
```

KEYSERVER

Specifies the URL of the default key server. The key server specified by the `KEYSERVER` parameter will be used for any operations involving the key server. You can also set a key server URL on the command line by specifying `--keyserver`.

Notes

The default keyserver is `ldap://keyserver.pgp.com`.

The URL may be in any of the following formats: `ldap://`, `ldaps://`, or `http://URL`. If no method is specified, then `ldap://` is assumed. The default ports (389, 636 and 11371 respectively) are assumed if no port number is specified.

If the key server is not an E-Business Server key server, then use `--keyserver-type` to set the type of server you are using.

KEYSERVER-TYPE

This parameter specifies the type of key server being used during key server operations.

Default Value

```
KEYSERVER-TYPE = PGP
```

Values are as follows:

- `PGP`. Use this option if the server you are connecting to is the E-Business Server Keyserver via LDAP, LDAPS, HTTP, or for interfacing with other HTTP key servers.
- `LDAPPGP`. Use this option if the server you are connecting to is an LDAP or LDAPS server, such as the Netscape Directory Server.
- `LDAPX509`. Use this option if the server you are connecting to is an LDAP-based X.509 server, such as Microsoft's Directory Server.

KEY-SIZE

This parameter sets the default key size used during key generation.

Default Value

```
KEY-SIZE = 2048
```

KEY-TYPE

This parameter sets the default key type for key generation.

Default Value

```
KEY-TYPE = DSS
```

Values are as follows:

- `DSS`. Use this option to set the default key type to Diffie-Hellman/DSS keys.
- `RSA`. Use this option to set the default key type to the new RSA v4 keys.
- `RSA-LEGACY`. Use this option to set the default key type to the older RSA Legacy keys.

LICENSEFILE

This points directly to the E-Business Server license file, `EBusSvr.lic`.

Default Value

```
LICENSEFILE = ""
```

LOCAL-TIMES

If the `LOCAL-TIMES` parameter is enabled, E-Business Server displays dates/times in your local time instead of UTC (Universal Coordinated Time).

Syntax

```
LOCAL-TIMES = on
```

LOG-FILE

This parameter contains the path to where you want E-Business Server to write its logging information. This text file is viewable in the Administration Utility. When the `LOG-FILE` parameter has a setting, you will see a **Log** link in the Administration Utility's main panel when you log in.

Default Value

```
LOG-FILE = " "
```

MARGINALS-NEEDED

The configuration parameter `MARGINALS-NEEDED` identifies the minimum number of marginally trusted introducers required to fully certify a public key on your public keyring. For more information on trusted introducers, see *An Introduction to Cryptography*.

Default Value

```
MARGINALS-NEEDED = 2
```

OVERWRITE

When an output file already exists, this setting instructs E-Business Server to overwrite it without first prompting you for confirmation.

Syntax

```
OVERWRITE = off
```

Notes

- You can set this option on the command line by specifying `--overwrite`.
- In earlier versions, `FORCE` was used to force file overwriting. For compatibility purposes, `FORCE` sets the `--overwrite` flag when legacy mode is enabled.

PAGER

E-Business Server's `--secure-viewer` option lets you view decrypted plaintext output on your screen, one screen at a time, without writing the output to a file. If you want to be able to page backwards, one screen at a time, set `PAGER=less`, and use the `--secure-viewer` option.

Default Value

```
PAGER = ""
```

E-Business Server includes a built-in page display utility. If you prefer to use a different page display utility, use the `PAGER` parameter to identify the utility. The `PAGER` parameter specifies the shell command E-Business Server uses to display a file.

For further details, see *Viewing decrypted plaintext output on your screen* on page 85.

PASSPHRASE-FD

If `PASSPHRASE-FD` is specified, then E-Business Server reads the passphrase from the specified file descriptor. Use this parameter to transmit your passphrase from one program to another. Set the `PASSPHRASE-FD` parameter equal to a file descriptor number.

If `PASSPHRASE-FD` is not specified in the configuration file (and the PGPPASSFD environment variable is also not set), then the passphrase is read from standard input (STDIN).

The `PASSPHRASE-FD` value specified in the configuration file supersedes a value set in the PGPPASSFD environment variable. For more information on your passphrase options, see *Alternative ways to work with passphrases* on page 93.

If you need to supply both your regular E-Business Server passphrase as well as a conventional passphrase in a single operation, then you may use the `CONVENTIONAL-PASSPHRASE-FD` parameter. For more information, see *CONVENTIONAL-PASSPHRASE-FD* on page 120.

PASS-THROUGH

When `PASS-THROUGH` is set to `on`, E-Business Server does not generate errors when a decryption operation encounters non-E-Business Server data.

Default Value

```
PASS-THROUGH = off
```

PIN-FD

If `PIN-FD` is specified, then E-Business Server will try to read the smart card PIN number (passphrase) from the specified file descriptor. Use this parameter to transmit a PIN from one program to another. Set the `PIN-FD` parameter equal to a file descriptor number.

This option is only necessary if you must supply both your regular E-Business Server passphrase and a smart card PIN in a single operation. Otherwise, you can use the PASSPHRASE-FD option to supply your passphrase, whether it's your key passphrase, a conventional passphrase, or a smart card PIN number. For more information on this option, see *PASSPHRASE-FD* on page 131.

For more information on the various ways you can supply your passphrase to E-Business Server, see *Alternative ways to work with passphrases* on page 93.

PGP-MIME

Use to specify compatibility with PGP/MIME. The PGP-MIME parameter creates messages in EBS/MIME format.

PGP/MIME format produces both a message "header" and a message "body." The PGP-MIME parameter causes the output to include both of these parts, separated by a blank line.

This parameter should be used in conjunction with the ARMOR parameter.

Default Value

```
PGP-MIME = off
```

PGP-MIMEPARSE

Use to instruct E-Business Server to try to parse MIME body parts. To receive messages in PGP/MIME format properly, both the message "header" and message "body" need to be input to the program.

In some cases, the mail client does not save the message header. Use PGP-MIMEPARSE to tell E-Business Server that only the message body is being provided as input, and the header is missing. E-Business Server attempts to parse the message body and find the PGP/MIME information even though the header is missing.

This method is not completely reliable at this time. It is preferable that you include the MIME headers whenever possible.

Default Value

```
PGP-MIMEPARSE = off
```

PRESERVE-NAME

Retains the name of the originally-encrypted file during a decryption operation. Normally a decrypted file is named by the same name as the encrypted file minus the .pgp extension. If PRESERVE-NAME is set to on, the original filename, encoded in the encrypted file, is used.

Default Value

```
PRESERVE-NAME = off
```

PUBRING

You may want to keep your public keyring in a directory separate from your E-Business Server configuration file (that is, the directory specified by your PGPPATH environment variable or the PGPPATH parameter). Use the PUBRING parameter to identify the full path and filename for your public keyring.

Default Value

Unix

```
PUBRING = "<PGPPATH>/pubring.pkr"
```

For example:

```
PUBRING = ~/mykeyrings/pubkeys.pkr
```

Windows NT

```
PUBRING = "<USERPROFILE>\Personal\pgp\pubring.pkr"
```

Windows 2000

```
PUBRING = "<USERPROFILE>\My Documents\pgp\pubring.pkr"
```

For example:

```
PUBRING = c:\personal\keyrings\public_keys.pkr
```

Notes

The <PGPPATH> and <USERPROFILE> portions of the paths must be replaced with the value of the current environment variables of the same names.

You can also use this feature on the command line to specify an alternative keyring. For example, you might set the value of PUBRING as shown below.

```
ebs --pubring pubkeys.pkr --key-list
```

QUESTION

The QUESTION parameter is used during key reconstruction to specify a prompt. We recommend that you use this parameter with extreme caution. Each user should pick 5 unique questions.

If any questions are specified, then all five questions must be specified by some combination of using the QUESTION parameter in the config file, or by specifying the question on the command line by entering --question.

No questions are set by default.

RANDOM-DEVICE

(UNIX only.) Identifies the system entropy pool, /dev/random. E-Business Server tries to open this device to acquire entropy, and if that fails, will try to acquire entropy from user keystrokes.

Default Value

```
RANDOM-DEVICE = /dev/random
```

RANDSEED

The random number seed file, `randseed.rnd`, is used to generate session keys. You may want to keep your random number seed file in a more secure directory or device (this file generally resides in the directory specified by your `PGPPATH` environmental variable).

Use the `RANDSEED` parameter to identify the full path and filename for your random seed file.

Default Value

Unix

```
RANDSEED = "<PGPPATH>/randseed.rnd"
```

Windows NT

```
RANDSEED = "<SYSTEMROOT>\Profiles\All Users\Application
  Data\Network Associates\pgp"
```

Windows 2000

```
RANDSEED = "<ALLUSERSPROFILE>\Application Data\Network
  Associates\pgp\randseed.rnd"
```

Notes

The `<PGPPATH>`, `<ALLUSERSPROFILE>` and `<SYSTEMROOT>` portions of the paths must be replaced with the value of the current environment variables of the same names.

REVERSE

When set to `on`, the `REVERSE` parameter reverses the sorting order set by the `SORT` parameter during `--key-list` displays.

Default Value

```
REVERSE = off
```

RSAVER

The `RSAVER` parameter has been deprecated. Previously, this option specified the type of RSA key you could generate. In the current version, you can create a key of any key type (DSS, RSAv4 or RSA Legacy) during key generation.

SDA

Use of this setting in the configuration file or as a +OPTION is deprecated. SDA is only supported for compatibility purposes.

Use to instruct E-Business Server to create a Self-Decrypting Archive (SDA) every time the `--encrypt --conventional` option is used. For more information on SDAs, see *Creating Self-Decrypting Archives (SDAs)* on page 83.

Default Value

```
SDA = off
```

SECRING

You may want to keep your secret keyring in a directory separate from your E-Business Server configuration file (that is, the directory specified by your PGPPATH environmental variable or the PGPPATH parameter). Use the SECRING parameter to identify the full path and filename for your secret keyring.

Default Value

Unix

```
SECRING = "<PGPPATH>/secring.skr"
```

Windows NT

```
SECRING = "<USERPROFILE>\Personal\pgp\secring.skr"
```

Windows 2000

```
SECRING = "<USERPROFILE>\My Documents\pgp\secring.skr"
```

Notes

The <PGPPATH> and <USERPROFILE> portions of the paths must be replaced with the value of the current environment variables of the same names.

SECURE-VIEWER

Specifies that the internal viewer be used to view decrypted information.

The internal viewer protects potentially sensitive information from being written to disk while being displayed on your screen after a decryption operation. Setting the SECURE-VIEWER parameter in the configuration file prevents decrypted data from being written to disk and disables the use of the `--output` option. It also forces any encrypted content to carry the "Eyes Only" attribute, requesting that the decrypter not write the information to disk.

Default Value

```
SECURE-VIEWER = off
```

SHOWPASS

Causes E-Business Server to echo your typing during passphrase entry.

Default Value

```
SHOWPASS = off
```

E-Business Server, by default, does not let you see your passphrase as you type it. This makes it harder for someone to look over your shoulder while you type and learn your passphrase. However, you may have problems typing your passphrase without seeing what you are typing or you may feel confident that you have sufficient privacy and do not need to hide your keystrokes.

SIG-TYPE

Applies a type to a signature on a key. Signature types are discussed below.

Default Value

```
SIG-TYPE = exportable
```

You can also set the signature type on the command line by entering the following:

```
ebs --sig-type <type> --key-sign <their_userID> [--sign-with
  <your_userID>] [<keyring>]
```

Values for `<type>` are as follows:

- `Exportable (export)`. Exportable signatures can be exported to a certificate server so other users can view them.
- `Local (non)`. Local (non-exportable) signatures apply only to your keyring. You cannot export local signatures to a certificate server.
- `Meta`. Meta signatures (always non-exportable) bestow meta-introducer status on the key. Any key considered trusted by the meta-introducer is considered a trusted introducer by you, and any key considered valid by the trusted introducer is considered valid to you.
- `Introducer`. Introducer signatures bestow trusted introducer status on the key. Any key considered valid by a trusted introducer is considered valid to you.

SIGNEDBY

Decrypt a file that has been signed by a particular key. If you use this option, you do not need to specify `--authenticate` on the command line, or AUTHENTICATE in the configuration file.

Syntax

```
SIGNEDBY = <keyid>
```

Default Value

```
SIGNEDBY = ""
```

Details

You can also set this on the command line by entering `--signed-by <keyid>`. The `<keyid>` is the eight hex character ID, not the userID.

SIGN-ONLY

Specifies that newly-generated keys are sign-only. You cannot encrypt to sign-only keys. This changes the meaning of the `KEY-SIZE` parameter to refer to the size of the signing key. The `KEY-TYPE` may not be set as `RSA-LEGACY` and the `KEY-SIZE` must be valid for the type of key.

Default Value

```
SIGN-ONLY = no
```

Notes

You can also set this on the command line by entering `--sign-only yes|no`. If `SIGN-ONLY` is set to `yes` in the configuration file and is not overridden on the command line by entering `--sign-only no`, then subkey generation cannot be performed.

SMARTCARD-DLL

(Windows only.) Specifies the DLL to use for support of smart cards other than those E-Business Server explicitly supports. Use this option in conjunction with the `SMARTCARD-TYPE` option.

The DLL must conform to the PKCS11 standard for smart card interface.

Syntax

```
SMARTCARD-DLL = <path to DLL>
```

Default Value

```
SMARTCARD-DLL = ""
```

SMARTCARD-TYPE

(Windows only.) Specifies the type of smart card you want to use. This smart card type is used for all smart card operations. If you want to use a smart card other than one that we have listed as being supported, then you must set the `SMARTCARD-TYPE` to `other`, as well as specify the path to the DLL to use with it by setting the `SMARTCARD-DLL` parameter.

Default Value

```
SMARTCARD-TYPE = ""
```

Values for `SMARTCARD-TYPE` are as follows:

- GemPlus
- Rainbow
- Schlumberger

- Other

SMTP-SERVER

Specifies the SMTP server that you want E-Business Server to connect to during encryption operations. Note that E-Business Server does not support SMTP user name and password for some SMTP server configurations.

Syntax

```
SMTP-SERVER = <host name|IP address>
```

Default Value

```
SMTP-SERVER=
```

Notes

You can also add the `--smtp-server` option to the command line when encrypting files.

For example:

```
ebs --encrypt --smtp --smtp-server <server information>
```

SMTP-PORT

Specifies the port that you want E-Business Server to use when connecting to an SMTP server during encryption operations.

Syntax

```
SMTP-PORT = <port number>
```

Default Value

```
SMTP-PORT =
```

Notes

You can also add the `--smtp-port` option to the command line when encrypting files.

For example:

```
ebs --encrypt --smtp --smtp-port <port number>
```

SMTP-USERNAME

Specifies the user name that you want E-Business Server to use when connecting to an SMTP server during encryption operations. If you do not specify a user name, E-Business Server connects as an anonymous user.

This parameter works with the `SMTP-PASSWORD` parameter. If you do not specify a value for `SMTP-USERNAME`, E-Business Server assumes that you want to connect as an anonymous user. In this case you do not need to specify a value for `SMTP-PASSWORD`. E-Business Server will automatically use `anonymous@` as the password.

Note that E-Business Server does not support SMTP user name and password for some SMTP server configurations.

Syntax

```
SMTP-USERNAME = <username>
```

Default Value

```
SMTP-USERNAME =
```

Notes

You can also add the `--smtp-username` option to the command line when encrypting files.

For example:

```
ebs --encrypt --smtp --smtp-username <user name>
```

SMTP-PASSWORD

Specifies the password that you want E-Business Server to use when connecting to an SMTP server during encryption operations.

This parameter works with the `SMTP-USERNAME` parameter. If you do not specify a value for `SMTP-USERNAME`, E-Business Server assumes that you want to connect as an anonymous user, and you do not need to specify a value for `SMTP-PASSWORD`. In this case, E-Business Server will automatically use `anonymous@` as the password.

Note that E-Business Server does not support SMTP user name and password for some SMTP server configurations.

Syntax

```
SMTP-PASSWORD = <password>
```

Default Value

```
SMTP-PASSWORD =
```

Notes

You can also add the `--smtp-password` option to the command line when encrypting files.

For example:

```
ebs --encrypt --smtp --smtp-password <password>
```

SMTP-SENDER

Specifies the e-mail address that you want E-Business Server to put in the "From:" field when sending encrypted files. This address must use the format `username@domainname.com`.

Syntax

```
SMTP-SENDER = <e-mail address>
```

Default Value

```
SMTP-SENDER =
```

Notes

You can also add the `--smtp-sender` option to the command line when encrypting files.

For example:

```
ebs --encrypt --smtp --smtp-sender <e-mail address>
```

SMTP-RECIPIENT

Specifies the default e-mail address that you want E-Business Server to use when sending encrypted files. This address must use the format `username@domainname.com`.

Syntax

```
SMTP-RECIPIENT = <e-mail address>
```

Default Value

```
SMTP-RECIPIENT =
```

Notes

You can also add the `--smtp-recipient` option to the command line when encrypting files.

For example:

```
ebs --encrypt --smtp --smtp-recipient <e-mail address>
```

SMTP-SUBJECT

Specifies the default subject line that you want E-Business Server to use when e-mailing encrypted files.

Syntax

```
SMTP-SUBJECT = <subject text>
```

Default Value

```
SMTP-SUBJECT=
```

Notes

You can also add the `--smtp-subject` option to the command line when encrypting files.

For example:

```
ebs --encrypt --smtp --smtp-subject <subject text>
```

SMTP-CC

Specifies e-mail addresses to which you want E-Business Server to send copies of encrypted files. The addresses must use the format `username@domainname.com`, and be separated by commas.

Syntax

```
SMTP-CC = <e-mail address>
```

Default Value

```
SMTP-CC =
```

Notes

You can also add the `--smtp-cc` option to the command line when encrypting files.

For example:

```
ebs --encrypt --smtp --smtp-cc <e-mail address>
```

SMTP-BCC

Specifies e-mail addresses to which you want E-Business Server to send copies of encrypted files. These are "blind copies," meaning that the recipients do not see the e-mail addresses of any other message recipients. The addresses must use the format `username@domainname.com`, and be separated by commas.

Syntax

```
SMTP-BCC = <e-mail address>
```

Default Value

```
SMTP-BCC =
```

Notes

You can also add the `--smtp-bcc` option to the command line when encrypting files.

For example:

```
ebs --encrypt --smtp --smtp-bcc <e-mail address>
```

SMTP-NOTE

Specifies the default message text that you want E-Business Server to use when e-mailing encrypted files.

You can also use SMTP-NOTE-FILE to define default message text.

Syntax

```
SMTP-NOTE = <message>
```

Default Value

```
SMTP-NOTE =
```

Notes

You can also add the `--smtp-note` option to the command line when encrypting files.

For example:

```
ebs --encrypt --smtp --smtp-note <subject text>
```

SMTP-NOTE-FILE

Specifies a file that contains the message text that you want E-Business Server to use when e-mailing encrypted files.

You can also use SMTP-NOTE to define default message text.

Syntax

```
SMTP-NOTE-FILE = <filename>
```

Default Value

```
SMTP-NOTE-FILE =
```

Notes

You can also add the `--smtp-notefile` option to the command line when encrypting files.

For example:

```
ebs --encrypt --smtp --smtp-notefile <filename>
```

SORT

Sets the default sorting order when displaying a list of keys, such as in `--key-list` or `--key-detail`. This sorting order is also used when displaying a list of multiple keys matched for operations such as `--keyserver-search` or when `--multi` is needed to allow multiple key matches for other operations.

Syntax

```
SORT = <key_attribute>
```

Default Value

```
SORT = ""
```

Values for `SORT` are as follows:

- Keysize.

- Subkeysize.
- Keyid.
- Userid.
- Trust.
- Validity.
- Creation.
- Expiration.

Notes

If `SORT` is not set, then keys appear in the order found on your keyring. You can use the `REVERSE` parameter to reverse the order set by `SORT`.

STATUS-FD

The `STATUS-FD` parameter is not supported in this version. It is allowed for compatibility purposes, but is ignored.

TEXTMODE

The `TEXTMODE` option has been deprecated and is only supported in Legacy mode. Use `--text` instead.

Causes E-Business Server to assume the plaintext is a text file, not a binary file, and converts the plaintext to *canonical text* before encrypting it. Canonical text has a carriage return and a new line at the end of each line of text.

Default Value

```
TEXTMODE = off
```

Notes

- The configuration parameter `TEXTMODE` is equivalent to the `--text` command line option.
- If you intend to use E-Business Server primarily for email purposes, you may wish to set `TEXTMODE=on`.
- For further details, see *Sending ASCII text files to different machine environments* on page 91.

TMP

Specifies the directory E-Business Server uses for temporary files. If `TMP` is undefined, the temporary files are written in the current directory. If the shell environment variable `TMP` is defined, E-Business Server stores temporary files in the named directory.

Default Value

```
TMP = ""
```

VERBOSE

The `VERBOSE` parameter is only supported for compatibility purposes. A warning appears if your configuration file contains this setting. Use the `--info` option instead. For more information, see *INFO* on page 127.

WARN-ADK

Instructs E-Business Server to warn the user before encrypting to an Additional Decryption Key (ADK).

Typically, encryption to a key containing an ADK happens without the user being informed. The ADK may be a key specified by the `ADK-KEY` parameter or one set by a preference on the public key's "Additional Recipient Request" packet.

Default Value

```
WARN-ADK = no
```

Notes

If `WARN-ADK` is set to `yes`, E-Business Server prompts the user for confirmation that they want to also encrypt to the ADK. If the user answers no, the ADK is not put on the list of recipients and the encryption operation continues.

If `ENFORCE-ADK` and `WARN-ADK` are both set to `yes`, and the user answers no to encrypting to the ADK when prompted for a confirmation, then an error is generated.

WIDTH

The configuration parameter `WIDTH` sets the number of characters allowed on a single line during a key-list display. If you want an unlimited width display, set the width equal to zero (the default).

If the information displayed for a key on your keyring exceeds the number of characters allowed, then the user ID is truncated and a dollar sign ($) appears at the end of the user ID indicating that the additional information was cut. The key information is not wrapped to the next line.

Default Value

```
WIDTH = 0
```

Notes

- You can also set the display width on the command line by specifying `--width <number>` during a `--key-list` operation.
- If the width is set to less than 50 characters, then 50 is used instead.

WIPE

If `WIPE` is enabled, then E-Business Server automatically overwrites and deletes all plaintext files after producing ciphertext files for you. Use the `--wipe` option when a plaintext file contains sensitive information; it prevents someone from recovering the file with a disk block scanning utility.

Default Value

```
WIPE = off
```

You can set the number of times E-Business Server writes over a file by setting the `WIPE-PASSES` configuration option. For more information, see *WIPE-PASSES* on page 145.

WIPE-PASSES

Specifies the number of times E-Business Server should write over a file during `--wipe` operations.

After E-Business Server produces a ciphertext file for you, you can request E-Business Server to automatically overwrite and delete the plaintext file, leaving no trace of plaintext on the disk using the `--wipe` option. This prevents someone from recovering the file with a disk block scanning utility. Typically, E-Business Server wipes over a file 3 times.

Raising this setting may increase security, but may also decrease performance.

Default Value

```
WIPE-PASSES = 3
```

12 Using Command Line Options

This chapter lists the primary E-Business Server command line options, and includes the modifiers you can apply to them. The options are listed in alphabetical order.

Conventions used in this section

Convention	Meaning
angle brackets < >	Angle brackets (<>) indicate a variable. You supply a value of the type indicated.
square brackets [ ]	Square brackets ([]) indicate an option. The value indicated is not required.
[...]	[...] indicates that you can list several of the type, such as file names or user IDs. For example, you might enter several files you want to encrypt to the same key, or several users you want to encrypt one file to.

For command line options that include the **--output** modifier:

If the **--output** modifier is not specified, then files will be placed in the input file location by default.

Primary command line options

armor

Use the `--armor` option to produce ASCII-armored formatted files. This option enables you to transmit binary encrypted data through 7-bit channels, or to send binary encrypted data as normal email text. In the E-Business Server program, ASCII armored text files are given the default .asc filename extension, and they are encoded and decoded in the ASCII radix-64 format. For more information on working with ASCII armored files, see *Encrypting and transmitting binary data* on page 90.

Syntax

```
ebs --armor [--output <filename>]
```

Modifiers

`--output`	Specifies the name of the file where the output should be saved.

cert-request

Use the `--cert-request` option to request an X.509 certificate. You can either specify the PKCS10 output (to standard output or to a particular file) or specify a Certificate Authority (CA) where you would like to request an X.509 certificate. For more information on adding X.509 certificates to your key, see *Working with X.509 Certificates* on page 66.

Syntax

```
ebs --cert-request --pkcs10 | --ca-type <type> --ca-url <url>
  --ca-root-cert <certid> [--output <file>] [--cert-attribute
  <name=value>]
```

Modifiers

`--ca-root-cert`	Specifies the X.509 certificate that represents the CA's root certificate. <certid> is the ID belonging to the X.509 self-signature, which must be on your keyring.
`--ca-type`	Specifies the Certificate Authority. Your options are NetTools, VeriSign, Entrust, iPlanet, or Win2k.
`--ca-url`	Specifies the URL of the Certificate Authority.
`--cert-attribute`	Specifies one or more of the certificate attributes. Common X.509 certificate attributes include, but are not limited to, an email address (E), organization name (O), organizational unit name (OU), or country (C).
`--output`	Specifies the name of the file where the output should be saved.
`--pkcs10`	Specifies that a certificate request will be output as a local PKCS #10 formatted file.

Notes

- Typically, the CA information is set in the E-Business Server configuration file, and only one CA can be specified at a time. A duplicated `--ca-*` setting results in an error message.
- If `--pkcs10` is specified, then the appropriate functions are used to output the PKCS10-formatted certificate request, either to standard output or to a specified file using `--output <filename>`.
- If `--pkcs10` is not specified, then the other `--ca-*` options must be specified either on the command-line or in the E-Business Server configuration file.
- If the `<name=value>` argument contains spaces, then it must be enclosed in quotes. For example, to specify that the organization name is McAfee, you would enter the following:

```
--cert-attribute O="McAfee"
```

cert-retrieve

Use the `--cert-retrieve` option to get an X.509 certificate previously requested from a Certificate Authority (CA). For more information on adding X.509 certificates to your key, see *Working with X.509 Certificates* on page 66.

Syntax

```
ebs --cert-retrieve <keyID> --ca-type <type> --ca-url <url>
  --ca-root-cert <certid>
```

Modifiers

`--ca-root-cert`	Specifies the X.509 certificate that represents the CA's root certificate. <certid> is the ID belonging to the X.509 self-signature, which must be on your keyring.
`--ca-type`	Specifies the Certificate Authority. Your options are NetTools, VeriSign, Entrust, iPlanet, or Win2k.
`--ca-url`	Specifies the URL of the Certificate Authority.

Notes

`<keyID>` represents the you key want to use to request the certificate.

decrypt

Use the `--decrypt` option to specify decryption of previously encrypted data. For more information on decryption, see *Decrypting information* on page 85.

Syntax

```
ebs --decrypt [--passphrase <passphrase>]
  [--allow-passphrase-retry] [--preserve-name] [--secure-viewer]
  [--authenticate] [--ftp --ftp-pathname <remote path>
  [--ftp-secure] [--ftp-port <port number>] [--ftp-server <server
  name>] [--ftp-username <user name>] [--ftp-password <password>]]
  [--signed-by <keyid>] [--output <file>] [filename [...]]
```

Modifiers

`--allow-passphrase-retry`	Tells E-Business Server to abort an encryption operation if the user does not specify a passphrase in the original encryption command, or the supplied passphrase is not correct.
`--authenticate`	Tells E-Business Server not to decrypt a file unless it has been signed.
`--ftp`	Tells E-Business Server to FTP to a server you specify using additional `ftp` options, and then retrieve and decrypt a file stored there. To save the decrypted information, you must specify a file name using the `--output` option. Otherwise E-Business Server decrypts the data and sends it to `stdout`. For a more secure transfer, use `--ftp-secure` instead of `--ftp`. Note that you can specify default values for some `--ftp` modifiers in the E-Business Server configuration file. See *Using the Configuration File* on page 110.
`--ftp-secure`	Tells E-Business Server to do the FTP transfer securely, using the FTPS protocol. This protocol uses Transport Layer Security (TLS) to protect the user name, password, file name and data. The remote FTP server's X.509 certificate must be signed and on your keyring for –ftp-secure to work. E-Business Server only supports servers that can use FTPS and export their X.509 certificates.
`--ftp-port`	Specifies a port to connect to on the target FTP server. If you do not list a port, E-Business Server uses port 21 for normal FTP, or port 990 if you specified `--ftp-secure` for secure FTP.
`--ftp-server`	Specifies the target FTP server's host name or IP address. You must include `--ftp-server` whenever using `--ftp`.
`--ftp-username`	Specifies the user name E-Business Server uses when connecting to the target FTP server. If you do not specify a user name, E-Business Server connects as an anonymous user.
`--ftp-password`	Specifies the password that E-Business Server uses when connecting to the target FTP server. If you do not define a value for `--ftp-username` and `--ftp-password`, then E-Business Server uses `anonymous@`. If you do set a value for `--ftp-username`, but do not set a value for `--ftp-password`, then E-Business Server prompts for a password.

`--ftp-pathname`	Specifies the directory that E-Business Server uses for the transfer, as well as the file name that you want it to decrypt. You must include `--ftp-pathname` whenever decrypting using `--ftp`.
`--output`	Specifies the file where you want the decrypted information to be saved.
`--passphrase`	Specifies the passphrase for your private key, or the passphrase used for conventional encryption. If the passphrase contains spaces, then it must be enclosed in quotes. If you do not specify the passphrase on the command line, then E-Business Server prompts for it.
`--preserve-name`	Retains the name of the originally encrypted file. Normally a decrypted file is named by the same name as the encrypted file minus the .pgp extension. With this setting, the original filename (encoded in the encrypted file) is used.
`--secure-viewer`	Specifies that the internal viewer be used to view the decrypted information protecting it from being written to disk while displayed.
`--signed-by`	Specifies a key (by key ID) that the file must be signed with in order for E-Business Server to decrypt it.

Notes

- You must own the private key. If the private key is a split key, an implicit join is performed; all key-join flags apply in this case.
- If standard input is used, the output is delivered to standard output unless a `--output <file>` is specified. If there are multiple input files, the `--output <file>` must refer to a directory where all decrypted files will be written.
- You can list several files you want to decrypt in a single operation, unless you are using the `--ftp` option.

encrypt

Use the `--encrypt` option to encrypt data either by conventional encryption, to a particular passphrase, or to another user's public key. For more information on encryption see *Encrypting information* on page 78.

Syntax

```
ebs --encrypt --conventional | --sda | --archive | --user <userID>
  [...] [--sign [--sign-with <userID>] [--passphrase <passphrase> |
  --passphrase-fd <file descriptor>] [--allow-passphrase-retry]]
  [--conventional-passphrase <passphrase> |
  --conventional-passphrase-fd <file descriptor>] [--text]
  [--encrypt-to-self] [--wipe [--wipe-passes]] [--warn-adk yes|no]
  [--armor] [--secure-viewer] [--ftp [--ftp-secure]
  [--ftp-port <port number>] [--ftp-server <server name>]
  [--ftp-username <user name>] [--ftp-password <password>]
  [--ftp-pathname <remote path>]] [--smtp
  [--smtp-port <port number>] [--smtp-server <server name>]
  [--smtp-username <user name>] [--smtp-password <password>]
  [--smtp-sender <e-mail>] [--smtp-recipient <e-mail>]
  [--smtp-subject <description>] [--smtp-cc <e-mail>]
  [--smtp-bcc <e-mail>] [--smtp-note <message> |
  --smtp-notefile <message source>]] [--output <file name>
  [--overwrite]] [file ...]
```

Modifiers

`--allow-passphrase-retry`	Tells E-Business Server to abort an encryption operation if the user does not specify a passphrase in the original encryption command, or the supplied passphrase is not correct.
`--archive`	Specifies creation of a PGP archive. PGParchives are the same as SDAs except they don't have the executable stub added to the beginning. Since PGParchives are regular data files, there is no limit to their length. Add the –discard-paths option to strip any relative path information from the files you want to include in the archive. Upon decryption, the files are placed in the current directory, instead of in subdirectories of the current directory.
`--armor`	Specifies ASCII-armored output, which you can send through email channels.
`--conventional`	Specifies encryption to a particular passphrase instead of to a public key.
`--conventional-passphrase`	Specifies the passphrase to be used for conventional encryption (used in conjunction with –conventional, –sda or –archive). If the passphrase contains spaces, then it must be enclosed in quotes.
`--conventional-passphrase-fd`	Specifies the file descriptor for reading the conventional passphrase from a file handle (used in conjunction with –conventional, –sda or –archive).
`--discard-paths`	Strip path information that would be included in a PGParchive or SDA.
`--encrypt-to-self`	Specifies encryption to your own key specified by the DEFAULT-KEY parameter in the configuration file.

`--ftp`	Tells E-Business Server to FTP the encrypted data to a target you specify using additional `ftp` options. E-Business Server does not leave a copy of the encrypted file on the local computer when you use the FTP feature. The only copy is the one sent via FTP to the remote server. For a more secure transfer, add `--ftp-secure` to `--ftp`. Note that you can specify default values for all `--ftp` modifiers in the E-Business Server configuration file. See *Using the Configuration File* on page 110.
`--ftp-secure`	Tells E-Business Server to do the FTP transfer securely, using the FTPS protocol. This protocol uses Transport Layer Security (TLS) to protect the user name, password, file name and data. The remote FTP server's X.509 certificate must be signed and on your keyring for –ftp-secure to work. E-Business Server only supports servers that can use FTPS and export their X.509 certificates.
`--ftp-port`	Specifies a port to connect to on the target FTP server. If you do not list a port, E-Business Server uses port 21 for normal FTP, or port 990 if you specified `--ftp-secure` for secure FTP.
`--ftp-server`	Specifies the target FTP server's host name or IP address. You must include `--ftp-server` whenever using `--ftp` or `--ftp-secure`.
`--ftp-username`	Specifies the user name E-Business Server uses when connecting to the target FTP server. If you do not specify a user name, E-Business Server connects as an anonymous user.
`--ftp-password`	Specifies the password that E-Business Server uses when connecting to the target FTP server. If you do not define a value for `--ftp-username` and `--ftp-password`, then E-Business Server uses `anonymous@`. If you do set a value for `--ftp-username`, but do not set a value for `--ftp-password`, then E-Business Server prompts for a password.

`--ftp-pathname`	Specifies which directory E-Business Server transfers the encrypted data to. E-Business Server uses this pathname to determine a name for the transferred file. If the directory path you use ends with a "/" character, for instance, then E-Business Server creates the final file name by appending the local file name to the specified directory name. For instance, "`/test/ebs/`" would create an encrypted file using the local filename, and located in the `ebs` directory. If the directory path does not end in a "/" character, then E-Business Server takes the last portion of the path and uses that as the final file name. For instance, "`/test/ebs`" would create an encrypted file called `ebs` in the directory `test`.
`--output`	Specifies the file where you want the encrypted information to be saved. With this option, you can also add `--overwrite`, which tells E-Business Server to automatically overwrite an existing file by the same name without prompting. Use –output - to encrypt data to standard out.
`--passphrase`	Specifies the passphrase for signing. If the passphrase contains spaces, then it must be enclosed in quotes.
`--passphrase-fd`	Specifies the file descriptor for reading your passphrase from a file handle.
`--sda`	Creates a self-decrypting archive (SDA) containing the encrypted files. Add the –discard-paths option to strip any relative path information from the files you want to include in the archive. Upon decryption, the files are placed in the current directory, instead of in subdirectories of the current directory.
`--secure-viewer`	Specifies that the recipient's decrypted plaintext be shown only on the recipient's screen and not saved to disk.
`--sign`	Signs a message before encrypting it. By default, the key specified by the DEFAULT-KEY parameter in the configuration file is used.
`--sign-with`	Specifies a specific key you want to sign with. You can specify the user ID or key ID.
`--smtp`	Tells E-Business Server to e-mail the encrypted data to a target you specify using additional `smtp` options. E-Business Server does not leave a copy of the encrypted file on the local computer when you use the SMTP feature. The only copy is the one sent via e-mail to the recipient(s) you specify. Note that you can specify default values for all `--smtp` modifiers in the E-Business Server configuration file. See *Using the Configuration File on page 110*.

`--smtp-port`	Specifies a port to connect to on the your SMTP mail server. If you do not list a port, E-Business Server uses port 25 by default.
`--smtp-server`	Specifies the host name or IP address of your target SMTP mail server. You must include `--smtp-server` whenever using `--smtp`.
`--smtp-username`	(Use only if your SMTP mail server requires authentication.) Specifies the user name that E-Business Server uses to authenticate itself with your SMTP mail server. E-Business Server supports "AUTH LOGIN" authentication.
`--smtp-password`	(Use only if your SMTP mail server requires authentication.) Specifies the password that E-Business Server uses to authenticate itself with your SMTP mail server. E-Business Server supports "AUTH LOGIN" authentication.
`--smtp-sender`	Specifies the e-mail address that E-Business Server uses for the "From:" field of the resulting e-mail message. This address must use the format username@domainname.com. You must include `--smtp-sender` whenever using `--smtp`.
`--smtp-recipient`	Specifies the e-mail address that E-Business Server sends the encrytped data to. The address appears in the "To:" field of the resulting e-mail message. This address must use the format username@domainname.com. You must include `--smtp-recipient` whenever using `--smtp`.
`--smtp-subject`	Provides a short line of text that E-Business Server uses as the subject line of the resulting e-mail message. If you do not use `--smtp-subject`, then E-Business Server will leave the subject line of the e-mail message blank.
`--smtp-cc`	Specifies additional e-mail addresses to which E-Business Server sends copies of the encrytped data. These addresses appear in the "Cc:" field of the resulting e-mail message. Addresses must use the format username@domainname.com, and be separated by commas.

`--smtp-bcc`	Specifies additional e-mail addresses to which E-Business Server sends copies of the encrytped data. Recipients will not see the addresses of any other message recipients; this information is hidden from them. E-Business Server places these addresses in the "Bcc:" field of the resulting e-mail message. Addresses must use the format username@domainname.com, and be separated by commas.
`--smtp-note`	Specifies a message that E-Business Server includes in the body of the resulting e-mail. The encrypted data is sent as an attachment. You can use `--smtp-notefile` instead of `--smtp-note`. If you use neither, E-Business Server sends the e-mail message with the following message text: "E-Business Server encrypted email message."
`--smtp-notefile`	Works similarly to `--smtp-note`, but this option lets you specify the name of a file. This file should contain the text that you want E-Business Server to use in the e-mail message when it sends encrypted data.
`--text`	Specifies that the input data is text and should be converted to canonical new lines.
`--user`	Specifies the key to which a message will be encrypted. Specify a 32-bit or 64-bit key ID or a user ID that identifies the key(s) on your keyring. You can list –user several times, if you want to encrypt to multiple keys at one time.
`--warn-adk`	If set to yes, you are warned before encrypting to an Additional Decryption Key (ADK); If set to no, then E-Business Server does not warn you before encrypting to an ADK.
`--wipe`	Wipes the input file after encryption. With this option, you can also add `--wipe-passes`, which specifies the number of times E-Business Server should write over the file.

Notes

- If no `--user` is specified on the command-line, then E-Business Server prompts the user for a user ID or key ID.
- If you are encrypting to multiple users, then list each of the user IDs or key IDs with a separate `--user` modifier, such as `--user <userID1> --user <userID2> --user <userID3>`.
- If you are encrypting multiple files, then list each of the filenames separated by a space at the end of the string, such as `<file1> <file2> <file3>`. They are all encrypted to the same specified key(s) or passphrase. The output files will be located in the current directory or the directory specified by `--output`.

- If standard input is used, the output is delivered to standard output unless a `--output <file>` is specified. If there are multiple input files, the `--output <file>` must refer to a directory where all decrypted files will be written.
- If you are conventionally encrypting and signing in the same operation, then use `--conventional-passphrase` or `--conventional-passphrase-fd` to specify the passphrase you want to encrypt to, and use `--passphrase` or `--passphrase-fd` to specify the passphrase for your signing key. For more information on various ways to specify your passphrase, see *Alternative ways to work with passphrases* on page 93.

extract-photoid

Use the `--extract-photoid` option to retrieve a photo ID from a key in JPEG format.

Syntax

```
ebs --extract-photoid <userID> [--output <filename> [--overwrite]]
```

Modifiers

`--output`	Specifies the file where you want the encrypted information to be saved. With this option, you can also add `--overwrite`, which tells E-Business Server to automatically overwrite an existing file by the same name without prompting.

Notes

- `<userID>` is the key whose photo ID you are retrieving. This key must be on your keyring.
- If no `--output` filename is specified, a filename is created from the key's primary user ID.

group

Use the `--group` option to display a list of group commands. See *Working with groups* on page 96 for more information.

Syntax

```
ebs --group
```

group-add

Use the `--group-add` option to create a new group, or add members to an existing group. To create a new group, simply specify a name for the new group. To add members to an existing group, then you must specify the group name and the user IDs of the new members. See *Working with groups* on page 96 for more information.

Syntax

```
ebs --group-add <groupname> [<userID1> <userID2> <userID3> ...]
```

group-detail

Use `--group-detail` to list the name and description for each group as well as the members belonging to each group. If you do not specify a group, `--group-detail` shows the contents of all of your groups. See *Working with groups* on page 96 for more information.

Syntax

```
ebs --group-detail <groupname>
```

group-list

Use `--group-list` to display the name and description for each of your groups. See *Working with groups* on page 96 for more information.

Syntax

```
ebs --group-list <groupname>
```

group-remove

Use `--group-remove` to remove members or groups from a group. See *Working with groups* on page 96 for more information.

Syntax

```
ebs --group-remove <groupname> [--multi] --user <userID>
```

Modifiers

`--multi`	Specifies that all user IDs that match the user ID entered should be removed.
`--user`	Specifies the user ID or key ID for the user you want to remove from the group.

help

Use the `--help` option to display help on E-Business Server command options.

Syntax

```
ebs --help [--<command name>]
```

Modifiers

`--<command name>`	Specifies the command name for which you want help. Help is available for most commands described in this chapter.

key-add

Use the `--key-add` option to import keys to your keyring.

Syntax

```
ebs --key-add [--multi] [--x509 [--with-private]] [file ...]
```

Modifiers

`--multi`	Instructs E-Business Server to import all keys from the input files without being prompted for a confirmation.
`--x509`	Imports an X.509 certificate to your keyring and accepts a PEM-encoded certificate file. With this option, you can add –with-private to import a PKCS12-formatted X.509 certificate with its private key.

Notes

Adds keys from standard input or from the filename(s) specified on the command line.

- If the filename extension is `.pem`, then PEM-encoded X.509 certificate format is used.
- If the filename extension is `.pfx`, then PKCS12 X.509 key pair format is used.

key-detail

Use the `--key-detail` option to display information about each matching key such as the creation dates, fingerprints, expiration dates, subkeys, ADKs, Revokers, tokens, or photo IDs present.

Syntax

```
ebs --key-detail [--sort <field>] [--fingerprint-view hex | words]
  [--multi] [--reverse] [userid ...]
```

Modifiers

`--fingerprint-view`	Sets the format for displaying fingerprint information. Specify hex to display the hexadecimal fingerprint, or specify words to display a unique word list that represents the fingerprint.
`--multi`	Tells the program to display information on all matching keys.
`--reverse`	Tells E-Business Server to reverse the sort order of the displayed information.
`--sort`	Specifies the field you want to sort the keys by. By default, E-Business Server sorts in ascending order (a to z). You can sort by any of the following fields: keysize, subkeysize, keyid, userid, trust, validity, creation, expiration. To list the keys in descending order (z to a), include the `--reverse` option.

Notes

When one or more user IDs are specified on the command-line, E-Business Server displays a key list and asks you to either use `--multi`, or select one specific entry.

key-edit

Use the `--key-edit` option to make changes to a key.

Syntax

```
ebs --key-edit <userID_or_keyID> [--add-userid <userID>]
  [--set-primary-userid <userID>] [--trust <trust-level>]
  [--change-passphrase <passphrase> [--new-passphrase
  <passphrase>]] [--allow-passphrase-retry] [--add-revoker
  <keyID>] [--add-photoid <filename>] [--remove-userid <userID>]
  [--remove-photoid <filename>] [--disable] [--enable] [--revoke]
  [--revoke-sig <userID>] [--remove-sig <userID>] [--remove-subkey
  <subkeyID>] [--revoke-subkey <subkeyID>]
```

Modifiers

`--add-photoid`	Adds a photo ID to the specified key where <filename> is the name of the file containing your photo in JPEG format. The standard size for the photo ID is 120 pixels wide by 144 pixels tall.
`--add-revoker`	Adds a designated revoker to your key pair. The revoker must be specified by a key ID and the key must be on your keyring. You will need to re-sign your self-signature.
`--add-userid`	Adds a new user ID (149 characters, maximum) to an existing keypair. For example, you may wish to add a secondary email address to your key.
`--allow-passphrase-retry`	Tells E-Business Server to abort an encryption operation if the user does not specify a passphrase in the original encryption command, or the supplied passphrase is not correct.
`--change-passphrase`	Sets a new passphrase for your E-Business Server key. Optionally, you can include –new-passphrase to specify the new passphrase.
`--disable`	Disables a key on the keyring. The program prompts for confirmation before disabling the key, unless –force is also specified.
`--enable`	Enables a key that was previously disabled.
`--remove-photoid`	Removes the photo ID from your key. The program prompts for confirmation of the removal, unless –force is also specified.
`--remove-sig`	Removes a signature attached to a key on the keyring where the user ID or key ID specified belongs to the signature you want to remove.
`--remove-subkey`	Removes an encryption subkey from an RSA v4 key. If the key ID specified belongs to an RSA v3 key, an error message appears.
`--remove-userid`	Removes a specified user ID from your key. For example, if you change jobs and your email address is different, then you should remove the old email address from your key.
`--revoke`	Revokes a keypair.
`--revoke-sig`	Revokes a signature on a key.
`--revoke-subkey`	Revokes an encryption subkey from an RSA v4 key. If the key ID specified belongs to an RSA v3 key, an error message appears.

`--set-primary-userid`	Sets a new user ID on a key to be the primary user ID.
`--trust`	Specifies the level of trust on a key that bears your signature. You can set one of the following levels of trust: none, marginal, complete or implicit. None indicates no trust. Marginal indicates that you usually trust the owner of the key to act as a trusted introducer. Complete indicates that you fully trust the owner of the key to act as a trusted introducer. Implicit indicates that this is your own key.

key-export

Use the `--key-export` option to export a key from your keyring to a file.

Syntax

```
ebs --key-export <userID_or_keyID> [--armor] [--output <filename>]
  [--multi] [--with-private] [--smartcard [--pin <pin>]]
  [--export-format compatible|complete]
  [--x509 [--issuer-dn <DN> --issuer-serial <number]]
```

Modifiers

`--armor`	Specifies that keys are exported in ASCII-armored format, using .asc instead of .pgp as the extension.
`--export-format`	If you specify "compatible", then E-Business Server strips newer key features from the keys being exported. This may be necessary if the key is to be used by clients with E-Business Server versions prior to 6.0, which does not recognize such attributes as X.509 certificates or photo IDs. Specify "complete" if you do not want to strip features.
`--issuer-dn`	Identifies the X.509 certificate issuer for the certificate you want to export. This is useful with –x509 when there is more than one X.509 certificate on the key.
`--issuer-serial`	Specifies the X.509 certificate issuer's assigned serial number for the certificate you want to export. When used in conjunction with the –issuer-dn option, this uniquely identifies the certificate you want to export.
`--multi`	Allows all keys matching the user ID to be exported without prompting.
`--output`	Specifies the location where the exported keys should be saved. If the output file specified is a directory name, then all keys are exported to that directory in separate files identified by the primary user ID. Files are overwritten if `--overwrite` is also specified. The use of a directory for output is not supported when exporting X.509 certificates because the user IDs can be difficult to read.

`--pin`	Specifies the smart card PIN number.
`--smartcard`	Indicates that the key-export operation takes place on the smart card.
`--with-private`	When specified, all key pairs exported include the private portion of the key. By default, E-Business Server only exports the public portion of your key pair. **Note:** Smart cards cannot export their private keys.
`--x509`	Indicates that only the X.509 certificate associated with the key should be extracted. You must specify the certificate ID if the key you want to export contains more than one X.509 certificate. You cannot specify a directory name for the output filename when using –x509. .crt is appended to the filename if the filename does not contain a period. The key is exported in DER format, unless –armor is used specifying PEM format.

Notes

- If E-Business Server finds several matching keys on your keyring for the specified user ID and `--multi` is not also used, then an error message appears.
- E-Business Server only prompts for the user ID and filename when running in Legacy mode. Otherwise, a missing user ID produces a help message and an error results.
- Unless `--output` is specified, the primary user ID is used to generate an output filename. If that file already exists, an error is generated (unless you also specified `--overwrite`).

key-gen

Use the `--key-gen` option to create a new key pair, or to create an encryption subkey for an existing key.

Syntax

```
ebs --key-gen [--userid <name>] [--subkey] [--key-type <type>]
  [--expires-after <time>] [--keyserver <url>] [--key-size <size>]
  [--passphrase <passphrase>][--sign-only] [--smartcard
  [--smartcard-dll <path> [--smartcard-type <type>]]]
```

Modifiers

`--expires-after`	Specifies the validity period for a signature or a newly generated key where <time> is a number of days or an absolute date in YYYY-MM-DD format. If you specify a start date using the –start-date option, then the number of days follows the start date.
`--keyserver`	Sends the newly generated key to a keyserver. If the <url> parameter is not specified, the default keyserver is used.
`--key-size`	Selects a custom key size where <size> is the number of bits to make the new key. Any size may be entered, with a minimum of 1024 bits and a maximum of 4096 bits. If not specified, then the key will default to the size specified in the configuration file. **Note:** If creating an RSA LEGACY key, the maximum size is 2048 bits.
`--key-type`	Specifies the type of key you want to create. The available key types are DSS, RSA, and RSA-LEGACY. This option is disallowed if –subkey is also specified.
`--passphrase`	Specifies the passphrase for your new key. If the passphrase contains spaces, then it must be enclosed in quotes.
`--sign-only`	When creating a new key, this option specifies that the key is a signing key only without an encryption subkey. If used, then –keysize must specify a valid signing key size (1024 for DSS keys; less than 4096 for RSA v4 keys). **Note:** This option is not valid for RSA LEGACY keys.
`--smartcard`	Specifies that you want to generate the new key on a smartcard.
`--smartcard-dll`	Specifies the path to the dll provided by the manufacturer. This is only necessary if you are using a smartcard type that is not supported by E-Business Server.
`--smartcard-type`	Specifies the type of smartcard you are using. Your options are GemPlus, Rainbow, Schlumberger, or other. If you specify other, then you must also specify the path to the dll using –smarcard-dll.
`--start-date`	When generating a subkey, this option specifies the starting date for the validity period of the subkey. Enter the date in YYYY-MM-DD format. If you do not specify a date, then the current date is used. **Note:** This option is only valid when generating a subkey.

`--subkey`	Specifies that key generation produces a new subkey.
`--userid`	149 characters maximum. When generating a new key, this option identifies the name and email address for the new key. A user ID for a new key should have a name and an email-address in angle brackets. When generating a subkey, this option identifies the keypair to which the new subkey is added.

Notes

- For DH/DSS keys, `--key-size` indicates the encryption key size, which ranges from 1024 bits to 4096 bits with a default value of 2048 bits. The DSS signing key is always 1024 bits.
- For RSAv3 keys, `--key-size` indicates the size of the (only) key, which ranges from 1024 bits to 2048 bits (the default).
- For RSAv4 keys, `--key-size` indicates the size of both the signing key and the encryption subkey, which ranges from 1024 bits to 4096 bits with a default value of 2048 bits. To create an RSA key that has different signing and encryption key sizes, you must first create a sign-only RSAv4 key with the desired signing key size. Then, in a separate operation, create an encryption subkey of the desired size.
- When generating a subkey, the specified key size is the size of the encryption key generated.
- For more information on generating new keys, see *Creating a key pair* on page 26. For more information on generating subkeys, see *Creating subkeys* on page 28. For more information on creating keys on a smart card, see *Creating a key pair on a smart card* on page 29.

key-join

Use the `--key-join` option to restore a previously split key.

Syntax

```
ebs --key-join <userID_or_keyID>
```

Notes

Key joins will only be done as an interactive process. The only parameter to this option is the User ID or the Key ID of the key to be joined. The other information is provided via prompts just like in previous releases.

key-list

Use the `--key-list` option to display keys on a keyring. For more information or examples of the various listing options, see *Key List Displays* on page 205.

Syntax

```
ebs --key-list [--with-sigs | --with-userids] [--sort <field>
  [--reverse]] [userid ...]
```

Modifiers

`--sort`	Specifies the field you want to sort the keys by. By default, E-Business Server sorts in ascending order (a to z). You can sort by any of the following fields: keysize, subkeysize, keyid, userid, trust, validity, creation, expiration. To list the keys in descending order (z to a), include the –reverse option.
`--width`	Sets the number of characters displayed on a line. By default, the key list display is set to an unlimited number of characters. If the information displayed for a key on your keyring exceeds the number of characters allowed, then the user ID is truncated and a dollar sign ($) appears at the end of the user ID indicating that there was more information. The key information is not wrapped to the next line.
`--with-sigs`	Instructs E-Business Server to also display signatures on keys.
`--with-userids`	Instructs E-Business Server to also display all user IDs on keys. The normal key-list view does not show the list of userids on each key. With this option set, more lines of display are used to provide more information to the user.

Notes

- When you specify `--with-sigs`, `--with-userids` is implied.
- If `--width` is set to less than 50 characters, then 50 is used instead.

key-reconstruct

Use the `--key-reconstruct` option to restore a private key, which was previously split into shares, encrypted, and sent to a key reconstruction server.

Syntax

```
ebs --key-reconstruct <userid> [--keyserver <url>] [--passphrase
  <new_passphrase>] [--answer [...]] [--auth-user <userid>
  --auth-passphrase <passphrase>]
```

Modifiers

`--answer`	Specifies an answer to one of the 5 questions supplied when the key was initially sent to the key reconstruction server. You must be able to supply answers for at least 3 of the 5 questions.
`--auth-passphrase`	Specifies the password to login to the generic LDAP server. If the password contains spaces, then it must be enclosed in quotes. This option is not necessary when using an E-Business Server key server.
`--auth-user`	Specifies your login name for logging into a generic LDAP server. This option is not necessary when using an E-Business Server key server.
`--keyserver`	Specifies the key reconstruction server's URL. Enter the keyserver URL in the following format: ldap://<IP address or DNS name of key server>.
`--passphrase`	After reconstructing your key, you must change your passphrase. Use this option to specify your new passphrase for this key.

Notes

- If you specify answers on the command line using the `--answer` modifier, then you must supply at least 3 out of 5 of the answers. If you do not specify any answers on the command line, then E-Business Server lists each of the questions and prompts you for the answers.
- If this function is called from the E-Business Server E-Business Engine without all the answers being provided, E-Business Server displays the 5 questions without prompting you for the answers, and an error is generated.
- For more information on how to reconstruct your key, see *Reconstructing your key* on page 56. For more information key reconstruction and how to send your key to a key reconstruction server, see *What is key reconstruction?* on page 32

key-remove

Use the `--key-remove` option to delete keys from a keyring.

Syntax

```
ebs --key-remove <userID> [--force] [--multi] [--with-private]
  [--smartcard]
```

Modifiers

`--force`	Forces deletion of the matching key without prompting for confirmation.
`--multi`	Specifies the deletion of all matching keys. If multiple keys match the user ID and –multi is not specified, then an error appears.
`--smartcard`	Indicates that the key you want to delete resides on a smartcard.
`--with-private`	Specifies that the private portion of the key pair is also deleted.

key-sign

Use the `--key-sign` option to digitally sign a key or create an X.509 certificate.

Syntax

```
ebs --key-sign <userID_or_keyID> [--multi] [--sig-type <type>]
  [--sign-with <keyid>] [--passphrase <passphrase>]
  [--allow-passphrase-retry] [--expires-after <expiration>]
  [--depth] [--x509 [--issuer <DN> [--issuer-serial <number>
  [--cert-attribute <name=value> [--start-date <date>]]]]]
```

Modifiers

`--allow-passphrase-retry`	Tells E-Business Server to abort an encryption operation if the user does not specify a passphrase in the original encryption command, or the supplied passphrase is not correct.
`--cert-attribute`	Adds certificate attributes to the certificate you are creating.
`--depth`	Specifies how many levels deep you can set trust for a meta or trusted introducer signature.
`--expires-after`	Specifies the number of days your signature is considered valid or a future date when your signature should expire. Enter the date in YYYY-MM-DD format. By default, the signature never expires.
`--issuer-dn`	Identifies the certificate issuer's distinguished name.
`--issuer-serial`	Identifies the issuer's assigned serial number for the certificate. When used in conjunction with the –issuer-dn option, this uniquely identifies the certificate.
`--multi`	Signs all keys that match the user ID.
`--passphrase`	Specifies the passphrase used for key signing.

`--regexp`	Specifies a regular expression to attach to your signature. If the regular expression contains spaces, then you must enclose it in quotes. Note that you may need to escape special shell characters.
`--sign-with`	Selects the key you want to sign with. By default, E-Business Server uses the key specified by the DEFAULT-KEY parameter in the E-Business Server configuration file.
`--sig-type <type>`	Specifies the type of signature you want to add to the key you are signing. Your options are: local (non) | exportable (export) | meta | introducer (trusted). By default, the signature type specified by the SIG-TYPE parameter in the E-Business Server configuration file is used.
`--start-date`	This option is only valid when creating X.509 certificate signatures. Specifies a future date when your signature becomes valid. By default, this is the creation date. Enter a future date in YYYY-MM-DD format.
`--x509`	Indicates that you want to create an X.509 certificate signature instead of a regular signature.

Notes

- The `--issuer-dn`, `--issuer-serial` and `--cert-attribute` options are only valid when `--x509` is also specified.
- Certificate attributes are entered in *name=value* format. *Name* represents the type of attribute you want to define, such as Email (E), OrganizationName (O), or Location (L). *Value* represents your definition for the corresponding attribute. If the value contains spaces, then you must enclose it in quotes. For example, O="McAfee" indicates that the organization that owns the certificate is McAfee. You can list several certificate attributes when creating X.509 certificates. Simply precede each name=value pair with `--cert-attribute`. For more information, see *Specifying certificate attributes on page 67*.
- For more information on signing keys, see *Signing a key* on page 63. For more information on creating X.509 certificates, see *Issuing X.509 certificates* on page 73.

key-split

Use the `--key-split` option to split a private key into shares. This is recommended for extremely high security keys. For more information on splitting keys, see *Creating a split key* on page 50.

Syntax

```
ebs --key-split <userID_or_keyID>
```

key-update

Use the `--key-update` option to update keys on your local keyring from a key server. E-Business Server searches the specified key server or generic LDAP server for all keys on your local keyring and merges the matching keys back into your keyring.

Syntax

```
ebs --key-update [--keyserver <url>] [--adk | --keys | --revokers |
  --introducers | --x509 | --crl] [userid ...]
```

Modifiers

`--adk`	Updates and adds Additional Decryption Keys (ADKs) associated with keys on your keyring. If ADK-KEY is set in the E-Business Server configuration file, then that key is also updated or added to your local keyring.
`--crl`	Downloads the latest certificate revocation list from the –ca-revocation-url and merges any new revocations onto the keyring.
`--introducers`	Specifies that E-Business Server updates or adds introducer keys to your keyring for all keys with meta-introducer signatures on them. E-Business Server searches your local keyring for keys with valid meta-introducer signatures. Then, E-Business Server searches the key server for all keys signed by this set of introducer keys and all matching keys are added to your keyring.
`--keys`	E-Business Server searches the specified key server or generic LDAP server for all keys on your local keyring and merges the matching keys back into your keyring.
`--keyserver`	Specifies the URL for the key server or generic LDAP server that you want to compare your keyring to. Enter the keyserver URL in the following format: ldap://<IP address or DNS name of key server>.
`--revokers`	Specifies that all designated revoker associated with keys on your keyring are also updated from the key server. If the designated revoker's key is not currently on your keyring, E-Business Server adds it from the key server.
`--x509`	Specifies that all keys with X.509 signature certificates associated with them are updated from the key server. Any revocations found on the key server are merged into the key.

keyserver-delete

Use the `--keyserver-delete` option to delete a key from a keyserver.

Syntax

```
ebs --keyserver-delete [--sign-with <userid>] [--passphrase
  <quoted-passphrase>] [--force]
  --keyserver <url> userid ...
```

Modifiers

`--force`	Forces key deletion of all matching keys without first prompting for confirmation.
`--keyserver`	Specifies the URL for the key server or generic LDAP server that you want to delete your key from. Enter the keyserver URL in the following format: ldap://<IP address or DNS name of key server>
`--passphrase`	Specifies the passphrase for your private key.
`--sign-with`	Selects the key you want to sign the deletion request with. By default, E-Business Server uses the key specified by the DEFAULT-KEY parameter in the E-Business Server configuration file.

Notes

If the keyserver URL specifies a TLS connection, the signing key is used to authenticate the client to the server at the connection protocol layer, instead of for signing the delete request sent through the connection.

keyserver-disable

Use the `--keyserver-disable` option to temporarily disable keys on a keyserver.

Syntax

```
ebs --keyserver-disable [--sign-with <userid>] [--passphrase
  <quoted-passphrase>] [--force] --keyserver <url> userid ...
```

Modifiers

`--force`	Forces key disabling of all matching keys without first prompting for confirmation.
`--keyserver`	Specifies the URL for the key server or generic LDAP server that you want to disable your key from. Enter the keyserver URL in the following format: ldap://<IP address or DNS name of key server>.
`--passphrase`	Specifies the passphrase for your private key.
`--sign-with`	Selects the key you want to sign with. By default, E-Business Server uses the key specified by the DEFAULT-KEY parameter in the E-Business Server configuration file.

Notes

If the keyserver URL specifies a TLS connection, the signing key is used to authenticate the client to the server at the connection protocol layer, instead of for signing the disable request sent through the connection.

keyserver-fetch

Use the `--keyserver-fetch` option to get a key (or keys) from a keyserver and import them to your local keyring.

Syntax

```
ebs --keyserver-fetch [--keyserver <url>] [--add-all] [--key-detail
  [--with-userids|--with-sigs]] userid ...
```

Modifiers

`--add-all`	Adds all matching keys without any prompting. If not specified, then each matching key is displayed in turn, and E-Business Server prompts for confirmation that you want to import each key.
`--key-detail`	Lists more information for all matching keys on the keyserver such as creation dates, fingerprints, expiration dates, subkeys, ADKs, Revokers, tokens, or photo IDs present.
`--keyserver`	Specifies the URL for the key server or generic LDAP server that you want to disable your key from. Enter the keyserver URL in the following format: ldap://<IP address or DNS name of key server>.
`--with-sigs`	Instructs E-Business Server to also display signatures on keys. This option implies –with-userids.
`--with-userids`	Instructs E-Business Server to also display all user IDs on keys. The normal key-list view does not show the list of userids on each key. With this option set, more lines of display are used to provide more information to the user.

keyserver-search

Use the `--keyserver-search` option to search a keyserver for key (or keys) and display all matching keys.

Syntax

```
ebs --keyserver-search [--keyserver <url>] [--key-detail]
  [--with-userids] [--with-sigs] userid ...
```

Modifiers

`--key-detail`	Lists more information for all matching keys on the keyserver such as creation dates, fingerprints, expiration dates, subkeys, ADKs, Revokers, tokens, or photo IDs present.
`--keyserver`	Specifies the URL for the key server or generic LDAP server that you want to search. Enter the keyserver URL in the following format: ldap://<IP address or DNS name of key server>.
`--with-sigs`	Instructs E-Business Server to also display signatures on keys. This option implies –with-userids.
`--with-userids`	Instructs E-Business Server to also display all user IDs on keys. The normal key-list view does not show the list of userids on each key. With this option set, more lines of display are used to provide more information to the user.

keyserver-send

Use the `--keyserver-send` option to send keys to the keyserver.

Syntax

```
ebs --keyserver-send --keyserver <url> [--multi] userid [...]
```

Modifiers

`--keyserver`	Specifies the URL for the key server or generic LDAP server that you want to send your key(s) to. Enter the keyserver URL in the following format: ldap://<IP address or DNS name of key server>.
`--multi`	Sends all keys that match the user ID. If not specified, the list of matching keys appears and no keys are sent to the server.

list-aliases

Use the `--list-aliases` option to display the current active aliases. For information on adding aliases to the E-Business Server configuration file, see *ALIAS* on page 112.

Syntax

```
ebs --list-aliases
```

reconstruct-data

Use the `--reconstruct-data` to generate data for later key reconstruction. You need to provide recovery information—five questions and five secret answers—and send your key to the key reconstruction server.

For more information on creating reconstruction data, see *What is key reconstruction?* on page 32.

Syntax

```
ebs --reconstruct-data <userid> [--auth-user <login>
  [--auth-passphrase <passphrase>]] --keyserver <url> [--question
  ...] [--answer ...]
```

Modifiers

`--answer`	Specifies an answer. You must provide 5 answers. Your answers can be up to 255 characters in length are case sensitive.
`--auth-passphrase`	Specifies the password to login to the generic LDAP server. If the password contains spaces, then it must be enclosed in quotes. This option is not necessary when using an E-Business Server key server.
`--auth-user`	Specifies your login name for logging into a generic LDAP server. This option is not necessary when using an E-Business Server key server.
`--keyserver`	Specifies the URL for the key server or generic LDAP server that you want to send your key(s) to. Enter the keyserver URL in the following format: ldap://<IP address or DNS name of key server>.
`--question`	Specifies a question. You must provide 5 questions. Your question can be up to 95 characters in length.

Notes

- If you want to specify the questions and answers on the command line, then you must supply all 5 questions and answers.
- If you do not specify the questions and answers on the command line, then E-Business Server prompts for this information.

send-share

Use the `--send-share` option to send local key shares to a remote hostname in order to restore a split key.

Syntax

```
ebs --send-share <share_filename> [--hostname <hostname>]
  [--auth-user <userID> [--auth-passphrase>] [--passphrase]
```

Modifiers

`--auth-passphrase`	The passphrase for the private key of the key pair specified by the –auth-user option. If not supplied, E-Business Server prompts you for it.
`--auth-user`	Specifies the user that is used to authenticate the client side of the TLS connection used to join the key. This option must refer to a key pair, and the key must be trusted by the remote side.
`--hostname`	Specifies the IP address or DNS name for the remote computer the share file is being sent from.
`--passphrase`	If the share-file was encrypted using public key encryption, then passphrase refers to the passphrase for the private portion of the key pair it was encrypted to. If the share-file was encrypted using conventional encryption, then the passphrase refers to the conventional passphrase used for the encryption. If not supplied, E-Business Server prompts you for it.

Notes

If the remote host is not listening on the standard SKEP port (14747), an error appears. This port cannot be changed.

sig-detail

Use the `--sig-detail` option to display more information about signatures on keys on your keyring.

Syntax

```
ebs --sig-detail [--userid <userid>] [--signer <userid>]
  [--multi] userid ...
```

Modifiers

`--multi`	Displays signature details for multiple keys that match the –userid. If not specified, the list of matching keys appears in –key-list format.
`--signer`	Displays details of signatures by the specified signer.
`--userid`	Displays details for signatures on the key you specify.

sign

Use the `--sign` option to perform a cryptographic signature on a plaintext file using your default private key. For more information on signing data, see *Signing information on page 58*.

Syntax

```
ebs --sign [--detached] [--armor] [--text] [--clearsig]
  [--sign-with <userid>] [--passphrase <passphrase>]
  [--allow-passphrase-retry]
```

Modifiers

`--allow-passphrase-retry`	Tells E-Business Server to abort an encryption operation if the user does not specify a passphrase in the original encryption command, or the supplied passphrase is not correct.
`--armor`	Specifies ascii-armored output. The output is in a base-64-encoded format called ascii-armoring. This makes the output safe for sending through 7-bit email or other systems.
`--clearsig`	Used in conjunction with –armor and –text, produces a clear-signed message, one that can be read with human eyes, and without the aid of E-Business Server.
`--detached`	This instructs E-Business Server to produce a separate, detached signature certificate file. The signature is not attached to the text that was signed. Typically, the signature file has a .sig extension added to the input filename.
`--passphrase`	Specifies the passphrase used for key signing.
`--sign-with`	Specifies the key you want to use for the signing operation.
`--text`	Specifies that the input is text that should be converted to canonical new lines before signing. Trailing whitespace is ignored with this setting.

version

Displays version information about the E-Business Server executable. If the `INFO` configuration parameter (or `--info` on the command line) is set to `verbose`, then additional information is displayed, such as build stage, build number, and whether debugging information is included.

Syntax

```
ebs --version [--info <level>]
```

Modifiers

`--info`	Controls the amount of detail you receive from E-Business Server diagnostic messages.

wipe

Use the `--wipe` option to perform a secure deletion of files. For more information on wiping files, see *Wiping a sensitive data file* on page 92.

Syntax

```
ebs --wipe [--multi] [--wipe-passes <number>] [--smartcard [--pin
  <PIN> [--smartcard-type <type> [--smartcard-dll]]]] file [...]
```

Modifiers

`--multi`	Specifies that you want to delete multiple files.
`--pin`	Specifies the PIN (or passphrase) for your smartcard.
`--smartcard`	Indicates that you want to wipe all data and keys from your smartcard.
`--smartcard-dll`	Specifies the path to the dll of the smartcard. This is only necessary if you are using a smartcard type that is not supported by E-Business Server.
`--smartcard-type`	Specifies the type of smartcard you are using. Your options are GemPlus, Rainbow, Schlumberger, or other. If you specify other, then you must also specify the path to the dll using –smarcard-dll.
`--wipe-passes`	Specifies the number of times E-Business Server should write over a file. By default, E-Business Server wipes a file 3 times.

13 Using the E-Business Server Administration Utility

About the SupportingProductName X.X

The ePolicy Orchestrator software is an optional console that provides a graphical user interface for E-Business Server — specifically its key management and configuration management features.

The ePolicy Orchestrator software consists of two components — a service, and a console. The service resides on the same computer as E-Business Server, but you can install the console on any Windows computer. This means that you can use the ePolicy Orchestrator software to access and configure E-Business Server remotely.

For instructions on installing the ePolicy Orchestrator software, see the *E-Business Server Installation Guide*.

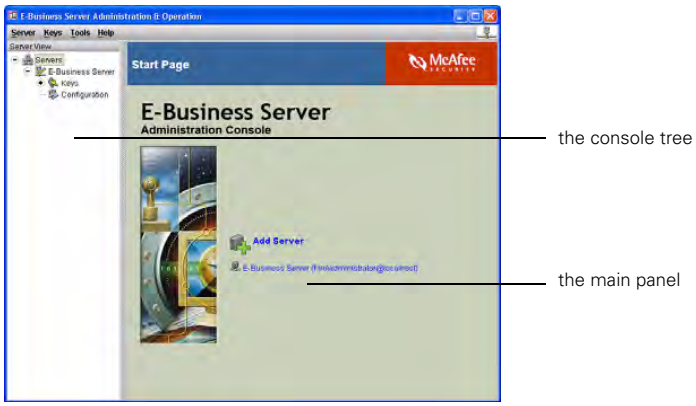
The ePolicy Orchestrator software interface

The ePolicy Orchestrator software console consists of two main areas — the console tree, and the main panel. The console tree lists any servers you are connected to, with **Keys** and **Configuration** nodes available for each one.

The main panel is the area where you work with E-Business Server. It provides links to the **Keys** and **Configuration** views. When you select one of these links, the main panel changes to show the related view. The **Keys** view contains a key list, and a Key Properties area. The **Configuration** view displays a series of tabs, each relating to an E-Business Server feature that you can configure.

If you enable logging, both the console tree and main panel include **Log** entries as well. See *Enabling logging* on page 191 for more information.

Figure 13-1 The ePolicy Orchestrator software console



Getting Started with the SupportingProductName X.X console

Once you have installed the ePolicy Orchestrator software console, you must:

1 Start the ePolicy Orchestrator software console.

2 Add your E-Business Server to the utility's server list.

3 Connect to your E-Business Server.

You can then use the console to create and manage keys, and configure E-Business Server.

Starting the ePolicy Orchestrator software

To start the Administration Utility console, go to the Windows **Start** menu and select **Programs** > **McAfee E-Business Server** > **Administration Client**.

If the Administration Utility console does not start, make certain that you have the Java Runtime Environment (JRE) installed The console requires this software. See the *E-Business Server Installation Guide* for details.

Adding a server

1 In the ePolicy Orchestrator software's main panel, click **Add Server**.

The **Server Details** dialog box appears.

2 In the **Description** field, enter a short identifier for your E-Business Server.

This name will appear in the console tree.

3 Do one of the following:

If	then
you are running the console on a different computer than E-Business Server	enter the IP address for your E-Business Server in the **Server Address** field.
you are running the console on the same computer as E-Business Server	select the **Local machine** checkbox.

4 In the **Username** field, type the login user name of the account you want to manage on your E-Business Server computer.

5 If necessary, type the name of the domain that your E-Business Server computer resides on in the **Domain** field.

6 Click **OK**.

The ePolicy Orchestrator software adds this server to its console tree list.

Connecting to a server

1 In the ePolicy Orchestrator software, select your E-Business Server from the list in the console tree.

2 In the main panel, click **Connect**.

3 When the ePolicy Orchestrator software prompts you for a password, enter the login password for your E-Business Server computer and click **OK**.

At this point, you may see a license-related error message. This message appears if you are using E-Business Server for the first time, and have made no changes to its configuration file. It indicates that E-Business Server cannot find its license file.

If you do not see this error message, the ePolicy Orchestrator software connects to the server you selected. Skip the rest of this procedure.

4 Click **OK**.

5 In the main panel, click on **Configuration**.

6 On the **Files** tab, locate the **License file** field.

7 Click **Browse**, and navigate to the location of your `EBusSvr.lic` file.

If E-Business Server is on a different computer, the **Browse** button will not work. In this case, determine the location of your `EBusSvr.lic` file on your E-Business Server computer, and type it in the **License file** field.

The `EBusSvr.lic` file is located in the directory where you installed E-Business Server.

8 Click **Apply**.

9 Click **Yes** when the console asks you to verify your configuration change, then click **OK** when it confirms the change.

10 Using the ePolicy Orchestrator software console, disconnect from the server and then reconnect.

Your changes will not take effect until you reconnect. You should no longer receive license-related errors,

A new **Keys** option appears in the ePolicy Orchestrator software main panel.

Disconnecting from a server

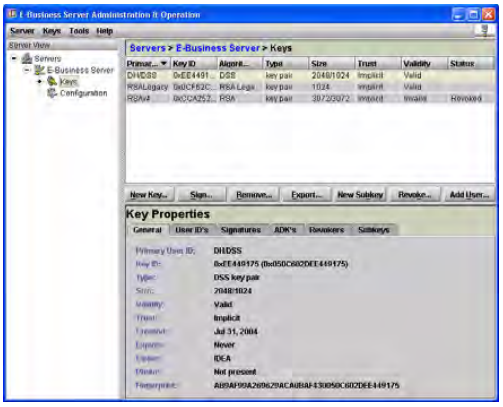
1 In the ePolicy Orchestrator software, select your E-Business Server from the list in the console tree.

2 From the **Server** menu, select **Disconnect**.

Accessing the E-Business Server key management tools

To use the ePolicy Orchestrator software to create and manage keys, first log on to your E-Business Server. The console displays several options in its main panel. Click **Keys** to see the key view. The console shows a key list and a key properties area. The key list shows all the key pairs currently managed by your E-Business Server. You can work with these, or create new key pairs. The key properties area displays detailed information for the key pair you select from the key list.

Figure 13-2 The ePolicy Orchestrator software console key view



Creating a new key pair

1 In the ePolicy Orchestrator software, select your E-Business Server from the list in the console tree.

2 In the main panel, click **Keys**.

The main panel displays a list of keys and key properties.

3 From the **Keys** menu, select **Create new key-pair**.

The **Create New Key-Pair** dialog appears.

4 From the **Key type** list, select one of **DH/DSS**, **RSA v4**, or **RSA Legacy**.

5 Using the **Key size** field, either type a valid key size or select a key size from the list.

6 In the **User ID** field, type the user name that you want to associate with this key pair.

7 In the **Passphrase** field, type the passphrase that you want to associate with the key pair. Type this passphrase again in the **Re-type passphrase** field, in order to confirm it.

8 In the **Options** area:

The **Options** fields are not available if you selected **RSA Legacy** as your key type.

- Use the **Default cipher** list, select the cipher that you want to use to encrypt the session key—**CAST5**, **3DES**, **Twofish**, **AES128**, **AES192**, **AES256**, or **IDEA**.
- Use the **ADK** list to select a key that you want to use as an Additional Decryption Key (ADK).

  For more information, see *ADK-KEY* on page 111.
- Select the **Expires on** checkbox if you want this key to expire on a specific date, and select a date.
- Select the **Sign-only key** checkbox if you only want to use the new key for signing purposes.

If you select this option, you will not be able to use the key for encrypting data.

9 Click **OK** to generate your key pair.

10 Click **OK** when the ePolicy Orchestrator software finishes generating your key pair.

The new key pair appears in your ePolicy Orchestrator software key list. To see details, select if from the list. Details appear in the **Key Properties** area.

Signing a key

1 From the **Keys** menu, select **Sign**.

The **Sign Key** dialog box appears.

2 From the **Key** list, select the key you want to sign.

3 From the **Sign with** list, select the key you want to use for signing.

4 In the **Passphrase** field, type the passphrase for the signing key.

5 In the **Options** area:

- Select an option from the **Signature Type** list: **Local**, **Exportable**, **Meta**, or **Introducer**.

  If you chose **Meta** or **Introducer**, use the **Depth** field to enter the number of nested introducer levels you want to allow. (See *DEPTH* on page 121 for more information.)

- Select the **Expires on** checkbox if you want the signature to expire on a specific date, and select a date.

- In the **Regular Expression** field, enter the regular expression that you want to attach to this signature.

  See *Attaching Regular Expressions to Signatures* on page 203 for more information.

6 Click **OK** to sign the key.

7 Click **OK** when the ePolicy Orchestrator software finishes signing the key.

Deleting a key

1 Select the key you want to delete from your key list.

2 From the **Keys** menu, select **Remove**.

3 Click **Yes** when the ePolicy Orchestrator software prompts you for confirmation.

4 Click **OK** when the software verifies that it deleted the key.

Exporting a key

1 Select the key you want to export from your key list.

2 From the **Keys** menu, select **Export**.

The **Export Key** dialog box appears.

3 If necessary, select a different key to export from the **Key to export list**.

4 Select **Complete** or **Compatible** from the **Export format** list.

See *EXPORT-FORMAT* on page 123 for descriptions of these options.

5 In the **Options** section:

- Select the **Export private key** option to include the private portion of your key pair. If you do not select this option, E-Business Server only exports the public key.

- Select the **Armored file** checkbox to export the key in ASCII-armored format, using .asc instead of .pgp as the extension.

6 Click **OK**.

The **Save** dialog box appears.

7 Navigate to the directory where you want to save the exported key file, then click **Save**.

8 Click **OK** when the ePolicy Orchestrator software verifies that it exported the key.

Creating a subkey

You cannot add a subkey to an RSA Legacy key.

1 From your key list, select the key you want to add the subkey to.

2 From the **Keys** menu, select **Create a new subkey**.

The **Create Subkey** dialog box appears.

3 If necessary, use the **Master key** list to select a different key to add the subkey to.

4 In the **Key size** field, type or select a key size for the subkey.

5 In the **Passphrase** field, type a passphrase for the subkey.

6 In the **Lifetime** area:

- Select the **Start date** checkbox if you want the subkey to become valid on a calendar date. Specify the date.
- Select the **Expiry date** checkbox if you want the subkey to become invalid on a calendar date. Specify the date.

7 Click **OK**.

8 Click **OK** when the ePolicy Orchestrator software verifies that it created the subkey.

Subkeys do not appear in the main key list. To see the subkey you just created, select the **Subkeys** tab in the **Key Properties** area. ePolicy Orchestrator software displays a list of all subkeys for the key you currently have selected.

Adding a new User ID

1 From your key list, select the key you want to add a user ID to.

2 From the **Keys** menu, select **Add new User ID**.

The **Add New User ID** dialog box appears.

3 If necessary, use the **Key to edit** list to select a different key to add the user ID to.

4 In the **Options** section:

- Type the user ID in the **User ID to add** field.
- Type a passphrase to associate with this user ID in the **Passphrase** field.

5 Click **OK**

6 Click **OK** when the ePolicy Orchestrator software verifies that it created the user ID.

To see the user ID you just created, select the **User ID's** tab in the **Key Properties** area. ePolicy Orchestrator software displays a list of all user IDs for the key you currently have selected.

Revoking a key

1 From your key list, select the key you want to revoke.

2 From the **Keys** menu, select **Revoke**.

The **Revoke Key** dialog box appears.

3 If necessary, use the **Key to revoke** list to select a different key to revoke.

4 In the **Revoke with** section:

- Use the **Key** list to select the key you want to use for revoking.
- Type the revoking key's passphrase in the **Passphrase** field.

5 Click **OK**

6 Click **OK** when the ePolicy Orchestrator software verifies that it revoked the key.

In your key list, **Revoked** appears in the **Status** column for this key.

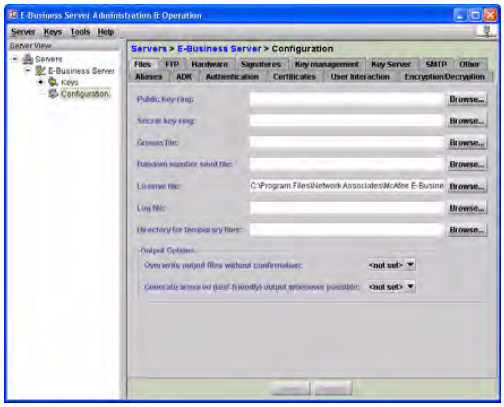
Accessing the E-Business Server configuration tools

E-Business Server's configuration file lets you specify default values and settings so that you do not have to enter these manually with each operation. For information on the configuration file, see *Using the Configuration File* on page 110.

The ePolicy Orchestrator software lets you define many of these configuration settings from the console. To do this, you use its configuration view.

To access the configuration view, first log on to your E-Business Server. The console displays several options in its main panel. Click **Configuration** to see the configuration view. The console shows several tabs , each corresponding to a major E-Business Server function or feature.

Figure 13-3 The ePolicy Orchestrator software configuration view



- To configure E-Business Server, fill in the values on each tab. Click **Apply** to enforce your changes.

The following sections describe the configuration parameters you can set up on each tab in the configuration view.

The Key management tab

Use the **Key management** tab to establish default settings for the keys and key pairs that E-Business Server creates.

Field	Command-line equivalent	See	Use to
Default key	ebs –default-key <key ID> (Works for next signing only.)	■ *DEFAULT-KEY* on page 120	Specify a specific key that you want to use for signing.
Default key size	`ebs --key-gen --key-size`	■ *KEY-SIZE* on page 129 ■ *key-gen* on page 163	Define a custom key size, in bits. The minimum key size is 1024 bits. The maximum key size is 4096 bits. If you do not specify a key size, E-Business Server uses a key size specified in the configuration file. **Note:** If you set the **Default key type** to RSA LEGACY, the maximum key size is 2048 bits.
Default key type	`ebs --key-gen --key-type`	■ *KEY-TYPE* on page 129 ■ *key-gen* on page 163	Specify the type of key you want E-Business Server to create by default. Choose from **DSS**, **RSA**, and **RSA-LEGACY**.
Default key export format	`ebs --key-export`	■ *EXPORT-FORMAT* on page 123 ■ *key-export* on page 162	Specify whether or not you want exported keys to be compatible with earlier versions of E-Business Server (which involves removing newer key features). Choose **Compatible** to make new keys backward-compatible, or **Complete** to export keys with all features intact.
Use fast key-generation method	`ebs --key-gen --fast-key-gen`	■ *FASTKEYGEN* on page 123	Turn fast key generation on or off. Select **Yes** to enable the feature, or **No** to disable it. With this setting enabled, E-Business Server generates **DSS** keys using "canned primes" for common key sizes (currently 1536, 2048, 3072, 4096). This speeds up key generation. This option has no effect if you choose **RSA** or **RSA-Legacy** for **Default key type**.
Generate sign-only keys by default	`ebs --key-gen --sign-only`	■ *SIGN-ONLY* on page 137 ■ *key-gen* on page 163	Specify that new keys should be for signing only (that is, you will not be able to use them for encrypting data). This option has no effect if you choose **RSA-Legacy** for your **Default key type**.
Question prompt for key reconstruction	`ebs --reconstruct -data <userid> [--question ...]`	■ *QUESTION* on page 133 ■ *reconstruct-data* on page 173	Specify five questions that E-Business Server will associate with any keys it generates. These keys are for later key reconstruction, if necessary. Users must answer three of the five questions in order to reconstruct their keys. Question can be up to 95 characters in length.

The Key Server tab

Use the **Key Server** tab to establish default settings for the keyserver that you want E-Business Server to connect to.

Field	Command-line equivalent	See	Use to
Default key server	`--keyserver` (Must be used in conjunction with other commands.)	■ *KEYSERVER* on page 128	Define the URL of the keyserver you want E-Business Server to use by default.
Default key server type	`--keyserver-type` (Must be used in conjunction with other commands.)	■ *KEYSERVER-TYPE* on page 129	Specify the type of keyserver you want E-Business Server to use by default. The choices include **PGP**, **LDAPPGP**, and **LDAPX509**.
--keyserver-fetch adds keys to keyring	`ebs --keyserver-fetch`	■ *keyserver-fetch* on page 172	Get a key (or keys) from a keyserver and import them to your local keyring.

The SMTP tab

Use the **SMTP tab** to configure E-Business Server to send encrypted data by e-mail.

The command-line equivalent is `ebs --encrypt --smtp`. Each field on this tab corresponds to an additional switch for this command. See *encrypt* on page 151 for more information.

Field	Command-line equivalent	See	Use to
Default SMTP server	`--smtp-server <server name>`	*SMTP-SERVER* on page 138	Specify the host name or IP address of your SMTP e-mail server.
Default SMTP port	`--smtp-port <port number>`	*SMTP-PORT* on page 138	Specify a port to connect to on the your SMTP e-mail server. If you do not list a port, E-Business Server uses port 25 by default.
Default SMTP username	`--smtp-username`	*SMTP-USERNAME* on page 138	Specify the user name that E-Business Server uses to authenticate itself with your SMTP mail server.
Default SMTP password	`--smtp-password`	*SMTP-PASSWORD* on page 139	Specify the password that E-Business Server uses to authenticate itself with your SMTP e-mail server.
Default sender	`--smtp-sender`	*SMTP-SENDER* on page 139	Specify the e-mail address that E-Business Server uses for the "From:" field of e-mail messages. Use the format username@domainname.com.
Default recipient	`--smtp-recipient`	*SMTP-RECIPIENT* on page 140	Specify the e-mail address that E-Business Server sends encrytped data to. The address appears in the "To:" field of e-mail messages. Use the format username@domainname.com.
Default subject line	`--smtp-subject`	*SMTP-SUBJECT* on page 140	Provide a short line of text that E-Business Server uses as the subject line of e-mail messages.

Field	Command-line equivalent	See	Use to
Default CC line	`--smtp-cc`	*SMTP-CC* on page 141	Specify additional e-mail addresses to which E-Business Server sends copies of encrytped data. These addresses appear in the "Cc:" field of the e-mail messages. Use the format username@domainname.com. Separate multiple addresses with commas.
Default BCC line	`--smtp-bcc`	*SMTP-BCC* on page 141	Specify additional e-mail addresses to which E-Business Server sends copies of encrytped data. Recipients do not see the addresses of any other message recipients; this information is hidden from them. These addresses appear in the "Bcc:" field of e-mail messages. Use the format username@domainname.com. Separate multiple addresses with commas.
Default note	`--smtp-note`	*SMTP-NOTE* on page 141	Provide a message for E-Business Server to include in the body of e-mail messages. The encrypted data is sent as an attachment to the message.
Default note file	`--smtp-notefile`	*SMTP-NOTE-FILE* on page 142	(Similar to **Default note file**.) Browse to a file that contains the text you want E-Business Server to include in the body of e-mail messages (when it sends encrypted data).

The Other tab

Use the **Other** tab to set miscellaneus software defaults.

Field	Command-line equivalent	See	Use to
Default cipher algorithm	—	■ *CIPHER* on page 117	Specify which cipher E-Business Server should use to when generating a new key pair, when changing the self-signature or passphrase on your private key, and when performing conventional encryption. Your choices include: **IDEA**, **Triple-DES**, **CAST**, **AES**, and **Twofish**.

Field	Command-line equivalent	See	Use to
Command-line version	—	■ *CMDLINE-FORMAT* on page 119	Make E-Business Server translate options from previous releases to those used by the current software release. Select **Legacy** to do this, or **Long** to use only current version options.
Default expiry-period (in days)	`--expires-after` (Must be used in conjunction with other commands.)	■ *EXPIRES-AFTER* on page 122 ■ *key-gen* on page 163 ■ *key-sign* on page 168	Specify the number of days your keys or signatures will be valid for. **Note:** When using the command-line version (`--expires-after`), you can specify an expiry date. When you use the ePolicy Orchestrator software, however, you can only specify expiry as a number of days.

The Encryption/Decryption tab

Use the **Encryption/Decryption** tab to configure default settings for encryption and decryption operations.

Field	Command-line equivalent	See	Use to
Only decrypt a file if it has been signed	`--authenticate`	■ *AUTHENTICATE* on page 114	Decrypt files only if they have been signed.
Comment (for armored files)	`--comment`	■ *COMMENT* on page 119	Add a comment to an armored file.
Discard paths when creating an archive or an SDA	`ebs --encrypt --discard-paths`	■ *DISCARD-PATHS* on page 121 ■ *encrypt* on page 151	Strip any relative path information from the list of files you want to include in a Self-Decrypting Archive (SDA) or PGParchive.
Encrypt to self	`ebs --encrypt --encrypt-to-self`	■ *ENCRYPT-TO-SELF* on page 122 ■ *encrypt* on page 151	Add the recipient specified in the configuration parameter `DEFAULT-KEY` to a list of recipients for an encrypted file. This generally ensures that you have the ability to decrypt anything that you encrypt for others, since the `DEFAULT-KEY` value is usually your own.
Default hash algorithm	–	■ *HASH* on page 127	Define the default hash algorithm for signing and encrypt-and-sign operations using an RSA Legacy key.
Pass through non E-Business Server data while decrypting	`--pass-through`	■ *PASS-THROUGH* on page 131	Tell E-Business Server not to generate errors if it encounters non-E-Business Server data when decrypting.
PGP-MIME compatibility	—	■ *PGP-MIME* on page 132	Specify compatibility with PGP/MIME. The PGP-MIME parameter creates messages in EBS/MIME format.
Try to parse MIME body parts	—	■ *PGP-MIMEPARSE* on page 132	Instruct E-Business Server to try to parse MIME body parts. To receive messages in PGP/MIME format properly, both the message “header” and message “body” need to be input to the program.

Field	Command-line equivalent	See	Use to
Preserve original filename when decrypting	`ebs --decrypt --preserve-name`	■ *PRESERVE-NAME* on page 132 ■ *decrypt* on page 148	Keep the name of the originally-encrypted file during a decryption operation.
Only decrypt files signed by this key	`--signed-by`	■ *SIGNEDBY* on page 136 ■ *decrypt* on page 148	Only decrypt files that have been signed by a particular key.
Wipe temporary plaintext files after encryption	`ebs --encrypt --wipe`	■ *WIPE* on page 145 ■ *encrypt* on page 151	Automatically overwrite and delete all plaintext files after producing ciphertext files.
Number of wipe passes	`ebs --wipe --wipe-passes <number>` `ebs --encrypt --wipe --wipe-passes`	■ *WIPE-PASSES* on page 145 ■ *encrypt* on page 151	Specify the number of times E-Business Server should write over a file during `--wipe` operations.

The Files tab

Use the **Files** tab to identify the locations of key E-Business Server files and folders. You also use this tab to enable logging for the ePolicy Orchestrator software console.

Field	Command-line equivalent	See	Use to
Public key-ring	`--pubring`	■ *PUBRING* on page 133	Identify the full path and filename for your public keyring.
Secret key-ring	`--secring`	■ *SECRING* on page 135	Identify the full path and filename for your secret keyring.
Groups file	—	■ *GROUPSFILE* on page 126 ■ *encrypt* on page 151	Specify the location of the E-Business Server groups file, `pgpgroup.pgr`.
Random number seed file	—	■ *RANDSEED* on page 134 ■ *encrypt* on page 151	Identify the full path and filename for your random seed file.
License file	—	■ *LICENSEFILE* on page 130	Specify the directory where E-Business Server's license file is stored.
Log file	—	■ *LOG-FILE* on page 130	Specify the directory and filename for E-Business Server to use for logging.
Directory for temporary files	`--tmp`	■ *TMP* on page 143 ■ *encrypt* on page 151	Specify the directory E-Business Server uses for temporary files.

Field	Command-line equivalent	See	Use to
Overwrite output files without confirmation	`--overwrite`	■ *OVERWRITE* on page 130 ■ *encrypt* on page 151 ■ *extract-photoid* on page 157 ■ *key-export* on page 162	Force E-Business Server to overwrite existing output files without prompting for confirmation.
Generate armored (text-friendly) output whenever possible	`ebs --armor`	■ *ARMOR* on page 114 ■ *encrypt* on page 151	Make E-Business Server emit ciphertext or keys in ASCII-armored format suitable for transport through email channels.

Enabling logging

1 On the **Files** tab, enter a path and file name in the **Log file** field.

2 Click **Apply**.

3 Disconnect from your E-Business Server, and then reconnect.

A new **Log** item appears on the main panel, and in the console tree.

The FTP tab

Use the FTP tab to set up FTP parameters for transferring encrypted files.

Field	Command-line equivalent	See	Use to
Default FTP server	`--ftp-server`	*FTP-SERVER* on page 125 *decrypt* on page 148 *encrypt* on page 151	Specify a target FTP server's host name or IP address.
Default FTP port	`--ftp-port`	*FTP-PORT* on page 125 *decrypt* on page 148 *encrypt* on page 151	Specify a port for E-Business Server to connect to on the target FTP server. If you do not list a port, E-Business Server uses port 21 for normal FTP, or port 990 if you enable the Use Secure FTP option.
Default FTP path	`--ftp-pathname`	*FTP-PATHNAME* on page 124 *decrypt* on page 148 *encrypt* on page 151	Specify which directory E-Business Server transfers encrypted data to.
Use Secure FTP	`--ftp-secure`	*FTP-SECURE* on page 125 *decrypt* on page 148 *encrypt* on page 151	Tell E-Business Server to do an FTP transfer securely, using the FTPS protocol. This protocol uses Transport Layer Security (TLS) to protect the user name, password, file name and data. The remote FTP server's X.509 certificate must be signed and on your keyring for –ftp-secure to work.
Default FTP username	`--ftp-username`	*FTP-USERNAME* on page 126 *decrypt* on page 148 *encrypt* on page 151	Specify a user name for E-Business Server to use when connecting to a target FTP server. If you do not specify a user name, E-Business Server connects as an anonymous user.
Default FTP password	`--ftp-password`	*FTP-PASSWORD* on page 124 *decrypt* on page 148 *encrypt* on page 151	Specify the password that E-Business Server uses when connecting to a target FTP server. If you do not specify a password, E-Business Server uses `anonymous@`.

The Hardware tab

Use the **Hardware** tab to establish default settings for operations involving smartcards.

Field	Command-line equivalent	See	Use to
SmartCard type	`--smartcard-type`	*SMARTCARD-TYPE* on page 137	Define the type of smart card you want to use.
SmartCard PIN file descriptor number	`--pin-fd`	*PIN-FD* on page 131	Specify a file descriptor. Tells E-Business Server to read the smart card PIN number (passphrase) from the specified file descriptor.
SmartCard DLL	`--smartcard-dll`	*SMARTCARD-DLL* on page 137	Specify the DLL that you want E-Business Server to use for unsupported smart cards. (Use with **SmartCard type**.)

The Signatures tab

Use the **Signatures** tab to establish default signature types, minimum trust levels, and other key signature settings.

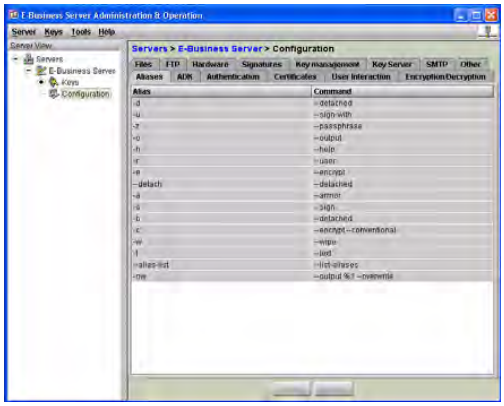
Field	Command-line equivalent	See	Use to
Number of complete signatures needed	`--completes-needed`	*COMPLETES-NEEDED* on page 120	Identify the minimum number of *completely* trusted introducers required to fully certify a public key on your public keyring.
Number of marginal signatures needed	`--marginals-needed`	*MARGINALS-NEEDED* on page 130	Identify the minimum number of *marginally* trusted introducers required to fully certify a public key on your public keyring.
Default signature type	`--sig-type`	*SIG-TYPE* on page 136	Define the default signature type of keys: **Local**, **Exportable**, **Meta**, or **Introducer**.
Clear-sign default	`--clearsig`	*CLEARSIG* on page 118	Generate signed messages that can be read without the aid of E-Business Server.
Maximum trust depth for Meta or Trusted introducers	`--depth`	*DEPTH* on page 121	Set the number of levels of trust E-Business Server accepts for meta and trusted introducer signatures.

The Aliases tab

The **Aliases** tab lets you create, manage, and delete aliases. These are shortcuts for common command line options. Entries must start with at least one dash, and any spaces must be surrounded by quotation marks.

E-Business Server displays its default aliases on this tab.

Figure 13-4 The Aliases tab



To add a new alias

1. On the **Aliases** tab, right-click anywhere in the list of existing aliases.
2. Select **New** from the resulting menu.

   The **Alias Entry** dialog box appears.
3. In the **Alias** field, type the shortcut that you want to use for a specific command.
4. Scroll through the **All Commands** list and select the command for which you want to create an alias.
5. Click the arrow button to move this command to the **Command(s)** field.
6. Click **OK**.

   E-Business Server adds the new alias to the list.

To edit an alias

You cannot edit the shortcut for an alias — you can only change the command that the alias refers to.

1. On the **Aliases** tab, select the alias that you want to edit .

   Right-click the alias, and select **New** from the resulting menu.
2. The **Alias Entry** dialog box appears.
3. Edit the command shown in the **Command(s)** field.
4. Click **OK**.

To delete an alias

1. On the **Aliases** tab, select the alias that you want to delete .
2. Right-click the alias, and select **Remove** from the resulting menu.

The ADK tab

Use the **ADK** tab to configure how E-Business Server treats Additional Decryption Keys (ADKs).

Field	Command-line equivalent	See	Use to
Enforce encrypting to ADKs	`--enforce-adk`	*ENFORCE-ADK* on page 122	Makes encryption attempts fail if you specify a default ADK (using the **Default ADK** field or `--adk-key` command) and E-Business Server cannot find the ADK on your key ring.
Default ADK	`--adk-key`	*ADK-KEY* on page 111	Specifies a specific Additional Decryption Key (ADK) to use when encrypting messages and generating keys.
Warn when encrypting to and ADK	`--warn-adk`	*WARN-ADK* on page 144	Make E-Business Server warn users before encrypting to an Additional Decryption Key (ADK).

The Authentication tab

Use the **Authentication** tab to define E-Business Server's default behavior when it performs authentication operations.

Field	Command-line equivalent	See	Use to
Remote Server Username	`--auth-user`	*AUTH-USER* on page 115	Specify a user ID for E-Business Server to use when authenticating with a remote user.
Remote Server Password	`--auth-passphrase`	*AUTH-PASSPHRASE* on page 115	Specify a password for E-Business Server to use when authenticating with a remote user.
Passphrase file descriptor number	`--passphrase-fd`	*PASSPHRASE-FD* on page 131	Make E-Business Server read a passphrase from the specified file descriptor.
Conventional passphrase file descriptor number	`--conventional-passphrase-fd`	*CONVENTIONAL-PASSPHRASE-FD* on page 120	Make E-Business Server read a passphrase from the specified file descriptor. Used for conventionally encrypting files.
Allow passphrase retries	`--allow-passphrase-retry`	—	Make E-Business Server allow users more than one attempt at entering their passphrase.

The Certificates tab

Use the **Certificates** tab to set up a Certificate Authority (CA) and basic certificate attributes.

Field	Command-line equivalent	See	Use to
Certificate Revocation List URL	`--ca-revocation-url`	*CA-REVOCATION-URL* on page 115	Define the URL used to fetch the Certificate Revocation List (CRL) from the CA.
Root Certificate	`--ca-root-cert`	*CA-ROOT-CERT* on page 115	Specify the key ID of the root Certificate Authority's X.509 certificate.

Field	Command-line equivalent	See	Use to
Certificate Authority URL	`--ca-url`	*CA-URL* on page 116	Define the default URL used to connect to the Certificate Authority (CA).
Certificate Authority Type	`--ca-type`	*CA-TYPE* on page 116	Identify the type of Certificate Authority (CA) that E-Business Server uses.
Certificate Attributes for new signatures / certificates	`--cert-attribute`	*CERT-ATTRIBUTE* on page 116	Specify certificate attributes that E-Business Server will always attach to certificate requests and X.509 signatures. See *Supported Certificate Attributes* on page 212 for more information.
Maximum nesting level of trusted introducers	`--cert-depth`	*CERT-DEPTH* on page 117	Define the maximum number of levels of nested trusted introducers.
Issuer DN	`--issuer-dn`	*ISSUER-DN* on page 128	Specify the default root certificate to use when issuing an X.509 certificate.
Issuer Serial Number	`--issuer-serial`	*ISSUER-SERIAL* on page 128	Identify the default root certificate to use for key signing.(Use with **Issuer DN**.)
VeriSign challenge passphrase file descriptor number	`--challenge-fd`	*CHALLENGE-FD* on page 117	Specify a file descriptor for supplying the challenge passphrase used by Verisign for its certificate revocation process.

The User Interaction tab

Use the **User Interaction** tab to control how E-Business Server displays information.

Field	Command-line equivalent	See	Use to
Use secure viewer	`--secure-viewer`	*SECURE-VIEWER* on page 135	Specify whether or not E-Business Server should use its built-in viewer to view decrypted information.
Secure viewer's name	`--pager`	*PAGER* on page 131	Make E-Business Server use a page display tool other than its built-in secure viewer. Use this option to specify the shell command used to display files.
View fingerprints as	`--fingerprint-view`	*FINGERPRINT-VIEW* on page 123	Specify whether E-Business Server should use Hex or Words for displaying fingerprint information.
Display local times	`--local-times`	*LOCAL-TIMES* on page 130	Make E-Business Server display dates and times in your local time instead of in Universal Coordinated Time (UTC).
When listing keys, sort by	`--sort`	*SORT* on page 142	Set the default sort order when displaying lists of keys.
Reverse sort order	`--reverse`	*REVERSE* on page 134	Reverses the sorting order defined by the `--sort` command, or by the **When listing keys, sort by** field.
Show passphrase when entering	`--show-pass`	*SHOWPASS* on page 136	Display the text that users enter, when typing in their passphrases.
Display width (specify '0' for unlimited)	`--width`	*WIDTH* on page 144	Set the number of characters shown on a single line in a display.

Field	Command-line equivalent	See	Use to
Runtime information level	`--info`	*INFO* on page 127	Control the amount of detail you receive from E-Business Server diagnostic messages.
Suppress warnings / confirmations	`--force`	*FORCE* on page 124	Eliminate interaction with E-Business Server in certain situations, and suppress warnings and confirmations.

Setting Preferences

Changing the ePolicy Orchestrator software's appearance

You can pick from two color schemes for the ePolicy Orchestrator software interface.

To change color scheme

1 From the ePolicy Orchestrator software **Tools** menu, select **Preferences**.

The **Preferences** dialog box appears.

2 In the **User Interface Style** section, choose one of the following styles:

Style	Description
Decorated (default)	The **Decorated** color scheme is based mostly on greys and neutral colors.
Standard	The **Standard** color scheme resembles that used by ePolicy Orchestrator, with a tan-colored background.

3 Click **OK**.

4 Click **OK** again.

5 Disconnect from your server and restart the ePolicy Orchestrator software to see the change.

Changing the default connection port

1 From the ePolicy Orchestrator software **Tools** menu, select **Preferences**.

The **Preferences** dialog box appears.

2 In the **Default Agent Port** field, type the port number that you want the ePolicy Orchestrator software console to use.

The default port is 1718.

3 Click **OK**.

SECTION 2

Appendices and Index

A Command Line Reference

Key options

The `--help --key` option lists all the key management functions available in E-Business Server.

`--key` is also used in combination with other options. The following table lists and describes these combinations, and lists the corresponding legacy options.

Long Option	Legacy Option	Description
`--help --key`	`-k`	Displays help on key options.
`--import --key-add`	`-ka`	Adds keys to the keyring.
`--key-check`	`-kc`	Checks signatures.
`--key-edit`	`-ke`	Edits your user ID, passphrase, trust options, default signing key, or adds a designated revoker for your secret key.
`--key-edit --disable`	`-kd`	Disables keys on the keyring.
`--key-edit --revoke`	`-kd`	Revokes keys on the keyring.
`--key-edit --revoke-sig`	`-kds`	Revokes signatures attached to keys on the keyring.
`--key-gen`	`-kg`	Generates a new key.
`--key-edit --split`	`-kl`	Creates a split key.
`--key-join`	`-kj`	Reconstitutes a split key.
`--key-join --network`	`-kq`	Reconstitutes a split key remotely, or over a network.
`--key-edit --remove-sig`	`-krs`	Removes signatures attached to keys on the keyring.
`--key-edit --remove-userid`		Removes a user ID from the keyring.
`--key-sign`	`-ks`	Signs keys on the keyring.
`--key-sign --expires-after`	`-ksx`	Signs keys on the keyring and adds an expiration date to your signature.
`--key-list`	`-kv`	Lists the keys on the keyring.
`--key-detail`	`-kvc`	Display key information, such as the key's fingerprint.
`--key-list --with-userids`	`-kvv`	List all user IDs on keys on the keyring.
`--key-list --with-sigs`	`-kvv`	List all signatures on keys on the keyring.

Long Option	Legacy Option	Description
`--key-export`	`-kx`	Extract keys from the keyring to exchange with others.
`--key-export --armor`	`-kxa`	Extract keys from the keyring in ASCII-armored format, which makes it easy to paste into email

Email and file options

The following table identifies and describes E-Business Server's command line options used to encrypt, decrypt, and manage files, and lists the corresponding legacy options.

Long Option	Legacy Option	Description
`--armor`	`-a`	Use ASCII-armored format for encrypt, sign, and export operations
`--conventional`	`-c`	Use conventional (symmetric) encryption, not using public-key operations. Only applies to –encrypt operations.
`--encrypt`	`-e`	Specify that an encryption operation is performed. By default, this is a public-key operation unless –conventional is specified in the config file. (–no-conventional entered on the command line overrides the config file)
`--text`	`-t`	Specifies that the plaintext be treated as text instead of binary data; linefeeds will be canonified on encryption and converted to appropriate local-system linefeeds on decryption.
`--sign`	`-s`	Sign the input with your secret key.
`--wipe`	`-w`	Automatically overwrite and delete the original plaintext file preventing someone from recovering it.
none (Simply don't enter input filename(s) and/or output filename(s).)	`-f`	Use Unix-style filter mode to read from standard input and write to standard output
none (Specify input using long options provided.)	`-i`	Specifies a file containing all the input required by E-Business Server when used in conjunction with full batch mode.
`--secure-viewer`	`-m`	On decryption, output displays plaintext output on your screen. On encryption, specifies that the output should be for your eyes only use.
`--output --overwrite`	`-o`	When used with other options such as encryption, decryption, checking signatures, and filter mode specifies the output filename.
`--preserve-name`	`-p`	Recovers the original plaintext filename
`--detached`	`-b`	Detach a signature from the file being signed. Applicable for sign operations, but not for encrypt and sign operations.

Long Option	Legacy Option	Description
`--sign-with`	`-u`	Identifies the key that will be used for the signing operation. Applies to --sign operations or --key-sign operations.
`--passphrase`	`-z`	Indicates that the text string which follows is your passphrase. A leading space can be used; quotes may be required; quote-escaping may also be required.

Keyserver options

The `--help --keyserver` option lists all the key server functions available in E-Business Server.

`--keyserver` is also used in combination with other options. The following table lists and describes these combinations.

Long Option	Legacy Option	Description
`--help --keyserver`		Displays help on key server operations.
`--keyserver-delete`		Deletes a key from the key server.
`--keyserver-disable`		Disables a key on the key server.
`--keyserver-fetch`		Fetches a key (or keys) from the key server and imports them to the local keyring.
`--keyserver-search`		Searches the key server and displays matching keys.
`--keyserver-send`		Sends keys to the key server.

Group options

The `--help --group` option displays a list of group options.

The `--group` option is always used in combination with another option. The following table lists these combinations, describes how they are used, and lists the corresponding legacy options.

Long Option	Legacy Option	Description
`--help --group`	`-g`	Displays help on group options.
`--group-add`	`-ga`	Adds recipients or groups to a group's membership.
`--group-remove`	`-gr`	Removes members from a group.
`--group-list`	`-gv`	View all groups.
`--group-detail`	`-gvv`	View a group and the recipients or groups it contains.

Help options

The `--help` option displays a quick command usage for E-Business Server.

`--help` can also be used in combination with other options. The following table lists these combinations, describes how they are used, and lists the corresponding legacy options.

Long Option	Legacy Option	Description
`--help`	`-h`	Displays a quick command usage for E-Business Server.
`--help --key`		Displays help on key operations.
`--help --key-edit`		Displays help on key edit operations.
`--help --keyserver`		Displays help on key server operations.
`--help --group`	`-g`	Displays help on group operations.
`--help --x509`		Displays help on X.509 operations.
`--help --smartcard`		Displays help on smartcard operations.
`--help <primary option>`		Displays command syntax for the primary option you specify.

B Attaching Regular Expressions to Signatures

This appendix describes the purpose of attaching a regular expression to a signature, lists the special characters used in a regular expression, and defines the regular expression syntax used in E-Business Server.

Attaching a regular expression to a signature

The purpose of a regular expression on a signature is to restrict the scope of the target key's signature power. For example, a corporate administrator might place a signature with an attached regular expression on a sub-administrator, who controls the HR department, that states that he/she can only sign keys from "hr.mcafee.com".

The following special characters can be used in a regular expression:

- a pipe (|)
- parenthesis ( )
- an asterisk (*)
- a plus sign (+)
- a question mark (?)
- brackets [ ]
- a period (.)
- a caret (^)
- a dollar sign ($)

When using any of these characters in a regular expression, put a backslash (\) in front of a literal character to distinguish it from one of the special characters.

For example, the following regular expression matches any email address from test.com, such as <user@test.com>.

```
<.*@test\.com>
```

Definitions of the regular expression syntax used in E-Business Server

- A *regular expression* is zero or more branches separated by a pipe (|). The regular expression matches anything that matches one of the branches.
- A *branch* is zero or more pieces, concatenated. The branch looks for a match for the first piece, then looks for a match for the second piece, etc.
- A *piece* is an atom possibly followed by one of these characters: an asterisk (*), a plus sign (+), or a question mark (?).
  - An atom followed by an asterisk (*) matches a sequence of 0 or more matches of the atom.
  - An atom followed by a plus sign (+) matches a sequence of 1 or more matches of the atom.
  - An atom followed by a question mark (?) matches a match of the atom or the null string.
- An *atom* can be a regular expression in parentheses ( ), a single character, or one of the following special characters. The atom matches a part of the userID as a sub expression within the larger regular expression, assuming that the beginning of the regular expression has already matched.
  - A period (.) represents matching any single character.
  - A caret (^) represents matching the null string at the beginning of the input string.
  - A dollar sign ($) represents matching the null string at the end of the input string.
  - A backslash (\) followed by a single character represents matching that character.
- A *range* is a sequence of characters enclosed in brackets [ ]. The range normally matches any single character from the sequence.
  - If the sequence begins with a caret (^), it matches any single character not from the rest of the sequence.
  - Two characters in a sequence separated by a dash (-) represents the full list of ASCII characters between them. For example, [0-9] matches any decimal digit.
  - To include a literal end bracket (]) in the sequence, make it the first character (following a possible ^). To include a literal dash (-) in the sequence, make it the first or last character.

C Understanding Key List Displays

This appendix shows examples of each of the `--key-list` display options, and defines the column headings and flags in a key list view.

For information on viewing your keys and key management, see *Managing Keys* on page 38.

Key List Displays

There are several variations of the `--key-list` option. You may simply want to display all the keys on your keyring, or you may want to display more information about each key such as additional user IDs or signatures associated with them.

By default, keys are unsorted on your keyring. If you want to sort the keys by a key attribute, such as the key's user ID or key ID, specify the field you want to sort by using the `SORT` parameter in the configuration file, or by specifying `--sort` on the command line.

```
ebs --key-list --sort <field>
```

Your field options are: keysize, subkeysize, keyid, userid, trust, validity, creation and expiration. The information appears in ascending order (a to z). Specify `--reverse` to list the information in descending order (z to a).

By default, the key list display is set to an unlimited number of characters. You can limit the number of characters displayed on a line by setting the `WIDTH` parameter in the configuration file (or `--width` on the command line) equal to the number of characters you want to display.

If the information displayed for a key on your keyring exceeds the number of characters allowed, then the user ID is truncated and a dollar sign ($) appears at the end of the user ID indicating that there was more information. The key information is not wrapped to the next line.

If the width is set to less than 50 characters, then 50 is used instead.

Example of --key-list option

The following example shows the contents of a keyring with its contents labeled:

```
Alg   Type   Size       Flags     Key ID      User ID

RSA   pair   1024       [-----]   0x7D75EB0F  Albert Reilly <arielly@domain.com

DSS   pub    2048/1024  [-----]   0xF5ED0CB   Corporate Key <CIO@domain.com>

DSS   pub    2048/1024  [-----]   0xFAEBD5FC  Philip R. Zimmermann <prz@mcafee.com>
```

Example of --key-list --with-userids option

The following example shows the additional user IDs listed under the primary user ID for each key. Each of the additional user IDs is represented with "`uid`" in the `Type` column.

```
Alg   Typ   Size       Flags     Key ID      User ID
      e

RSA   pai   1024       [-----]   0x7D75EB0F  Albert Reilly <arielly@domain.com>
      r

      uid              [-----]               Albert Reilly <albert@mcafee.com>

      uid              [-----]               Albert Reilly <bigal@mcafee.com>

DSS   pub   2048/1024  [-----]   0xF5ED0CB   Corporate Key <CIO@domain.com>

DSS   pub   2048/1024  [-----]   0xFAEBD5FC  Philip R. Zimmermann <prz@mcafee.com>

      uid              [-----]               Philip R. Zimmermann <prz@acm.org>
```

Example of --key-list --with-sigs option

The following example shows the signatures on each key listed below the user ID. The signatures are represented by "`sig`" in the `Type` column.

```
Alg    Type  Size/Typ   Flags     Key ID      User ID
             e

RSA    pair  1024       [-----]   0x7D75EB0F  Albert Reilly <arielly@domain.com>

       sig   RSA        [-----]   0x7D75EB0F  Albert Reilly <arielly@domain.com>

       sig   DSS        [-----]   OxAB123CD4  Unknown Signer

       uid              [-----]               Albert Reilly <albert@mcafee.com>

       uid              [-----]               Albert Reilly <bigal@mcafee.com>

DSS    pub   2048/102   [-----]   0xF5ED0CB   Corporate Key <CIO@domain.com>
             4

DSS    pub   2048/102   [-----]   0xFAEBD5FC  Philip R. Zimmermann <prz@mcafee.com>
             4
```

```
sig   DSS        [-----]   0xFAEBD5FC  Philip R. Zimmermann <prz@mcafee.com>
uid              [-----]              Philip R. Zimmermann <prz@acm.org>
```

Understanding the key list display

When you list the keys on a keyring, the following columns display specific information about each key: Alg, Type, Size, Flags, Key ID, and User ID. The information displayed in each of column is described in greater detail below.

Algorithm (Alg)

Specifies the algorithm used in the creation of the key.

- `DSS` indicates that the key is a DH/DSS key.
- `RSA` indicates that the key is either a new RSA key or an RSA Legacy key.

Type

Specifies the type of key:

- `pair` indicates that you have both the public and private keys.
- `pub` indicates that the key is a public key on your key ring.
- `sig` indicates a signature on a key.
- `uid` indicates an additional user ID associated with the key.
- `pid` indicates a photo ID on the key.

Size

Specifies the key size, the number of bits used to construct the key.

- A DH/DSS key displays 2 numbers; the first number represents the number of bits that make up the encryption key, and the second number represents the number of bits that make up the signing key. If there is no subkey, then the number displayed is the size of the signing key.
- A new RSA key (RSA4) also shows 2 numbers, one for the encryption key and one for the signing key, but they are the same number. If there is no subkey, then the number displayed is the size of the signing key.
- An RSA Legacy key has only one key for both encryption and signing.

Flags

There are 5 fields within the `Flags` column. Each of these fields represent specific key properties.

- The symbols in the first field represent the key's validity:

- a dash (–) indicates an invalid key.
- `v` indicates a marginally valid key.
- `V` indicates a valid key.

- The symbols in the second field represent the key's trust setting:
  - a dash (–) indicates an untrusted key
  - `t` indicates a marginally trusted key
  - `T` indicates a trusted key
  - `I` indicates that you implicitly trust this key.
- The symbols in the third field represent the status of the key—if the key is currently disabled or revoked:
  - a dash (–) indicates that the key is not disabled or revoked
  - `D` indicates that the key is currently disabled
  - `R` indicates that the key has been revoked
- The symbols in the fourth field tell you if key is expired:
  - a dash (–) indicates that the key is not expired
  - `E` indicates that the key is expired
- The symbols in the fifth field tell you if there's an Additional Decryption Key (ADK) present:
  - a dash (–) indicates that the key does not have an ADK
  - `A` indicates that there is an ADK present on the key

Key ID

Specifies the key ID, a legible code that uniquely identifies a key pair. Two key pairs may have the same user ID, but they will have different Key IDs.

User ID

Specifies the primary user ID, a text phrase that identifies a key pair. For example, one common format for a user ID is the owner's name and email address. The user ID helps users (both the owner and colleagues) identify the owner of the key pair.

D Exit and Error Codes

The tables in this appendix identify E-Business Server's exit and error codes.

General errors

Error	Description
0	Exit OK, no error
1	invalid file
2	file not found
3	unknown file
4	batchmode error
5	bad argument
6	process interrupted
7	out of memory error
8	environment error

Keyring errors

Error	Description
10	key generation error
11	non-existing key error
12	keyring add error
13	keyring extract error
14	keyring edit error
15	keyring view error
16	keyring removal error
17	keyring check error
18	key signature error or key signature revoke error
19	key signature removal error

Encode errors

Error	Description
20	signature error
21	public key encryption error
22	encryption error
23	compression error

Decode errors

Error	Description
30	signature check error
31	public key decryption error
32	decryption error
33	decompression error
34	keyring locked error

Split key errors

Error	Description
40	key splitting error
41	key joining error
42	sending share error

File errors

Error	Description
100	file wiping error
101	file parsing error

Smart card errors

Error	Description
200	smart card error

Group errors

Error	Description
300	group error
301	add group error
302	remove group error
303	view group error

Key reconstruction errors

Error	Description
400	key reconstruction error
401	generating key reconstruction data error

Key errors

Error	Description
500	extract photo ID from key error

Key server errors

Error	Description
600	delete key from key server error
601	disable key on key server error
602	search key server error
603	send keys to key server error
604	fetch keys from key server errors

Key update errors

Error	Description
700	update key error

E Supported Certificate Attributes

This appendix identifies the certificate attributes supported by E-Business Server.

General X.509 certificate attributes

- CertificateExtension
- CommonName (CN) **
- Country (C)
- Description
- DestinationIndicator
- DirectoryManagementDomain
- DNSName
- DomainComponent (DC)
- EmaiL (E)*
- GivenName
- HouseIdentifier
- Initials
- IPAddress
- ISDN
- Locality (L)*
- Name
- OrganizationalUnitName (OU)*
- OrganizationName (O)*
- PhysicalDeliveryOfficeName

** Attributes that E-Business Server is able to read and display from X.509 certificates issued by E-Business Server.

- POBOX
- PostalCode
- RFC822Name
- State (ST) **
- Street*
- SurName (SN)
- TelephoneNumber
- Title
- UnstructuredAddress
- UnstructuredName
- X121Address

Verisign-specific certificate attributes

- AdditionalField4
- AdditionalField5
- AdditionalField6
- Authenticate
- CertType
- Challenge
- EmployeeID
- MailFirstName
- MailLastName
- MailMiddleName
- MailStop

** Attributes that E-Business Server is able to read and display from X.509 certificates issued by E-Business Server.

F Compatibility with Previous Releases

For compatibility with earlier versions of E-Business Server, please refer to the notes outlined in this appendix.

Legacy compatibility

E-Business Server 8.5 is mostly compatible with the legacy options and configuration values used in previous versions. However, some on-screen messages, error strings and prompts have changed.

By setting the configuration value `CMDLINE-FORMAT` (or the environment variable `PGP_CMDLINE_FORMAT`) to `legacy`, E-Business Server translates legacy options into equivalent new options, and recognizes most previous configuration file options. (For more information on setting the compatibility mode, see *CMDLINE-FORMAT* on page 119.)

We advise all customers to carefully test existing scripts when running in legacy mode. Also, new scripts should be written using the new options since legacy options may not be supported in future releases.

If `INFO` is set to `Verbose` in the E-Business Server configuration file (same as `VERBOSE=2` in legacy mode), then E-Business Server displays the long-option equivalent for all legacy commands. This may assist you in porting legacy command line options to the new long-options. For more information on the `INFO` parameter, see *INFO* on page 127.

Using +OPTIONS on the command line

Previous versions of E-Business Server allowed configuration file options to be specified on the command line by preceding them with a plus (+) sign. This is supported for compatibility purposes only. We recommend using `--options` instead, since `+OPTIONS` will be removed in a future version of the product.

Upgrading from a previous release

E-Business Server 8.5 cannot reside on a machine that contains a previous version of other McAfee Security software. The older product(s) need to be removed from the system.

G Biometric Word Lists

The two-syllable word list

Two Syllable Word List				
aardvark	absurd	accrue	acme	adrift
adult	afflict	ahead	aimless	Algol
allow	alone	ammo	ancient	apple
artist	assume	Athens	atlas	Aztec
baboon	backfield	backward	banjo	beaming
bedlamp	beehive	beeswax	befriend	Belfast
berserk	billiard	bison	blackjack	blockade
blowtorch	bluebird	bombast	bookshelf	brackish
breadline	breakup	brickyard	briefcase	Burbank
button	buzzard	cement	chairlift	chatter
checkup	chisel	choking	chopper	Christmas
clamshell	classic	classroom	cleanup	clockwork
cobra	commence	concert	cowbell	crackdown
cranky	crowfoot	crucial	crumpled	crusade
cubic	dashboard	deadbolt	deckhand	dogsled
dragnet	drainage	dreadful	drifter	dropper
drumbeat	drunken	Dupont	dwelling	eating
edict	egghead	eightball	endorse	endow
enlist	erase	escape	exceed	eyeglass
eyetooth	facial	fallout	flagpole	flatfoot
flytrap	fracture	framework	freedom	frighten
gazelle	Geiger	glitter	glucose	goggles
goldfish	gremlin	guidance	hamlet	highchair
hockey	indoors	indulge	inverse	involve
island	jawbone	keyboard	kickoff	kiwi
klaxon	locale	lockup	merit	minnow
miser	Mohawk	mural	music	necklace
Neptune	newborn	nightbird	Oakland	obtuse
offload	optic	orca	payday	peachy
pheasant	physique	playhouse	Pluto	preclude

Two Syllable Word List				
prefer	preshrunk	printer	prowler	pupil
puppy	python	quadrant	quiver	quota
ragtime	ratchet	rebirth	reform	regain
reindeer	rematch	repay	retouch	revenge
reward	rhythm	ribcage	ringbolt	robust
rocker	ruffled	sailboat	sawdust	scallion
scenic	scorecard	Scotland	seabird	select
sentence	shadow	shamrock	showgirl	skullcap
skydive	slingshot	slowdown	snapline	snapshot
snowcap	snowslide	solo	southward	soybean
spaniel	spearhead	spellbind	spheroid	spigot
spindle	spyglass	stagehand	stagnate	stairway
standard	stapler	steamship	sterling	stockman
stopwatch	stormy	sugar	surmount	suspense
sweatband	swelter	tactics	talon	tapeworm
tempest	tiger	tissue	tonic	topmost
tracker	transit	trauma	treadmill	Trojan
trouble	tumor	tunnel	tycoon	uncut
unearth	unwind	uproot	upset	upshot
vapor	village	virus	Vulcan	waffle
wallet	watchword	wayside	willow	woodlark
Zulu				

The three-syllable word list

Three Syllable Word List				
adroitness	adviser	aftermath	aggregate	alkali
almighty	amulet	amusement	antenna	applicant
Apollo	armistice	article	asteroid	Atlantic
atmosphere	autopsy	Babylon	backwater	barbecue
belowground	bifocals	bodyguard	bookseller	borderline
bottomless	Bradbury	bravado	Brazilian	breakaway
Burlington	businessman	butterfat	Camelot	candidate
cannonball	Capricorn	caravan	caretaker	celebrate
cellulose	certify	chambermaid	Cherokee	Chicago
clergyman	coherence	combustion	commando	company
component	concurrent	confidence	conformist	congregate
consensus	consulting	corporate	corrosion	councilman
crossover	crucifix	cumbersome	customer	Dakota
decadence	December	decimal	designing	detector
detergent	determine	dictator	dinosaur	direction
disable	disbelief	disruptive	distortion	document
embezzle	enchanting	enrollment	enterprise	equation
equipment	escapade	Eskimo	everyday	examine
existence	exodus	fascinate	filament	finicky
forever	fortitude	frequency	gadgetry	Galveston
getaway	glossary	gossamer	graduate	gravity
guitarist	hamburger	Hamilton	handiwork	hazardous
headwaters	hemisphere	hesitate	hideaway	holiness
hurricane	hydraulic	impartial	impetus	inception
indigo	inertia	infancy	inferno	informant
insincere	insurgent	integrate	intention	inventive
Istanbul	Jamaica	Jupiter	leprosy	letterhead
liberty	maritime	matchmaker	maverick	Medusa
megaton	microscope	microwave	midsummer	millionaire
miracle	misnomer	molasses	molecule	Montana
monument	mosquito	narrative	nebula	newsletter
Norwegian	October	Ohio	onlooker	opulent
Orlando	outfielder	Pacific	pandemic	Pandora
paperweight	paragon	paragraph	paramount	passenger
pedigree	Pegasus	penetrate	perceptive	performance
pharmacy	phonetic	photograph	pioneer	pocketful
politeness	positive	potato	processor	provincial
proximate	puberty	publisher	pyramid	quantity
racketeer	rebellion	recipe	recover	repellent
replica	reproduce	resistor	responsive	retraction
retrieval	retrospect	revenue	revival	revolver

Three Syllable Word List				
sandalwood	sardonic	Saturday	savagery	scavenger
sensation	sociable	souvenir	specialist	speculate
stethoscope	stupendous	supportive	surrender	suspicious
sympathy	tambourine	telephone	therapist	tobacco
tolerance	tomorrow	torpedo	tradition	travesty
trombonist	truncated	typewriter	ultimate	undaunted
underfoot	unicorn	unify	universe	unravel
upcoming	vacancy	vagabond	vertigo	Virginia
visitor	vocalist	voyager	warranty	Waterloo
whimsical	Wichita	Wilmington	Wyoming	yesteryear
Yucatan				

Index

S

T