

e-speak

System Administration Guide

*Release A.03.14.00
August 2001*

COPYRIGHT NOTICE

© 2001 HEWLETT-PACKARD COMPANY

To anyone who acknowledges that this document is provided "AS IS" WITH NO EXPRESS OR IMPLIED WARRANTY: permission to copy, modify, and distribute this document for any purpose is hereby granted without fee, provided that the above copyright notice and this notice appear in all copies, and that the name of Hewlett-Packard Company not be used in advertising or publicity pertaining to distribution of this document without specific, written prior permission. Hewlett-Packard Company makes no representations about the suitability of this document for any purpose.

Windows and Windows NT are registered trademarks of Microsoft Corporation. Java and all Java-based marks are trademarks or registered trademarks of Sun Microsystems, Inc. Linux is a trademark of Linus Torvalds. All other trademarks are the properties of their respective owners.

Contents

Chapter 1	Getting Started	1
	Interpreting Examples	1
	Using the E-speak Library	2
Chapter 2	Installing on Windows NT	3
	E-speak Installation Requirements	3
	External Modules	4
	Memory, Cache, and Quota Usage	5
	Downloading and Installing the E-speak Software	7
Chapter 3	Installing on HP-UX™	9
	Creating an E-speak Account	9
	E-speak Installation Requirements	10
	External Modules	11
	Downloading and Installing E-speak	12
	Downloading E-speak	12
	Installing E-speak	13
Chapter 4	Installing on Linux™	15
	Creating an E-speak Account	15
	E-speak Pre-installation Checklist	16

	External Modules	17
	Downloading and Installing the E-speak Software	18
	Running the Setup Program	19
Chapter 5	Standard Configuration	21
	Basic Concepts	22
	Working with Properties	22
	Understanding Property File Syntax	23
	Tracing	24
	Setting the Tracing Level	24
	Traversing a Fire Wall	25
	Specifying the Firewall Traversal Method	25
	Configuring a Web Proxy	26
	Configuring a SOCKS V4 Proxy	28
	Configuring the Advertising Service	30
	E-services Village Proxy Settings	32
	Configuring Logging	33
Chapter 6	E-speak Desktop	39
	Desktop Interface	40
	Installing a Plug-in	41
	Desktop Management	42
	The Desktop Configuration File	42
	Starting the Desktop	44
	Accessing the System Pop-up Menu	45

Connecting to an e-speak-Enabled Host	45
Starting a Process	46
Configuring Multiple Processes	47
Managing Runtime Configuration	47
Counter Controls	49
Stopping a Process	49
Discovering Manageable e-speak Components and Services Running in a Managed Process	49
Managing e-speak Components and Services	50
Create New Cores	50
Shutting Down the Core	50
The Core Browser	51
Provided Plug-ins	51
Process Manager Plug-in	51
Process Lookup	52
Service Lookup	52
Persistent Processes	52
Host Groups	53
Connection Groups	53
The Element Manager Plug-in	56
The Configuration Manager Plug-in	56
The Service-Manager Plug-in	58
The Core Manager Plug-in	58
Desktop Walkthrough	59
Desktop Reference	70
Service Control Window Menu and Options	73

Chapter 7	Running Standard Services	77
	Running the Advertising Service	78
	Working in Offline Mode	79
	Installing a Directory Server	79
	Download the LDAP JDK	80
	Starting the Advertising Service	81
	How the Advertising Service reads Configurable Properties	81
	Defining Properties in the Command Line	82
	Guidelines for Starting the Advertising Services	82
	Examples	82
	Process Manager	84
	Starting the Process Manager	85
Chapter 8	Expanding Functionality	87
	Gateway Usage	87
	Implementation Restrictions	87
	Features	88
	Installation	88
	Conventions	89
	Changes in Webmacro.properties	89
	Changes in WebAccess.xml	89
	Setting the Shell Environment for Unix and Linux™ Platforms	90
	Installing Apache Web Server on Windows NT®	90
	Installing Apache JServ on Windows NT®	91

Changes in jserv.properties	92
Changes in zone.properties	93
Debug Logging Changes When Using Apache/Jserv	93
Installing Apache Web Server and JServ on HP-UX™ and Linux™	94
Testing Gateway Usage Configuration	94
Testing Gateway Usage Configuration When Using Apache/Jserv	94
Troubleshooting Gateway Usage with Apache/Jserv	96
Installing Tomcat	97
Installing Tomcat on Windows NT®	98
Installing Tomcat on HP-UX™ and Linux™	101
Testing Gateway Usage Configuration When Using Tomcat	106
Troubleshooting Gateway Usage When Using Tomcat	107
Firewall Gateway	109
Orientation	109
Infrastructure Requirements	111
Firewall Gateway Configuration	111
E-speak-Specific Configuration File	111
Gateway-Specific Configuration File	111
Configuration Mode	113
Gateway Inside and Outside Address	113
External and Internal Listeners	114
SocketPool	114
Logger	114
Policy configuration	114

	Internal Engine Configuration	115
	Gateway parameters for connection:	117
	How an External Engine Connects to the Firewall Gateway	117
	Running the Firewall Gateway	120
	Reconfiguring the Firewall Gateway	121
	Limitations	121
Chapter 9	espeak Utility	123
	Displaying Help	123
	Help Text	124
Chapter 10	Troubleshooting	129
	Installation and Configuration	130
	J-ESI	131
	J-Kernel	132
	Client API	133
	Sample Application	135
	WebAccess	136
	Security	138
	Working with the Desktop	140
Appendix A	Advanced E-speak Configuration Parameters .	141
	Tracing	141
	Tracing Security Components	141
	Security	141

Security	143
Security Properties	143
Sample espeak.cfg file	144
Determining Whether Stub Class Files are Downloaded	145
Specifying the Security Setup Duration	145
Setting Threads for Polling	149
Configuring the Engine to the Firewall Gateway .	152
Preventing Denial of Service Attacks	154
Configuring the Advertising Service	155
Multicast Settings	157
LDAP Settings	159
E-services Village (ESV) Proxy Settings	162
Configuring the Core to Gateway Connection ...	163
Configuring Repository Variables	165
Configuring JDBC-related Parameters	169
Connecting to the Database	170
Configuring Logging	177
Appendix B Configuring Security	183
Private Secure Environment (PSE) Overview	184
Private Secure Environment	184
Delegation and Trust Assumptions	185
PSE Manager	185
Starting the PSE Manager	186
Creating a PSE	186

Generating Key Pairs	187
Certificates	188
Managing Certificates Using PSE Manager	188
Creating Certificates	189
Trust Assumptions	191
Using PSE Key Labels as Symbolic Principals	192
Editing and Browsing Certificates	192
Managing Keyrings	193
Creating a Keyring	193
Passing a Keyring to a Remote Client	194
Using PSE Manager as an Attribute Certificate Issuer	196
Sample Certificates Generated by Client A and Client B	197
Security Examples	199
Bootstrap Process for Testing	202
Simplified espeak.cfg file	203
Appendix C Uninstalling E-speak	205
Windows NT	205
Linux	206
HP-UX	206

Chapter 1 Getting Started

This manual's purpose is to guide you through installing, configuring, and monitoring e-speak. You need not be familiar with e-speak to effectively use this manual; however, must have a basic knowledge of your operating system.

TIP: Although it isn't required, you may find it useful to go through the *e-speak Tutorial* before using this manual.

Before you begin the e-speak installation, please read this short introduction chapter. It will ultimately *save* you time during the installation since it discusses how to:

- Interpret the examples in this manual so that you can enter commands quickly and accurately
- Use the rest of the e-speak documentation set to learn more about the product

Interpreting Examples

One of the most common errors users make is entering a command line verbatim from the manual, when the line shown is only an example. To distinguish commands that you should enter exactly as shown from those you need to modify to fit your environment, the manual specifies variable information in italics.

For example, you can enter this command verbatim:

```
./setup.sh
```

But in this one, you must substitute your own installation directory path:

```
cd <installation_directory>/bin
```

[Table 1](#) lists other document conventions.

Table 1 Document conventions

This information ...	Is shown in ...
New and emphasized terms	bold type
Code samples, file and directory names, library names, package names, and methods	<code>monospace type</code>
Universal resource locators (URLs)	<code>monospace bold type</code>

Using the E-speak Library

In addition to this *System Administration Guide*, the e-speak library includes:

- *e-speak Architectural Specification*:
 - Defines the abstractions presented by the system and the components that implement those abstractions
 - Demonstrates how the components interact to create useful services
- *e-speak Programmers Guide* defines the interface for e-speak programmers and system developers building e-speak-enabled applications
- *e-speak Contributed Software* describes:
 - VFS (Virtual File System)
- *e-speak Tutorial* includes a walk-through and programming examples of the basic concepts and functions of espeak

For information or support on e-speak and e-speak-related issues, contact the e-speak support team at:

- <http://www.e-speak.hp.com>
- www.espeak.net

Chapter 2 Installing on Windows NT

This chapter discusses:

- E-speak installation requirements
- Downloading and installing e-speak in a Windows NT environment

NOTE: If you are upgrading to a new version of e-speak, you must uninstall your current version first. For more information, see [“Downloading and Installing the E-speak Software”](#).

E-speak Installation Requirements

During the e-speak installation, you will be prompted for the directory path of your Java™ Developer’s Kit (JDK). Locate your JDK now and made a note of its directory path. Before you begin the installation, ensure that your system meets the hardware and software requirements listed in [Table 2](#) and [Table 3](#).

Table 2 Hardware requirements

Hardware	Value
RAM	256 MB
Freespace during installation	128 MB minimum

Table 3 Software requirements

Software	Version/Value
Operating system	4.0/SP 4 or later
Sun™ Java™ Developer's Kit (JDK)	1.2.2 or higher

External Modules

To run some e-speak components, you may need to install modules or components not included in the e-speak release. The following table lists those dependencies. If you decide to use a certain e-speak component, make sure you install the corresponding external module on your machine before you install e-speak.

Table 4 Advertising service

Use	Lightweight Directory Access Protocol (LDAP) server
Version	3.10
Purpose	<p>This component allows you to form service communities so that you can share services outside of your local area network (LAN).</p> <p>LDAP is not required for services that are:</p> <ul style="list-style-type: none"> • Only accessed within a LAN • Advertised globally through Hewlett Packard's LDAP server at e-services Village.
Download Site	www.innosoft.com/ldapworld

Table 5 E-speak core in persistent mode

Use	Oracle backend database (accessible using JDBC) E-speak can run with an in-memory repository. In this case, it does not need an external database.
Version	8i
Purpose	This component prevents services from having to register themselves each time the Core disconnects and allows vocabularies and accounts to exist beyond Core disconnects.
Download Site	www.oracle.com

Table 6 Gateway usage

Use	Tomcat
Version	3.1
Purpose	This component allows you to make web applications (such as XML/SOAP-based services running on a web server) available as e-services.
Download Site	http://jakarta.apache.org/tomcat

Memory, Cache, and Quota Usage

Table 7 summarizes the typical interaction between an e-speak client/service and states the memory, cache, quota used.

NOTE: Memory usage does not include memory used by the Java Virtual Machine (JVM) environment.

Table 7 Memory, Cache, and Quota Usage

Description	Memory (KB)	Cache (KB)		Quota (KB)	
	Total	Total	Alone	Total	Alone
Core startup	1352	160	160		
Server					

Table 7 Memory, Cache, and Quota Usage

Description	Memory (KB)	Cache (KB)		Quota (KB)	
	Total	Total	Alone	Total	Alone
Initialization	1511	160	0	0	0
Making a connection	1623	170	10	9	9
Registering a vocabulary Note: The size of the vocabulary depends on the number of attributes it contains.	1815	176	6	14	5
Registering 100 services (using the default vocabulary)	6858	not available	200	2052	
Registering 100 services (using a customized vocabulary with two attributes)	4083	not available	200	2246	
Client					
Initialization	1862	183	0	0	0
Making a connection	1918	193	10	9	9
Finding a vocabulary	1963	192	0	9	0
Creating a ServiceFinder	1964	192	0	9	0
Finding two services	1970	192	0	9	0
Creating a folder Note: The initial cache size is higher because creating the first folder on a client also creates a home folder. Creating a single folder requires approximately 2.5 KB of cache and 2.0 KB of quota.	1991	199	7	15	6

Table 7 Memory, Cache, and Quota Usage

Description	Memory (KB)	Cache (KB)		Quota (KB)	
	Total	Total	Alone	Total	Alone
Creating 100 folders	2342	429	231	234	219
Creating 1000 folders	5864	2695	1	2386	1
Adding two services to a folder	5864	2695	1	2386	1
Adding 100 services to a folder	5872	2743	48	2434	48
Adding 1000 services to a folder	6034	3223	481	2915	481

Downloading and Installing the E-speak Software

NOTE: You must have administrator privileges to complete this section.

The e-speak software is packaged in a self-extracting InstallShield executable file:

```
espeak_<version>.exe
```

where *<version>* is the e-speak release number. Download this file from <http://www.e-speak.hp.com/developers>.

To start InstallShield, run `espeak_<version>.exe`. Follow the InstallShield prompts during the installation. When you see this prompt:

```
Enter the directory where java(tm) is installed
```

enter the JRE Java (1.2.x) install directory. You can either browse for the directory or enter the path directly.

By default, the e-speak installation directory is `c:\program files\e-speak`. If necessary, change this to the appropriate directory. Ensure that your system has at least 75 MB of free hard disk space.

After InstallShield creates the necessary directories and copies the files into them, it installs basic security certificates.

TIP: As part of the installation process, the e-speak Process Manager is created as a service. Because the Process Manager starts automatically when the system starts and can consume significant resources, you may want to stop this service when you are not running e-speak.

Chapter 3 Installing on HP-UX™

This chapter discusses:

- Creating an e-speak account
- E-speak installation requirements
- Downloading and installing e-speak

NOTE: If you are upgrading to a new version of e-speak, you must uninstall your current version first. For more information, see [“Downloading and Installing E-speak”](#).

Creating an E-speak Account

The e-speak Process Manager starts other processes. These processes have the same permission as the Process Manager; because of this, you must start the Process Manager service as an e-speak user account rather than a root account.

NOTE: An e-speak user account requires a `<home directory>`; however, it does not require a specified SHELL.

To create an e-speak account, use the System Administration Manager (SAM). For information on account options, refer to the SAM documentation.

E-speak Installation Requirements

TIP: Before you continue with the installation, be sure to create an e-speak account.

Ensure that your system meets the requirements outlined in [Table 8](#) and [Table 9](#).

Table 8 Hardware requirements

Hardware	Value
RAM	256 MB
Freespace during installation in /opt	128 MB minimum

Table 9 Software requirements

Software	Version/Value
Operating system	11.00
Operating system kernel parameters	maxfiles=256 (default=60) maxuprc=1024 (default=75) maxusers=1024 (default=32) NPTY=512 (default=60) max_thread_proc=2048 (default=64)
HP Java™ Developer's Kit (JDK)	1.2.2.05 (or later)
Perl	5.003 or later

NOTE: For a list of required HP-UX™ patches, downloads, JAVA™ downloads, and support information, see <http://www.unix.hp.com>.

External Modules

To run certain e-speak components, you may need to install modules or components not included in the e-speak release. If you want to use the advertising service, the e-speak core in persistent mode, or a gateway, install the corresponding external module on your system before you install e-speak.

Table 10 Advertising service

Use	Lightweight Directory Access Protocol (LDAP) server
Version	3.10
Purpose	<p>This component allows you to form service communities so that you can share services outside of your local area network (LAN).</p> <p>LDAP is not required for services that are:</p> <ul style="list-style-type: none"> • Only accessed within a LAN • Advertised globally through Hewlett Packard's LDAP server at e-services Village.
Download Site	www.innosoft.com/ldapworld

Table 11 E-speak core in persistent mode

Use	<p>Oracle backend database (accessible using JDBC)</p> <p>E-speak can run with an in-memory repository. In this case, it does not need an external database.</p>
Version	8i
Purpose	This component prevents services from having to register themselves each time the Core disconnects and allows vocabularies and accounts to exist beyond Core disconnects.
Download Site	www.oracle.com

Table 12 Gateway usage

Use	Tomcat
Version	3.1
Purpose	This component allows you to make web applications (such as XML/SOAP-based services running on a web server) available as e-services.
Download Site	http://jakarta.apache.org/tomcat

Downloading and Installing E-speak

NOTE: To complete this section, you must have root privileges and access to the `swinstall(1M)` utility.

The e-speak software is packaged in a Software Distribution depot:

```
espeak_<version>.depot
```

where `<version>` is the e-speak release number. Download this file from <http://www.e-speak.hp.com/developers>.

Downloading E-speak

To download and install the e-speak software:

1 Download `espeak_<version>.depot`.

2 Log in as `root`.

3 Move the installation file to a temporary directory, such as `/tmp`.

TIP: Ensure that the directory has enough disk space to hold both the compressed and uncompressed versions of the file.

4 Navigate to the temporary directory. For example:

```
cd /tmp
```

5 Use the `swinstall` utility to install the software. The default installation directory is `/opt/e-speak`.

```
/usr/sbin/swinstall -s /tmp/espeak_<version>
```

NOTE: You can add the `*` option to the command line to perform the installation without displaying a user interface:

```
/usr/sbin/swinstall -s /tmp/espeak_<version>\*
```

The e-speak files are located in the `/opt/` directory; `/opt/e-speak` is the e-speak installation directory.

Installing E-speak

NOTE: Ensure that you are still logged in as `root` before completing this section.

The e-speak setup utility automates part of the installation by accurately defining your e-speak environment variables. The setup utility:

- Locates the e-speak software
- Prompts for the location of your JDK (unless `/opt/java12` exists)
- Automatically defines environment variables for your JDK, Java™ Runtime Environment (JRE) and, if present, your java™ compiler (JAVAC)
- Stores the variables in an `espeak.env` file in the `bin` directory
- Installs basic security certificates

To run the setup utility:

- 1 Navigate to `<installation_directory>/bin`. The `bin` directory in the installation path contains the e-speak setup program `setup.sh`.

```
cd <installation_directory>/bin
```

NOTE: Ensure that you are working in the `bin` directory; otherwise, step 2 will generate errors.

- 2 Type `./setup.sh`.
- 3 Press **Enter** to start the utility.

NOTE: If you have not set up an e-speak account, you will receive an error message and the setup utility will exit. Create an account before re-running `setup.sh`.

- 4 Type `y`.
- 5 Press **Enter**.

- 6** Enter the JDK installation location if the setup utility does not automatically find it. For example:

```
Enter the directory where the Java(tm) Software Development Kit is
  installed
```

```
For example: /opt/java1.2
```

NOTE: Because this prompt has no default value, wording will vary.

- 7** Enter the path.
- 8** Press **Enter**. The setup program creates:

- Security certificates
- The `espeak.env` file in the `/bin` directory and displays:

```
Environment settings for e-speak is complete
```

Chapter 4 Installing on Linux™

This chapter discusses:

- Creating an e-speak account
- The e-speak pre-installation checklist
- Downloading and installing the e-speak software
- Running the Setup program

Creating an E-speak Account

The e-speak Process Manager starts other processes. These processes have the same permission as the Process Manager; because of this, you must start the Process Manager service as an e-speak user account rather than a root account.

NOTE: An e-speak user account requires a <home directory>; however, it does not require a specified SHELL.

To create an e-speak account:

- 1** Log in as `root`.
- 2** Type `useradd espeak`.

NOTE: To establish an e-speak account, `useradd` must be included in the path.

- 3** Press **Enter**.

E-speak Pre-installation Checklist

TIP: Before you continue with the installation, be sure to create an e-speak account.

When you run the e-speak setup program, you must enter the directory path of the Java™ Developer's Kit (JDK). You must have JDK version 1.2.2 or higher installed.

Locate your JDK now and have that information ready. Next, ensure that your system meets the requirements outlined in [Table 13](#) and [Table 14](#).

Table 13 Hardware requirements

Hardware	Requirement
RAM	256 MB
Freespace during installation in <code>/usr/local/e-speak</code>	128 MB minimum

Table 14 Software requirements

Software	Requirement	Download Site
Red Hat Linux™	6.2	www.redhat.com
Blackdown's Java™ Developer's Kit (JDK)	1.2.2	www.blackdown.org
Perl	5.003	www.perl.com

NOTE: The software version numbers listed in the table have been tested with e-speak. You should be able to use later versions of the software as well, with one exception: do not use JDK version 1.3.

External Modules

To run certain e-speak components, you may need to install modules or components not included in the e-speak release. If you want to use the advertising service, the e-speak core in persistent mode, or a gateway, install the corresponding external module on your system before you install e-speak.

Table 15 Advertising service

Use	Lightweight Directory Access Protocol (LDAP) server
Version	3.10
Purpose	<p>This component allows you to form service communities so that you can share services outside of your local area network (LAN).</p> <p>LDAP is not required for services that are:</p> <ul style="list-style-type: none"> • Only accessed within a LAN • Advertised globally through Hewlett Packard's LDAP server at e-services Village.
Download Site	www.innosoft.com/ldapworld

Table 16 E-speak core in persistent mode

Use	<p>Oracle backend database (accessible using JDBC)</p> <p>E-speak can run with an in-memory repository. In this case, it does not need an external database.</p>
Version	8i
Purpose	This component prevents services from having to register themselves each time the Core disconnects and allows vocabularies and accounts to exist beyond Core disconnects.
Download Site	www.oracle.com

Table 17 Gateway usage

Use	Tomcat
Version	3.1
Purpose	This component allows you to make web applications (such as XML/SOAP-based services running on a web server) available as e-services.
Download Site	http://jakarta.apache.org/tomcat

Downloading and Installing the E-speak Software

NOTE: To complete this section, you must have root privileges.

To install the e-speak software:

- 1 Log in as `root`.
- 2 Type `$rpm --query --all | grep E-speak` to display the version of e-speak installed on your system, if one exists. For example:
`E-speak-A.03.13.00-1`
- 3 Type `$rpm --erase <e-speak_version_number>` to uninstall the current version of e-speak, if one exists. For example:
`$rpm --erase E-speak-A.03.13.00-1`
- 4 Type `$rpm --install /tmp/espeak_<version>.rpm` to install e-speak. This example assumes that the rpm is in `/tmp`.
- 5 E-speak is located in the `/usr/local` directory. The e-speak installation directory is `/usr/local/e-speak`.

Running the Setup Program

NOTE: Before you start this process, ensure that:

- An e-speak account exists. If an account does not exist, `setup.sh` aborts and displays:

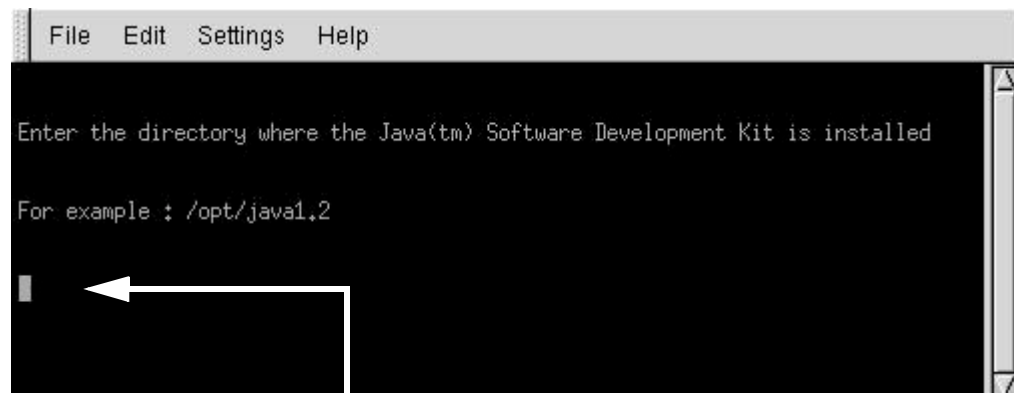
```
Please make sure that e-speak account is existing.
```

- You are logged in as `root`.

Next, run the e-speak setup program. This command-line program automates part of the installation by accurately defining your e-speak environment variables. The setup program:

- Locates the e-speak software
- Prompts for the location of the JDK
- Automatically defines environment variables for your JDK, Java™ Runtime Environment (JRE) and, if present, your `java™` compiler (JAVAC)
- Stores the variables in an `espeak.env` file in the `bin` directory
- Installs basic security certificates

TIP: The Setup interface does not use standard UNIX prompts. Type commands on the blank line below the prompt, then press **Enter**.



Type commands here,
then press **Enter**.

Figure 1 Setup interface

To run the setup utility:

- 1 Navigate to `<installation_directory>/bin`. The `bin` directory in the installation path contains the e-speak setup program `setup.sh`.

```
cd <installation_directory>/bin
```

NOTE: Ensure that you are working in the `bin` directory; otherwise, step 2 will generate errors.

- 2 Type `./setup.sh`.
- 3 Press **Enter** to start the utility.

Setup locates the e-speak software, displays its location, and prompts you to verify that this is your installation. This example uses the default installation path `/opt/espeak`.

```
Found /opt/e-speak ... Is this your installation (Y/N)
```

NOTE: This is an example only. The wording of the prompt may vary.

- 4 Type `y`.
- 5 Press **Enter**.
- 6 Enter the JDK installation location if the setup utility does not automatically find it. For example:

```
Enter the directory where the Java(tm) Software Development Kit is
installed
```

```
For example: /opt/java1.2
```

NOTE: Because this prompt has no default value, wording will vary.

- 7 Enter the path.
- 8 Press **Enter**. The setup program creates:

- Security certificates
- The `espeak.env` file in the `/bin` directory and displays:

```
Environment settings for e-speak is complete
```

Chapter 5 Standard Configuration

The e-speak configuration file, `espeak.cfg`, controls e-speak's runtime behavior. This file is located in `<installation_directory>/config`.

The configuration file is text file consisting of a series of parameters and values. Every parameter is configured with a placeholder value; parameters that are not used are commented out. Each e-speak parameters belongs to one of three basic categories:

- Parameters that you **must** modify with your site-specific values
- Parameters that you **may** want to modify after you've been using e-speak for a while
- Parameters that you **should not** modify unless you are an expert in that particular area of technology and you fully understand the impact your modifications will have on other parameters and on the system as a whole

This chapter discusses only those parameters you must modify as part of the basic e-speak installation. Refer to [Appendix A, "Advanced E-speak Configuration Parameters"](#) for additional parameters. [Appendix B, "Configuring Security"](#) discusses security-specific configuration.

The documentation for each parameter appears in this format:

The text of the parameter as it appears in the configuration file.

Purpose	How this parameter is used and the functions it controls
Default	A description of the default value, if any, and why that particular value was selected
Options	The alternative values (or range of values) you can specify
Modify when	Guidelines on when and why to modify the value
Impact	The impact of changing this parameter, if any, on performance, security, and so on
See also	Cross-references, when appropriate, to more detailed information about the topic or technology

Basic Concepts

E-speak follows this hierarchy when searching for the configuration file:

- 1** The `/config` directory in `<e-speak home>` as defined by the property `espeak_home`
- 2** The directory specified by the property `net.espeak.util.config.path`
- 3** The current directory (from the system property `user.dir`)
- 4** The directory specified by the `user.home` system property as a system resource from the `CLASSPATH`

Working with Properties

You can specify Java system properties on the command line using this syntax:

```
-Dproperty=value
```

You can specify the file name to search for by entering:

```
net.espeak.util.config.file property
```

The file defined by the property `net.espeak.util.config.master` is always loaded on top of all other files, if specified. This property's default value is null.

The files are formatted in Java properties file format, with the additions of `@mode=<mode>`. If the mode is `override` (default), the values found in this file take precedence over all previous values. Once the `espeak.cfg` parser encounters a file with its mode set to `override`, it stops parsing files. If the mode is `merge`, the `espeak.cfg` files are combined. If two files specify values for the same property, only the values in the last file to be parsed are used.

You can specify the name of the configuration file to search for using the system property `net.espeak.util.config.file`, which includes the default value `espeak.cfg`. If you set the system property `net.espeak.util.config.master`, the specified file is loaded on top of all other files found.

Understanding Property File Syntax

A Java properties file contains property names and definitions and allows the following conventions:

- Separate the name from the definition using `=`
- Spaces before the property name and around the `=` are ignored
- The value of the property extends to the end of line and includes trailing spaces
- Break long property values across lines using `\`
- Use `!` and `#` to introduce end-of-line comments
- You may use `:` instead of `=`

Tracing

Setting the Tracing Level

ESTrace

Purpose	<p>Defines the tracing level, output device, filters, and formatting. The ESTrace property is defined as:</p> <pre>ESTrace={writer{,writer}}</pre> <pre>writer=ClassName(paramString)=specification</pre> <p>ClassName</p> <p>The fully qualified ClassName of the writer class</p> <p>paramString</p> <p>The unquoted string parameter that is passed to the writer implementation; for example, the file name</p> <p>specification</p> <p>The filter and format information is as follows:</p> <pre>specification=filters;format</pre> <pre>filters=filter{:filter}</pre> <pre>format=item{+item}</pre> <p>item=* None TimeStamp Invoker Function Severity</p> <pre>filter=function-severity</pre> <p>function must match the name of the function you specified when creating the ESDebug class.</p> <p>severity must be EMERG, ALERT, CRIT, ERROR, EXCEPTION, WARNING, NOTICE, ADMIN, INFO, TRACE, or DEBUG1-DEBUG9.</p>
Default	off

ESTrace

Options	<pre>default=net.espeak.util.ESPrintWriter()=*-EMERG:*- ALERT:*-CRIT:*-ERROR:*-WARNING:*-NOTICE:*- INFO;Severity+Invoker debugAll=net.espeak.util.ESPrintWriter()=*-*;ALL debugEmerg=net.espeak.util.ESPrintWriter()=*- EMERG;all</pre> <p>Note: You should only use this parameter for tracing.</p>
Modify when	Turn on tracing when you need a more detailed understanding of e-speak activity
Impact	Tracing generates additional text messages Note: A large number of extra messages may effect e-speak performance.
See also	"Configuring Logging"

Traversing a Fire Wall

E-speak clients can communicate through firewalls using one of two methods:

- Web proxy
- SOCKS server

Specifying the Firewall Traversal Method

net.espeak.infra.cci.messaging.proxy

Purpose	Specifies the method clients use to traverse fire walls
Default	N/A
Options	Socks4 http none

net.espeak.infra.cci.messaging.proxy

Modify when	Typically, do not modify this parameter once it has been set
See also	"Configuring a Web Proxy" "Configuring a SOCKS V4 Proxy"

Configuring a Web Proxy

To use a web proxy, you must configure two parameters. You may also need to configure two additional parameters depending on your environment. You **must** configure the:

- Web proxy name
- Web proxy port number

You may also configure:

- The web proxy user name and password
- Whether or not certain domain names require a proxy (cases in which one client may communicate with another directly)

net.espeak.infra.cci.messaging.httpProxyName

Purpose	<p>Specifies a web proxy name</p> <p>Note: This parameter does not take effect if the Client is connecting the the Core though localhost.</p>
Options	A valid web proxy name
Modify when	<p>You change web proxies</p> <p>For example, you may change web proxies if the proxy goes down or has slow performance.</p>
Impact	If you do not set this parameter, clients cannot communicate with each other.
See also	net.espeak.infra.cci.messaging.httpProxyPort

net.espeak.infra.cci.messaging.httpProxyPort

Purpose	<p>Specifies the port number on which the web proxy is running.</p> <p>You configure this parameter when you configure:</p> <pre>net.espeak.infra.cci.messaging.httpProxyName</pre> <p>Note: This parameter does not take effect if the Client is connecting the the Core though localhost.</p>
Default	90
Options	The port number on which the web proxy is running
Modify when	Modify this parameter if you change web proxies
See also	net.espeak.infra.cci.messaging.httpProxyName

net.espeak.infra.cci.messaging.user

net.espeak.infra.cci.messaging.passwd

Purpose	<p>Some web proxies support user name and password authentication. These two parameters specify a user name and password that the client must use before going through the web proxy.</p> <p>The user name and password parameters are only when the following parameter is assigned an http value:</p> <pre>net.espeak.infra.cci.messaging.proxy=http</pre> <p>If the <code>http</code> parameter is not assigned, the authentication parameter is not active. This parameter does not have an effect on system performance.</p>
Options	A client can also a traverse firewall using a SOCKS server
Modify when	You change web proxies. Make sure the proxy that you change to has the correct username and password.

net.espeak.infra.cci.messaging.noHttpProxy

Purpose	<p>Indicates one or more domain names for which a web proxy should not be used</p> <p>In these cases, one client can contact the other directly.</p> <p>This parameter supports multiple values and IP addresses. The syntax for defining two or more domain names for which a web proxy should not be used:</p> <p>*.xxx.com *.yyy.com *.zzz.com.</p> <p>Note: This parameter does not take effect if the Client is connecting the Core through localhost.</p>
Options	One or more valid host names and IP addresses
Modify when	Typically, do not change this parameter once it has been set.

Configuring a SOCKS V4 Proxy

Clients can also communicate with each other through fire walls through a SOCKS server. To use a SOCKS server, you **must** configure:

- SOCKS proxy name
- SOCKS proxy port number

Depending on your environment, you may also need to configure:

- SOCKS proxy user name and password
- Whether or not certain domain names require a SOCKS proxy (cases in which one client may communicate with another directly)

net.espeak.infra.cci.messaging.socksProxyName

Purpose	Specifies a SOCKS proxy name.
Options	<p>A valid SOCKS proxy name</p> <p>Note: The e-speak core only supports SOCKS V4.</p>

net.espeak.infra.cci.messaging.socksProxyName

Modify when	You change SOCKS proxies For example, if the proxy goes down or has slow performance.
See also	net.espeak.infra.cci.messaging.httpProxyPort

net.espeak.infra.cci.messaging.socksProxyPort

Purpose	Specifies the port number on which the SOCKS server is running You configure this parameter at the same time you configure <code>net.espeak.infra.cci.messaging.socksProxyName</code> .
Default	1080
Options	A valid port number
Modify when	You change SOCKS proxies.
See also	net.espeak.infra.cci.messaging.httpProxyName

net.espeak.infra.cci.messaging.noSocksProxy

Purpose	Specifies one or more domains for establishing a direct connection with other clients This parameter supports multiple entries, each one separated by a space. The wild card ('*') and IPv4 options are also supported. An example of specifying this parameter is: <code>noHttpProxy=* .hp.com localhost 10.20.* 10.20.10.</code>
Options	One or more valid domain names
Modify when	Typically, do not change this parameter once it has been set.

Configuring the Advertising Service

The Advertising Service is similar to the concept of communities as described in the *e-speak Programmers Guide*. For information about how to pass these properties to the Advertising Service, see [“Chapter 7, “Running Standard Services.”](#)

net.espeak.services.advService.group

Purpose	<p>Specifies the group the Advertising Service belongs to</p> <p>Advertising Services that have the same group setting and share the same backend form a group. The group associates services running on these cores together. Services advertised in any of these cores can be found by another in any core(s) in the same group.</p> <p>If you do not intend to advertise your services to e-services Village, you may want to select your own group name. Two different service providers can advertise services with exactly the same descriptions (attribute values). If the service provider wants to prevent collisions across the advertised services or wants to protect the access to the services advertised, the service provider can specify a unique group name for the Advertising Service.</p> <p>Note: This parameter is equivalent to the <code>-group</code> option in previous versions of the Advertising Service.</p>
Default	speaktome, which is the group name of e-services Village, the HP sponsored global service directory
Options	Any valid group name
See also	<i>e-speak Programmers Guide</i>

net.espeak.services.advService.backend.protocol

Purpose	Specifies the type of backend service directory to be used by the Advertising Service
Default	esvProxy (use SLP if ESV is not available)

net.espeak.services.advService.backend.protocol

Options	<p>esvProxy</p> <p>The default</p> <p>slp</p> <p>Advertising Services discover one another by multicasting. Service ads are kept in an in-memory internal data structure. It is non-persistent and less scalable. Due to multicasting limitation, it usually cannot be used to associated Advertising Services across different subnets (LAN segments).</p> <p>ldap</p> <p>The Advertising Service makes its own connections to a common LDAP directory server. Ads are persistent and much more scalable.</p> <p>esvProxyOnly</p> <p>Do not revert to the <code>slp</code> option. This option uses a proxy service to keep service advertisements. By default it uses Advertising proxy of ESV and thus enable services to be advertised and found in the HP sponsored global service directory.</p>
Modify when	<p>When a specific backend mode should be used. In general, use SLP for simple localized deployment and enterprise deployment. Use <code>esvProxy</code> to connect to ESV (or if you want to use your customized proxy service that, for example, provides access control to your internal LDAP server.)</p>
Impact	<p>Your choice of backend mode largely affects the reaches of service advertisements. It determines whether service advertisements are persistent in case the Advertising Service goes down. It also affects the scalability of Advertising Service.</p>
See also	<p>For SLP mode, see the various multicast related properties in The Multicast Settings section.</p> <p>For LDAP mode, see the LDAP Settings section for information on available parameters for assorted LDAP settings.</p> <p>For <code>esvProxy</code> mode, see the ESV Proxy Settings section for information on available parameters for assorted ESV Proxy settings.</p> <p>This parameter is equivalent to the <code>-beproto</code> option in previous versions of the Advertising Service.</p>

E-services Village Proxy Settings

NOTE: The parameters in this section are only applicable if you set `net.espeak.services.advService.backend.protocol` to `esvProxyOnly`.

`net.espeak.services.advertise.esv_user_name`

`net.espeak.services.advertise.esv_password`

Purpose	Specifies the e-services Village username and password A username and password are provided for connecting to ESV Proxy settings.
Default	The e-services Village guest login and password
Options	Any valid username and password
Modify when	Set these parameters if you have an e-services Village account.

Configuring Logging

The following properties provide the logging configuration information for any logging process. The default loggers are the Error and Activity loggers. Do not remove these loggers.

You can provide your own loggers for your e-speak programs and take advantage of the e-speak logging framework to invoke and use loggers by simply adding the logger to the loggerManager list. You can then provide one or more of the properties described in this section.

loggerConfig.loggerManager.loggerList

Purpose	<p>Specifies a list of loggers that are used. The logger records activities and error messages that occur in the e-speak system. You can also specify a customized log. This parameter is a list and must be specified as follows:</p> <pre>loggerConfig.loggerManager.loggerList.<index>= <loggerName></pre> <p><index> is the index number (starting from 0) of the logger</p> <p><loggerName> is the name of the logger.</p> <p>The loggerName defines the properties of the logger later in the file. The loggerManager maintains a list of all the loggers and provides the loggers programmatically through its API. The default value for this parameter is to use the <code>ErrorLogger</code>. This default is used because each time the e-speak engine starts, the error messages should be logged no matter whether the parameter is specified or not. Therefore, you should not modify the existing values. If you want to add new loggers, you can add them to the list. The e-speak distribution comes with the following three loggers already set.</p> <pre>loggerConfig.loggerManager.loggerList.0=ErrorLogger loggerConfig.loggerManager.loggerList.1= ActivityLogger loggerConfig.loggerManager.loggerList.2= SecurityLogger</pre>
---------	--

loggerConfig.loggerManager.loggerList

Purpose, continued	<p>This property is only for the logger manager. Each individual logger is configured using the following property definition:</p> <pre>loggerConfig.<loggerName>.<property>[.<index>]=<value of property></pre> <p>where</p> <p><loggerName> is the same name as defined in the loggerConfig.loggerManager.loggerList property.</p> <p><property> is the logger property</p> <p><index> is the index if the property is of a list type.</p> <p>For example if you define:</p> <pre>loggerConfig.loggerManager.loggerList.3=MyLogger</pre> <p>Define the properties for this logger as</p> <pre>loggerConfig.MyLogger.class=mypackage.MyLogger</pre>
Default	<pre>loggerConfig.loggerManager.loggerList.0= ErrorLogger loggerConfig.loggerManager.loggerList.1= ActivityLogger loggerConfig.loggerManager.loggerList.2= SecurityLogger</pre>
Options	The name of the logger is required and this name can be any valid name.
Modify when	New loggers need to be added
See also	loggerConfig.<loggerName>.class loggerConfig.<loggerName>.logDir loggerConfig.<loggerName>.logFileName loggerConfig.<loggerName>.logLevel

loggerConfig.<loggerName>.class

Purpose	<p>Defines the name of the logger class</p> <p>The <i><loggerName></i> is the name of the logger specified in the <i>loggerList</i> property of the <i>loggerManager</i>. This class will be loaded by the <i>LoggerManager</i> and will be responsible for performing all the logging. This class can make use of the various components like the formatter, a handler to aid the logging process. This property is not mandatory and if it is not supplied the default "net.espeak.util.logger.ESLogger" will be used. Users can create their own loggers and supply the class here. This property helps users define their own logging. An example of how to define this property is as follows:</p> <pre>loggerConfig.ErrorLogger.class=net.espeak.util.logger.ESLogger</pre> <p>where <i>ErrorLogger</i> is defined in the <i>loggerList</i> parameter.</p>
Default	net.espeak.util.logger.ESLogger
Options	You can specify a fully qualified and existing class name for the logger.
Modify when	You add a new logger is added, a new property can be created that specifies the class to be used for the logger.
Impact	The specified class is loaded and is available as a new logger. If the class cannot be loaded, it is ignored.

loggerConfig.<loggerName>.logDir

Purpose	<p>Specifies the directory where the log files will be created. The log files are created in a directory called "logs" under the <i>logDir</i> specified. For example, if the <i>logDir</i> was specified as <i>C:\e-speak</i> the logs are created under the directory <i>C:\e-speak\logs\</i>. The default directory is the e-speak installation directory. An example is the directory where the <i>ESPEAK_HOME</i> environment variable is set. The directory for <i>ErrorLogger</i> is specified as:</p> <pre>loggerConfig.ErrorLogger.logDir=\$espeak_home</pre> <p>This is the e-speak home directory.</p>
Default	The e-speak installation directory. For example, the directory where the <i>ESPEAK_HOME</i> environment variable is set.

loggerConfig.<loggerName>.logDir

Options	Users can specify any valid directory name. If the directory does not exist, it is the log handler's responsibility to take care of such situations. The default log handler creates any non-existing directories before creating the log files.
Modify when	To change the directory into which log files are written
Impact	Log files are written to the new directory
See also	loggerConfig.<loggerName>.logFileName

loggerConfig.<loggerName>.logFileName

Purpose	<p>Specifies the name of the log file. The default value for the log file is <code><loggerName>.log</code>. For <code>ErrorLogger</code> the log file name is specified as:</p> <pre>loggerConfig.ErrorLogger.logFileName=error.log</pre> <p>Thus the fully qualified path for the log file is <code><logDir>/logs/<logFileName></code> on Unix systems and <code><logDir>\logs\<logFileName></code> on Windows.</p> <p>Since this file and the logger are used on a per JVM basis, having 2 JVMs access the same configuration may lead to a problem in the log file output, the messages may be mixed from the 2 JVMs. To avoid such mixed output, use separate configuration files for every JVM modifying the name of the log file. This same can be also achieved using the same config file but commenting the <code>logFileName</code> property for the logger and specifying this property on the command line of the JVM like:</p> <pre>espeak RC class= -DloggerConfig.ErrorLogger.logFileName=mylog</pre>
Default	<code><loggerName>.log</code>
Options	Any valid filename can be used.
Modify when	You want to change the log file.
Impact	Log messages are written to the new file.
See also	loggerConfig.<loggerName>.maxNumFiles loggerConfig.<loggerName>.maxFileSize

loggerConfig.<loggerName>.logLevel

Purpose	<p>Defines the logging levels or severity of the messages that the logger records. The log levels supported in this release are:</p> <ul style="list-style-type: none"> • all — Logs all messages • info — Logs information messages • warning — Log warning messages • error — Logs error messages • emerg — Logs emergency or fatal messages • emerg — Logs emergency or fatal messages <p>A combination of different levels can be specified as a comma separated list, for example, <code>warning,info</code> to specify the logger should only record information and warning messages. The <code>logLevel</code> for the <code>ErrorLogger</code> is:</p> <pre>loggerConfig.ErrorLogger.logLevel=all</pre>
Default	all
Options	all info warning error emerg
Modify when	You should not change this property.

Chapter 6 E-speak Desktop

This chapter discusses:

- **Basic Concepts** — the basic concepts, including the e-speak model and Desktops
- **User Interface and Functionality** — basic functions such as loading plug-ins, viewing loaded plug-ins, and managing e-speak
- **Desktop Management** — how you can use e-speak Desktop plug-ins to perform complex management functions
- **Quick Start** — a walk-through of Desktop functions
- **Desktop Reference** — information on Desktop menus, options, commands, and actions

Desktop Interface

The Desktop displays e-speak objects in a tree view; the top of the tree is the e-speak system. Highlight/select objects in the tree by left-clicking the mouse. A selected object has a blue background.

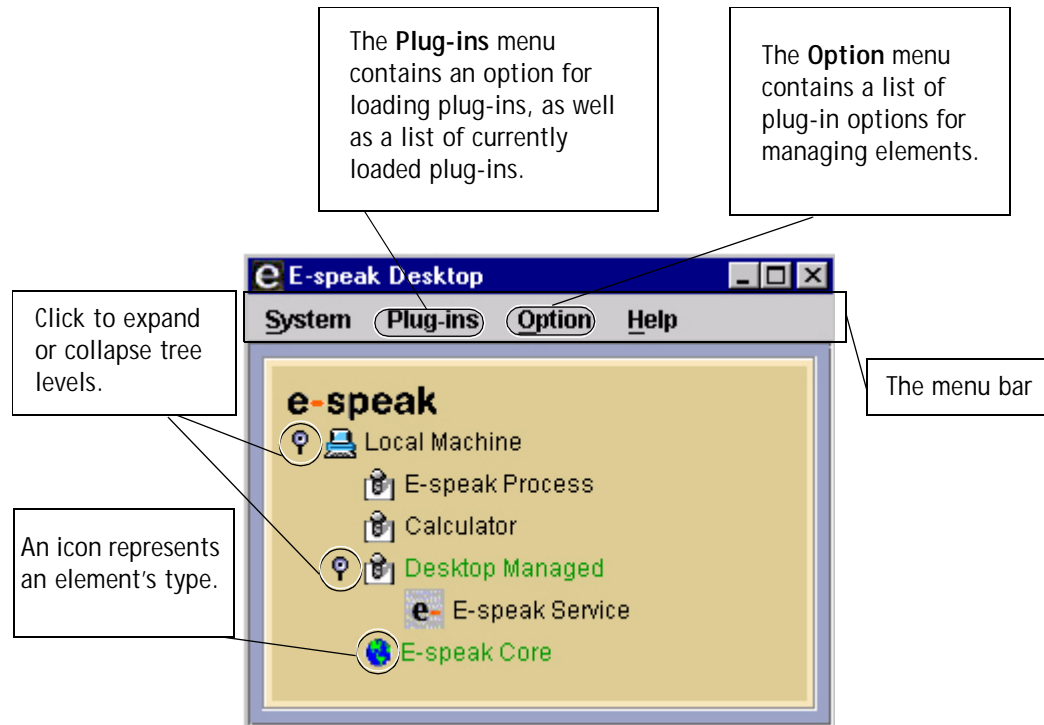


Figure 2 Desktop interface components

Right-click an object to access a pop-up menu containing options and/or processes that pertain to that object, such as plug-in options. Right-clicking multiple selected objects displays a pop-up menu that contains batch processes.

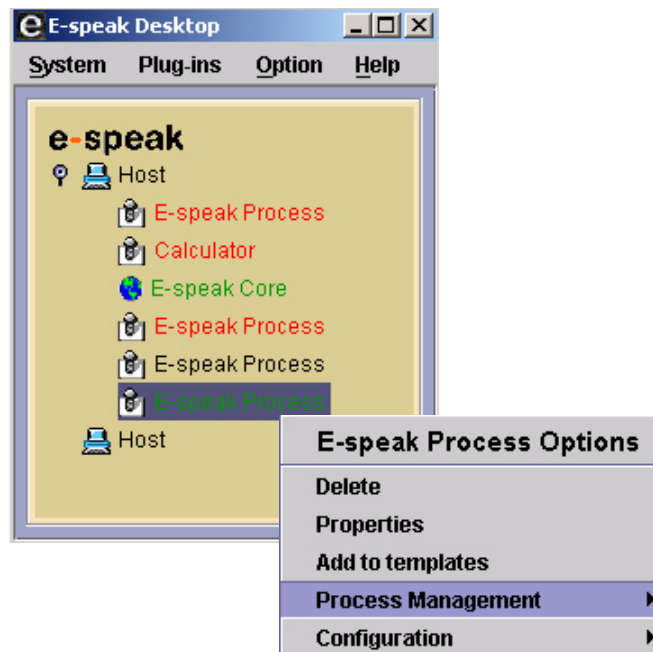


Figure 3 Pop-up menu options

Installing a Plug-in

In the e-speak Desktop:

- 1 Select **Plug-ins** to open the **Configuring** window.



Figure 4 Configuring window

- 2 Enter the plug-in class name. For example:

```
net.espeak.management.desktop.plugin.ProcessManager.  
ProcessManagerPlugin
```

- 3 Click **Accept**.

Desktop Management

The default configuration file provided with the Desktop automatically loads the plug-ins listed in [Table 18](#).

Table 18 Basic operations

Plug-in	Plug-in Java Class™
Service Manager	net.espeak.management.desktop.plugin. ServiceManager.ServiceManagerPlugin
e-speak configuration plug-in	net.espeak.managment.desktop.plugin. espeakConfiguration.espeakConfigurationPlugin
Element Manager	net.espeak.management.desktop.plugin. ElementManager.ElementManager
Core Manager	net.espeak.management.desktop.plugin.CoreManager. CoreManagerPlugin
Configuration Manager	net.espeak.management.desktop.plugin. ConfigurationManager.ConfiguratorPlugin
Process Manager	net.espeak.management.desktop.plugin. ProcessManager.ProcessManagerPlugin
Template Manager	net.espeak.management.desktop.plugin. TemplateManager.TemplateManagerPlugin

The Desktop Configuration File

The Desktop uses an XML configuration file that contains all of the element type schema/model information and the plug-in class names. The examples outlined in this section are based on the default file provided with the Desktop:

```
$ESPEAK_HOME/config/management/espeakmodel.xml
```

The configuration file contains the following code:

```
<?xml version="1.0" ?>
<DesktopConfiguration>
  <ElementTypes>
    <Host icon="compblue.gif" Description="An e-speak managed host"
      HostDomainName="127.0.0.1" Caption="Host" />
    <Process parent="Host" icon="screw.gif" Description="An e-speak
      managed process" CommandLine="!NULLVALUE!" Status="0"
```

```
ManagedTask.port="!NULLVALUE!" Caption="E-speak Process" />
<Core inherits="Process" parent="Host" icon="core.gif"
  Description="An e-speak core" CommandLine="java
  net.espeak.infra.core.startup.StartESCore -pop esip:2950"
  CorePort="!NULLVALUE!" Caption="E-speak Core" />
<Service parent="Process" icon="userDefinedEspeak.gif"
  Description="An e-speak service" Caption="E-speak Service"
  ServiceName="!NULLVALUE!" />
</ElementTypes>
<Plugins>
  <Plugin
    class="net.espeak.management.desktop.plugin.ServiceManager.Service
    ManagerPlugin" />
  <Plugin
    class="net.espeak.management.desktop.plugin.ElementManager.Element
    Manager" />
  <Plugin class="net.espeak.management.desktop.plugin.CoreManager.
    CoreManagerPlugin" />
  <Plugin
    class="net.espeak.management.desktop.plugin.ConfigurationManager.
    ConfiguratorPlugin" />
  <Plugin
    class="net.espeak.management.desktop.plugin.ProcessManager.Process
    ManagerPlugin" />
  <Plugin class="net.espeak.management.desktop.plugin.TemplateManager.
    TemplateManagerPlugin" />
</Plugins>
</DesktopConfiguration>
```

Starting the Desktop

Start the e-speak Desktop from the operating system desktop or from a command line.

Table 19 Starting the e-speak Desktop

<p>To start the Desktop from your operating system desktop:</p>	<p>Select Start>Programs>e-speak>e-speak Desktop.</p>
<p>To start the Desktop from a command line:</p>	<p>If you want to start from ESPEAK_HOME, type:</p> <pre>bin\espeak RC class=net.espeak.management. desktop.ESpeakDesktop arg=-model FILE_NAME</pre> <p>or</p> <p>change the current working directory to ESPEAK_HOME\bin and type:</p> <pre>espeak RC class=net.espeak.management. desktop.ESpeakDesktop arg=-model FILE_NAME</pre> <p>FILE_NAME is the name of the Desktop configuration file (usually <code>espeakmodel.xml</code>). Enter this file name at the command line or select the file from the file list when the Desktop load procedure begins.</p> <p>If you specify ESPEAK_HOME or start the desktop from ESPEAK_HOME, the default file is <code>espeakmodel.xml</code>. You can find this file in the directory <code>ESPEAK_HOME/config/management</code></p>

Accessing the System Pop-up Menu

The e-speak logo is the top of the management hierarchy. Right-click the logo to access the system pop-up menu.



Figure 5 System pop-up menu

The pop-up menu contains three options:

- **New Host** creates a new e-speak-enabled machine.
- **Properties** opens the **Element Properties Editor**.
- **System Settings** provides a variety of options. Refer to the [Table 24](#) for details.

Connecting to an e-speak-Enabled Host

Connecting to an e-speak-enabled host allows you to manage elements on different machines.

On the desktop:

- 1 Right-click the **e-speak logo**.
- 2 Select **New Host** from the pop-up menu.
- 3 Right click the new Host.
- 4 Select **Properties**.

By default, the `HostDomainName` is set to your local host (127.0.0.1).

NOTE: Make sure that the `ProcessManager` is running on the e-speak enabled remote Host

- 5 Set the **HostDomainName** to the appropriate machine.

NOTE: You cannot change this property once it is in use.

- 6 Click **Apply**.

- 7 Click **Close**.

You can save an active Host to a connection group, which you can then use to connect to a set or domain of hosts.

Starting a Process

The Process Manager has to be started in interactive mode on the NT p[lat]form in order to view the command prompt window from where the application is launched. Refer to the section “[Working with the Desktop](#)” in the [Chapter 10](#), “[Troubleshooting](#)”.

On the Desktop:

- 1 Right-click **Host**.
- 2 Select **Create New Process**. The new process appears under Host.
- 3 Right-click the new process.
- 4 Select **Process Management > Start Process**.

NOTE: Make sure that samples are compiled and copied to the directory `<ESPEAK_HOME>\lib`

or

give `CLASSPATH` while entering the command line using the switch `-cp` or `-classpath` (e.g. `java -cp<CLASS_PATH><FILE_NAME>`).

- 5 Enter the command line:

```
java CLASSNAME=samples.ManagedCounter.ManagedCounter
```

NOTE: If `CLASSNAME` is given, no `classDefFound` error is thrown.

The Process caption on the Desktop turns green when the process activates.

6 Right-click E-speak Process.



Figure 6 E-speak process

7 Select Process Management > Process Output to verify that the process is running.

Configuring Multiple Processes

You can configure, start, and stop, multiple processes using batch processes.

On the Desktop:

- 1 Highlight the processes on the Desktop.
- 2 Right-click any highlighted processes.
- 3 Select the appropriate process from the **Batch Element Options** pop-up menu.

Managing Runtime Configuration

On the Desktop:

- 1 Right-click **E-speak Process**.
- 2 Select **Configuration > Manage Runtime Configuration** to open the **Configuration Variable Editor**.
- 3 Adjust the configuration variable as necessary.

4 Set the three Managed Counter variables.

Table 20 Managed Counter Variables

Variable	Type	Description
Countup	Boolean	Set the variable to True to increment the count (1, 2, 3...) or False to decrement the count (...3, 2, 1)
Increment	Integer	Specify the amount the counter changes incrementally at each count
Interval	Integer	Specify the amount of time, in milliseconds, between each increment

NOTE: You can modify the counter's runtime behavior by adjusting these variables after the counter starts.

Counter Controls

On the Desktop:

- 1 Right-click **E-speak Process**.
- 2 Select **Service Control** to open the **Service Manager Control** window.

This window contains the counter's management interface controls. The managed counter example uses the management interface `DefaultManagedServiceIntf`. Most simple e-speak services and components use this interface.

- 3 Click **Init** to initialize and run the counter.

Refer to "[Service Control Window Menu and Options](#)" for information on the **Service Manager Control** window components.

Stopping a Process

On the Desktop:

- 1 Right-click **E-speak Process**.
- 2 Select **Process Management > Stop Process**. The E-speak process label on the Desktop turns red to indicate that the process has stopped.



Figure 7 Stopped process

Discovering Manageable e-speak Components and Services Running in a Managed Process

E-speak services are not explicitly created in the Desktop. During regular Desktop operation, new manageable e-speak services are automatically detected and created. If a service is not automatically detected, you can request an explicit query.

To perform an explicit lookup for services on a specific process: Right-click on the e-speak Process and select **Process Management > Find Hosted Services** from the pop-up menu.

Managing e-speak Components and Services

E-speak Services are components of a program that utilize the e-speak service management library to gain manageability.

To manage an e-speak Service, select the **Service Control** option from its pop-up menu. For more information see [“Service Control Window Menu and Options”](#) on page 73.

Create New Cores

On the Desktop:

- 1 Right-click the host on which you wish to run the core.
- 2 Select **Create New Core** from the pop-up menu.

A new Core appears. By default, the first section of the element’s command line property is already set to:

```
espeak RC class=net.espeak.infra.core.startup.StartESCore arg="--pop  
esip:"
```

NOTE: This example command line is incomplete. Please refer to the appropriate core user documentation for information on the command line options for the core.

- 3 Set this is property to:

```
espeak RC class=net.espeak.infra.core.startup.StartESCore  
arg="--pop esip:2950"
```

When this process starts, it will run a core using port number 2950.

- 4 Right-click the Core.
- 5 Select **Process Management > Start Process** the pop-up menu. The Core process on the Desktop turns green, indicating that it is running.

Shutting Down the Core

On the Desktop:

- 1 Right-click the Core.
- 2 Select **Core Management > Shutdown Core** from the pop-up menu.

The Core Browser

The Core Browser provided with the Desktop allows you to browse a core by vocabulary, by services registered in individual vocabularies, and by the vocabularies registered to individual services.

In the Desktop:

- 1 Right-click the Core.
- 2 Select **Core Management > Open Core Browser** from the pop-up menu.

The the left column of the Core Browser window contains a list of the vocabularies registered to the core's repository. When you select one of these vocabularies, a list of hyperlinked services registered in the vocabulary appears in the right column.

Select a hyperlink to open a service browser. The service browser displays the descriptions, vocabularies, and contracts associated with the selected service. You can edit service descriptions and add any attributes available in the vocabulary with which the service is registered. You can edit only attribute value types that can be expressed in text form.

Provided Plug-ins

Process Manager Plug-in

The Process Manager is an e-speak service that enables clients to run processes on any e-speak enabled machine. It has two main functions:

- Launch manageable processes
- Store runtime configuration properties for processes

These may include:

- The URLs for e-speak services running within processes
- The URL of the runtime configuration access for processes
- The element's process type
- Fields unique to individual processes, such as core ports and protocols

You can establish an event channel between the Process Manager and its clients to allow clients to monitor property modifications. The Process Manager plug-in is this type of client and can translate the implications of certain property settings into meaningful events within the Desktop. For example, when a new manageable

component registers itself using the service management library, a process property is set in the Process Manager with the service's URL as the value. The plug-in detects the setting through the event channel and creates a new e-speak Service to represent this new manageable component. Each e-speak-enabled machine has a single Process Manager agent and each machine is represented as a Host.

Process Lookup

The plug-in can search for processes not currently managed by the Desktop. When this query finds any such processes, it creates a new element with the appropriate element type. Normally, this search process is automatic. You can perform an explicit lookup using the **Process Lookup** option.

NOTE: Discovered e-speak process elements are non-persistent by default.

Service Lookup

The plug-in can search a process for subcomponents that were made manageable using the e-speak Service Management Library. If this query discovers any non-managed subcomponents, it instantiates them in the Desktop as new e-speak Service. Normally, the plug-in automatically finds new services. You can perform an explicit lookup using the **Find Hosted Services** option.

Persistent Processes

The plug-in allows you to control whether a process is persistent or non-persistent. Persistent processes auto-restart when they stop. The Process Manager maintains a persistent list of these processes. When the Process Manager restarts, this list starts much like a script.

NOTE: Use caution when setting a process as persistent. A persistent process runs indefinitely until the host machine is shutdown or you set the process as non-persistent.

Sticky Processes

The Process Manager plug-in registers a **Set Sticky** option for e-speak processes. A sticky process exist even if the process stops (persistent). Non-sticky process are deleted from the Desktop when they stop (non-persistent). By default, all processes you create are sticky (the **StickyProcess** property is set to true).

Set a Process as Persistent

In the Desktop:

- 1 Right-click the process.

- 2 Select **Process Management** > **Set persistent** from the pop-up menu.

Set a Process as Non-Persistent

In the Desktop:

- 1 Right-click the process.
- 2 Select **Process Management** > **Set non-persistent** from the pop-up menu.

Host Groups

Hosts represent individual e-speak-enabled machines. They contain properties such as the HostDomainName (the address of the machine) and a handler for interactions with Process Manager agents running on the machine.

Connection Groups

The Process Manager plug-in offers **connection groups** that allow you to manage multiple connections. Each connection group is a list of Hosts that you can load into the Desktop, connect to, and query for available manageable processes and services.

Loading a connection group initializes an e-speak management domain. You can store connection groups in a file that you can distribute via e-mail for other administrators to use.

NOTE: By default, the file name is `espeak_home/config/management/pmp_conf.xml`.

Connection Group Manager

The Process Manager plug-in includes a simple tool to manage connection groups.

In the Desktop:

- 1 Right-click **e-speak**.

- 2 Select **System Settings > Manage Connection Groups** from the pop-up menu.



Figure 8 Connection group manager

Create a Connection Group

In the Connection Group Manager:

- 1 Click **Add Group**.
- 2 Enter a group name.
NOTE: The name cannot contain blank spaces.
- 3 Select **Save All** to save all connected Hosts to the group.

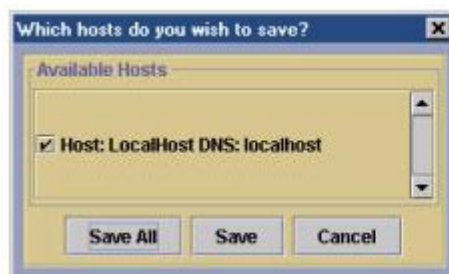


Figure 9 Host selector window

- 4 Specify whether or not the group will be automatically connected to at start-up.

NOTE: You can toggle this mode on and off by right-clicking the group and selecting **enable** or **disable** from the pop-up menu. The Process Manager plug-in automatically connects to any groups that have auto-connect enabled.

Connect to a Connection Group

In the Connection Group Manager:

- 1 Select the group.
- 2 Click **Connect**.

View Connection Group Information

In the Connection Group Manager:

- 1 Select the group.
- 2 Click **Information**.



Figure 10 Connection group information

Save Hosts to the Default Connection Group

E-speak includes a default connection group that you can use without accessing the Connection Group Manager.

In the Desktop:

- 1 Right-click **e-speak**.
- 2 Select **System Settings > Save Active Hosts as Default Connections** from the pop-up menu. All currently-connected Hosts are saved to the default group.

Connect to the Default Connection Group

In the Desktop:

- 1 Right-click **e-speak**.
- 2 Select **System Settings > Make Default Connections** from the pop-up menu.

The Element Manager Plug-in

The Element Manager plug-in allows you to delete elements and edit element properties. The plug-in provides an Element Properties Editor that you can use to add and modify properties.

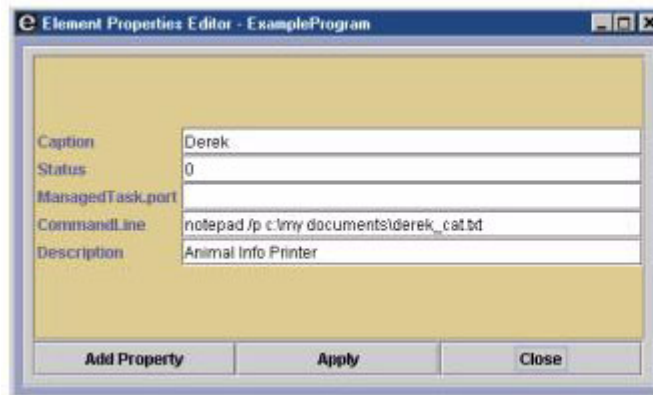


Figure 11 Element Properties Editor

The Configuration Manager Plug-in

The Configuration Manager plug-in allows you to manipulate process runtime configuration variables and edit configuration files. By managing this data, you can dynamically control the behavior of e-speak components.

Selecting **Configuration > Manage Runtime Configuration** from the process pop-up menu displays the runtime configuration variable editor.

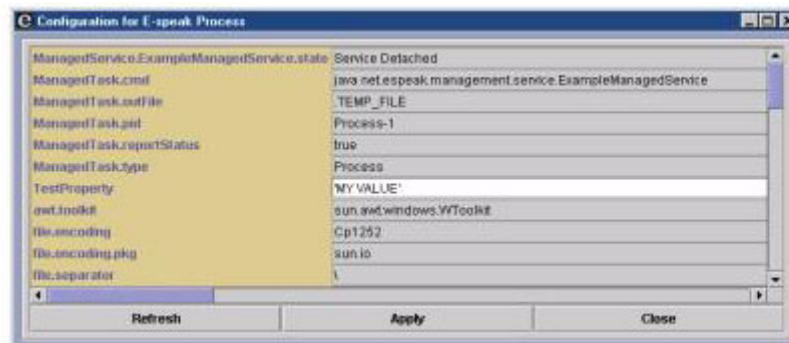


Figure 12 Variable editor

NOTE: You will receive a warning if you change a previously numeric or boolean (true or false) value to a different type. This ensures that you do not accidentally corrupt the configuration with an invalid value.

Selecting **Configuration > Edit Configuration File** from the process pop-up menu opens a standard text editor in which you can modify a configuration file.

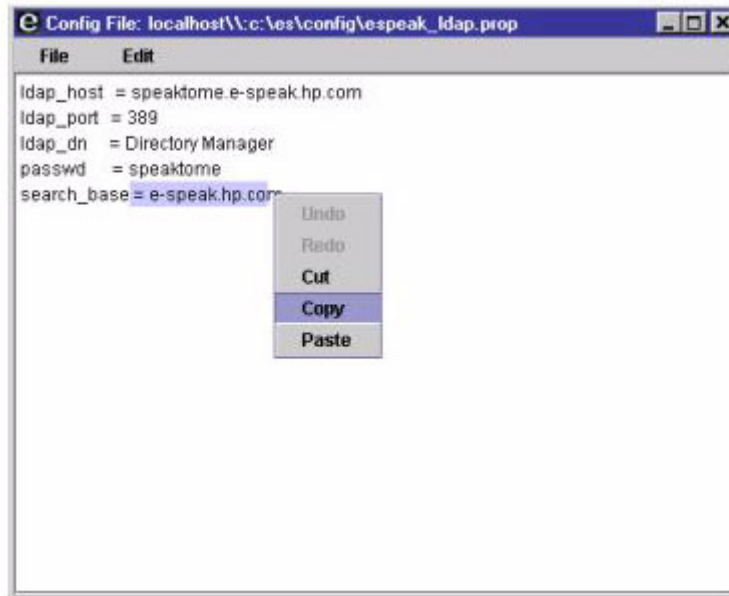


Figure 13 Configuration file editor

The configuration file editor includes standard editing features such as **Copy**, **Paste**, **Cut**, **Undo** and **Redo**.

NOTE: If the you are prompted to enter the path of the configuration file, then it is likely that the process cannot be configured in this way.

The Service-Manager Plug-in

The Service Manager plug-in allows the Desktop to manage components that use the service management library. The plug-in provides a **Control Service** option for e-speak Services that opens the e-speak **Service Manager Control Window**.



Figure 14 Generic Service Manager Control window

This window's appearance depends on the type of management interface provided by the manageable component. More common management interfaces may have more specialized interface support, e.g. the default service management interface is designed to be applicable to most basic e-speak services. The window displays all possible states and transitions to describe the state of the machine.

The Core Manager Plug-in

The Core Manager plug-in allows you to manage e-speak cores that use a customized implementation of the management libraries. These libraries:

- Transmit the information required to contact the core directly to the Process Manager service
- Provide an interface that enables a remote shutdown of cores using the Desktop

Using the core address information, the plug-in makes a direct connection to the Core represented by Cores. The plug-in provides a new Core browser that presents the core in terms of vocabularies, services registered in individual vocabularies, and the vocabularies registered to individual services. You can switch the browser to service-centric display to browse individual service resources.

You can edit Service descriptions and add attributes to the descriptions from the attributes available in the vocabulary with which they are registered. You can only edit value types that can be expressed in text form.

Desktop Walkthrough

This walkthrough introduces Desktop features and operations. For best results, work through this section from beginning to end.

1 Click Start > e-speak > e-speak Desktop.

or

Type this command at the command line:

```
java CLASSNAME=net.espeak.management.desktop.ESpeakDesktop  
arg="-model FILE_NAME"
```

FILE_NAME is the name of the Desktop configuration file (usually `espeakmodel.xml`). If you do not enter the name in the command line, you will be prompted to select the appropriate file before the Desktop loads.

TIP: You can use the `-nosplash` to disable the splash screen, if desired.

If you receive an error while the Desktop is loading, it is due to an error in the configuration file or an incorrectly-specified location for the file.



Figure 15 e-speak Desktop

2 Right-click e-speak.

3 Select New Host.

NOTE: By default hosts point to the local host.

4 Right-click Host.

- 5 Choose **Properties**. The **Element Properties Editor** window appears.



Figure 16 Element Properties Editor

- 6 Modify the **HostDomainName** to reflect the IP address or domain name of the machine that you want to manage, if you are configuring a remote machine.
- 7 Right-click **Host**.
- 8 Select **Create New Process**.

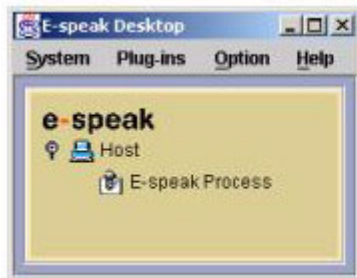


Figure 17 A newly-created process

- 9 Right-click the e-speak process.
- 10 Select **Process Management > Start Process** to connect to the Host.
- 11 Enter `calc` in the **Configuring** window.
NOTE: This command is specific to Microsoft Windows.
- 12 Click **Accept** to run the Microsoft Calculator through the Process Manager.
- 13 Right-click the process in the Desktop.
- 14 Select **Process Management > Stop Process** from the process pop-up menu.

15 Right-click the process.

16 Select **Process Management > Set Persistent** from the process pop-up menu.

Setting the process as persistent ensures that the process is always running when the Process Manager is running. If the process (the calculator in this example) shuts down, it automatically restarts if the Process Manager is running. If the Process Manager shuts down, then the process restarts when the Process Manager restarts.



Figure 18 Persistent process

17 Right-click the active (green) calculator process

18 Select **Process Management > Stop Process**. The process icon is removed.

19 Right-click the stopped (red) process.

20 Select **Properties** to open the **Elements Properties Editor** appears.



Figure 19 Element Properties Editor

21 Enter this code as the command line:

```
java CLASSNAME=samples.ManagedCounter.ManagedCounter.
```

22 Click **Close**.

23 Click **Yes** to accept the changes.



Figure 20 Confirmation request

24 Right-click the process.

25 Select **Process Management > Start Process**. The Managed Counter process starts. And the service icon for this service appears under the process.



Figure 21 Started process

This process now shows a manageable service.



Figure 22 A started process and its manageable service

NOTE: This is the default service state model. Other service developers can have a different state models when the default model does not apply to their services. If a service uses another state model that is not supported by the Desktop, the Desktop provides a generic service manager control window.

- 26 Right-click the Service to change it to **ManagedCounter**.
- 27 Right-click the managed counter.
- 28 Select **Service Control** to view the **Service Manager Control Window**.
- 29 Right-click a process.
- 30 Click **Process Management > Process Output**. The **Process Output** window displays the output of the process as it is running.

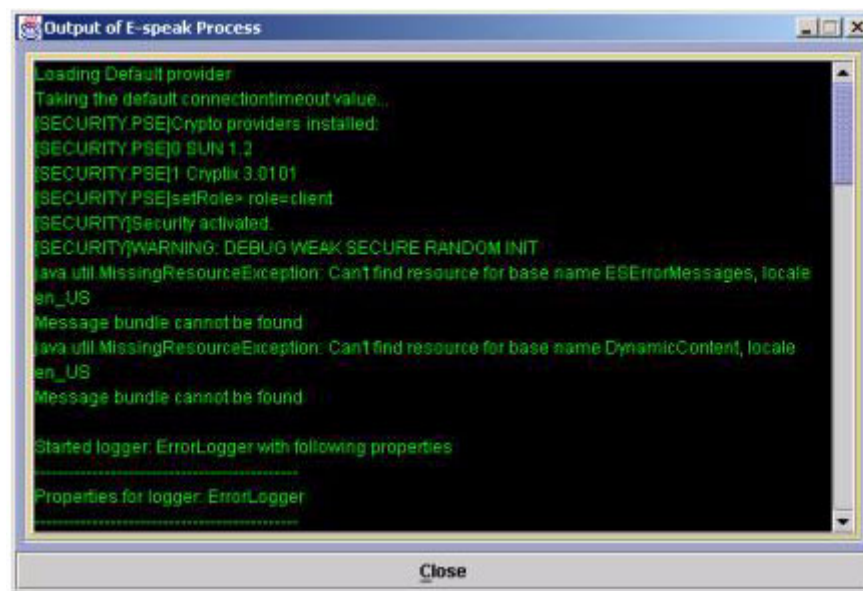


Figure 23 The process output window

- 31 Right-click a process.
- 32 Select **Configuration > Manage Runtime Configuration** to retrieve the process runtime variables.

The runtime configuration editor displays all of the process configurations. Some are editable, such as the first three lines in this example.

You can change these configurations, if necessary, to modify the service's behavior.

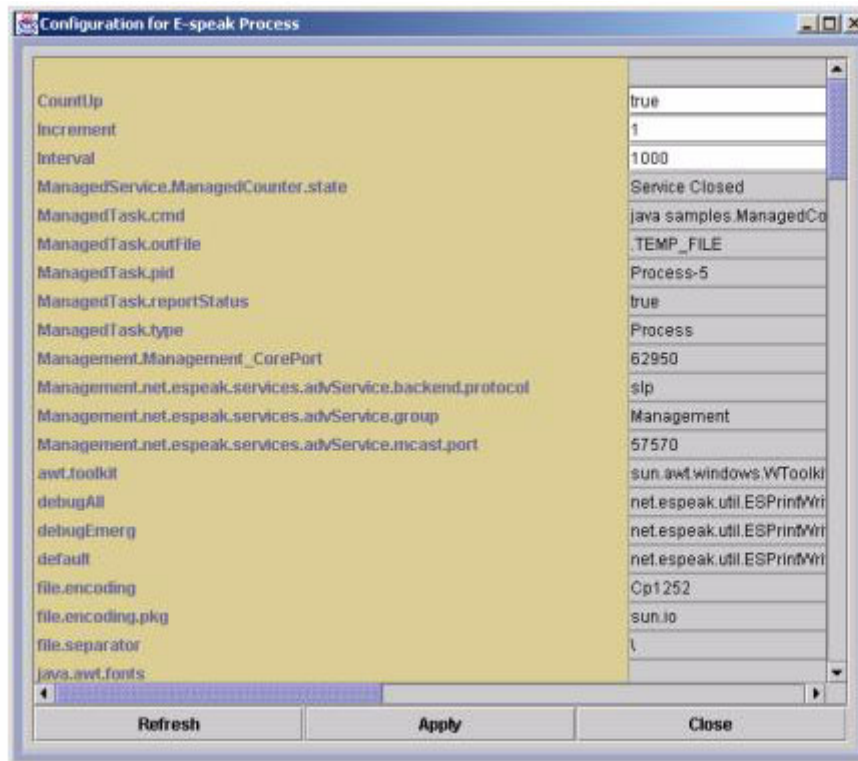


Figure 24 Process configuration

33 Right-click a process.

34 Select **Configuration > Edit Configuration File** to edit the configuration file.

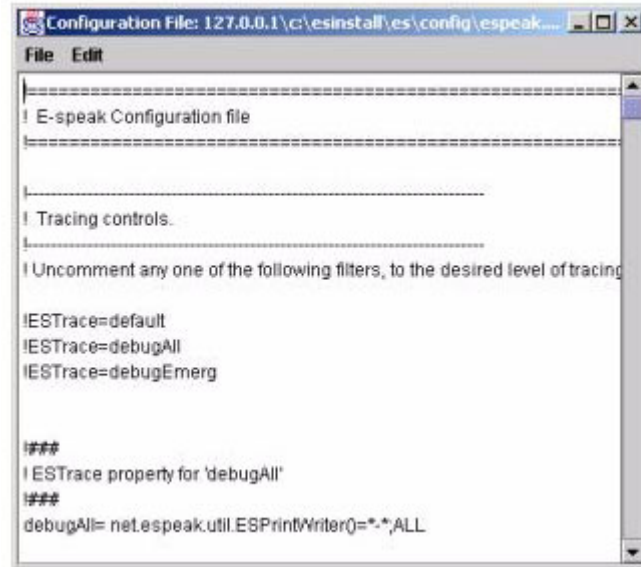


Figure 25 Configuration file editor

If the active process uses a configuration file during initialization, you can edit it remotely using the Desktop.

35 Enter the location of the config file in the text box that appears on the screen.

36 Right-click a host.

37 Select **Edit Configuration**.

The e-speak configuration editor allows you to edit the `espeak.cfg` file.

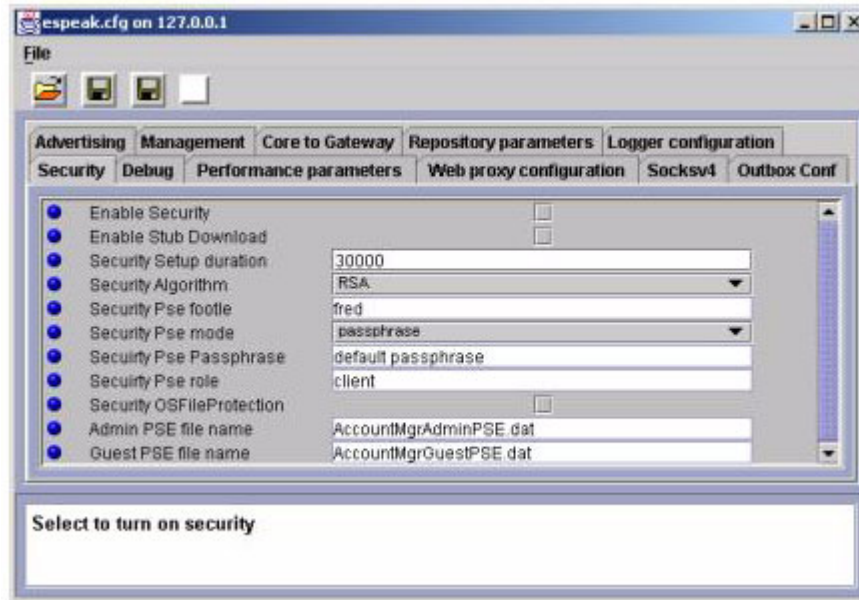


Figure 26 Configuration editor

38 Right-click a host.

- 39** Select **Create Process from a Template** to display the **Please select a template** window.

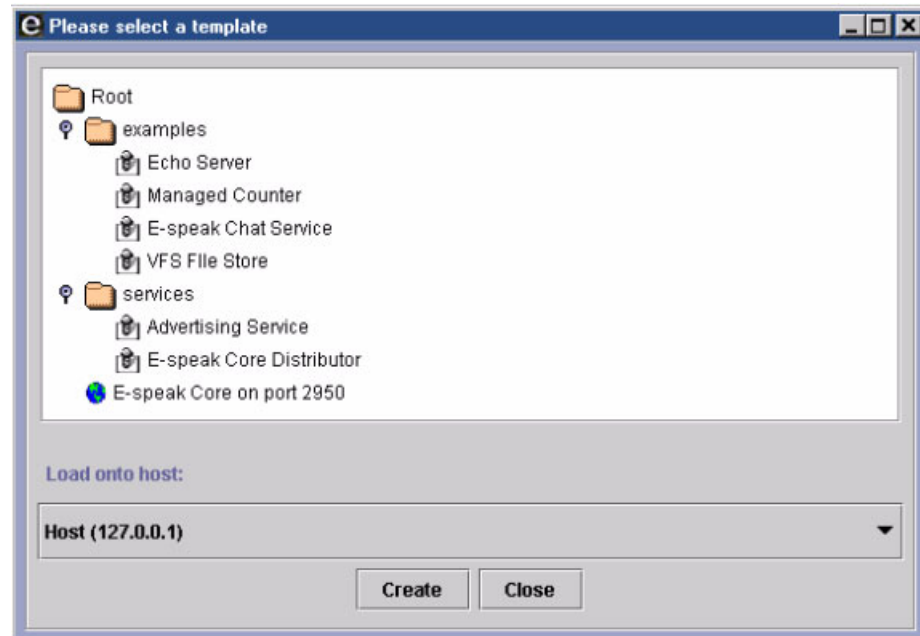


Figure 27 Template selector

- 40** Highlight a process.
- 41** Click **Delete**.
- NOTE: This does not stop the process; it simply removes its icon from the Desktop.
- 42** Select **Host > Process Lookup** to find all manageable processes running on the machine.
- 43** Highlight a process.
- 44** Select **Process > Add to templates**. The **Please select a destination folder** window opens.
- 45** Select a folder (such as `Root`) in which to save the process.
- 46** Enter a name for the process in the **Save template as** field.
- 47** Click **Save**.

- 48 Right-click a host.
- 49 Select **Create a Core**.



Figure 28 Creating a Core

NOTE: After creating a core, you may want to edit the command line properties to configure the core's initial settings.

- 50 Right-click the core.
- 51 Select **Process Management > Start Process** to start the core. The core turns green.
- 52 Right-click the host.
- 53 Select **Create Process** from a template to display the window **Please select a template**.
- 54 Highlight the **Advertising Service**.
- 55 Click **Create**. (This will create an Advertising Service icon on the desktop).
- 56 Right click the **Advertising Service** icon.
- 57 Select **Process Management > Start Process**.
- 58 Right-click the core.

59 Select Core Management > Open Core Browser.



Figure 29 Core browser

60 Select Base Vocabulary in the left panel. Services display in the right panel of the browser as hyperlinked URLs.

61 Select a hyperlinked URL to view the service resources.

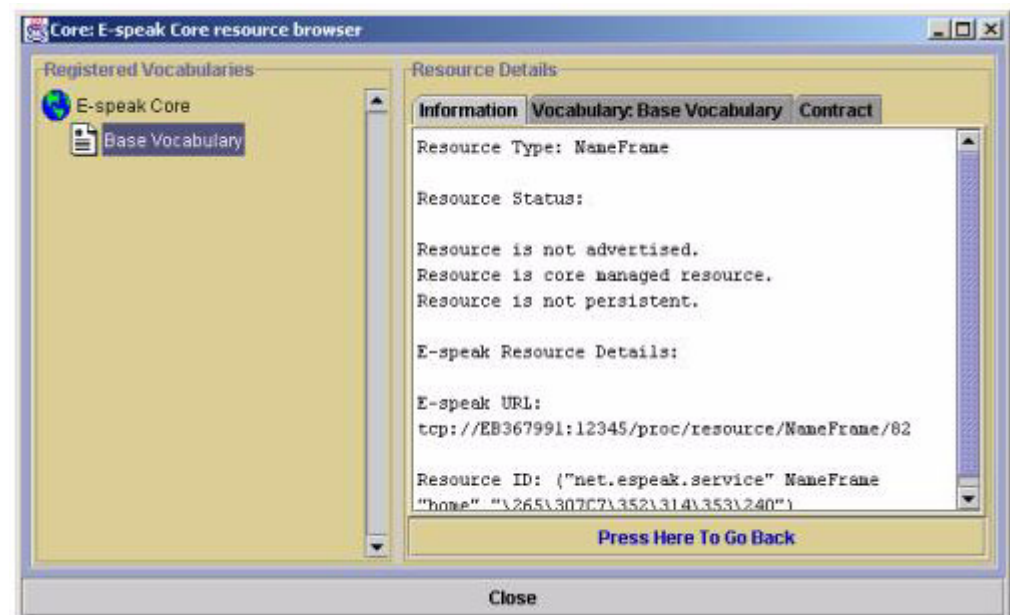


Figure 30 Core browser information tab

62 Right-click the core.

63 Select Core Management > Shutdown Core.

Desktop Reference

The Desktop main menu includes these menus and tabs.

Table 21 System menu options

Menu Option	Function
Refresh	Refreshes Desktop display
Save Settings	Saves the model and set of plug-ins in the model file Note: When you load a new plug-in while the Desktop running, the Desktop saves the new plug-in to the model file so that you do not have to load it the next time you launch the Desktop.
Exit	Exits the Desktop

Table 22 Plug-ins menu options

Menu option	Function
Load new plug-in	Allows you to enter a class name for the new plug-in to be loaded
Service Manager	Currently loaded plugin
e-speak configuration plug-in	Currently loaded plugin
Element Manager	Currently loaded plugin
Core Manager	Currently loaded plugin
Configuration Manager	Currently loaded plugin
Process Manager	Currently loaded plugin
Template Manager	Currently loaded plugin

Table 23 Options menu options

Menu option	Function
New host	Creates a new host
Properties	Displays the properties of the currently-selected element
System settings	Displays the System Settings menu

Table 24 System Settings menu options

Menu option	Function
Make default connections	For the default group, an alternative to selecting System settings > Managed Connection Groups , selecting the Default group name, and clicking Connect
Manage Connection Groups	<p>Contains the following options:</p> <ul style="list-style-type: none"> • Add group — Create a new group by specifying the hosts that belong in the group <p>Note: To enable a group, right-click the group name and click set auto connect enabled. To disable the group, click set auto connect disabled.</p> <ul style="list-style-type: none"> • Save — Updates the selected group and allows you to redefine the set of hosts (overrides an existing group) • Connect — Forms a batch connection to every host in the selected group • Information — States which hosts are in the selected group • Delete — Deletes the selected group. <p>Note: This deletes the group from the connection groups' file. You cannot undo this option.</p>
Save Active Hosts as Default connections	Saves every active host to the default group

Table 25 Core browser tabs

Menu option	Function
Information	<p>Displays the URL of the service, the ID of the service in the core, and the service's current state as an e-speak resource</p> <p>For example, this tab indicates whether or not the service:</p> <ul style="list-style-type: none"> • Has been advertised through the advertising service • Is persistent • Is an internal service that the core uses or an external service accessible to clients
Vocabulary	Contains a table of name value pairs
Contract	Displays the service contract

Table 26 Process menu options

Menu option	Function
Delete	<p>Deletes the element from the Desktop's database</p> <p>Note: You cannot undo this option.</p>
Properties	Displays the element's properties
Add to templates	Adds templates of the selected process element
Process Management	Displays the Process Management menu
Configuration	Displays the Configuration menu

Table 27 Process Management menu options

Menu option	Function
Set sticky	Ensures that the process exists even if the Process Manager stops
Process output	Displays the process output window

Table 27 Process Management menu options

Menu option	Function
Find hosted services	Looks for manageable services running in the process Note: You would not normally use this option since the Desktop automatically detects processes.
Stop process	Stops an active or running process
Start process	Starts a process
Set non-persistent	Sets the process as non-persistent
Set persistent	Sets the process as persistent

Service Control Window Menu and Options

The Service Control window contains three types of components:

- Geometric shapes
- Transitions (square buttons)
- States (oval buttons)

Table 28 Component descriptions

Component	Description
Solid lines	Transitions you can explicitly make For example, from a service's running state, you can force a pause transition into a stopped state by clicking pause . To transition the managed counter service from a degraded state to a closed state, click the pause and then reset .
Dashed lines	Transitions handled by the service For example, if the counter service enters an error state and you click reset , the service enter a closed state.

Table 28 Component descriptions

Component	Description
Double arrows	Represents the ability to move back and forth between states For example, the double arrow line connecting the degraded and running states indicates that a service can go from a degraded state to a running state and vice versa.
Single arrows	Displays the direction of the transition For example, a service can go from degraded state to a stopped state, but not vice versa.
Boxes	Buttons that initiate service transitions
Ovals	Displays the service state name
Highlighted	Indicates the current state
Non- highlighted	Indicates non-active states

Table 29 Transition descriptions

Transitions	Descriptions
Reset	Resets a service that was once running but was closed
Remove	Shuts down a service
Init	Initializes or reinitializes the service (active state)
Pause	Pauses the service and its resources

Table 30 State descriptions

States	Descriptions
Closed	Service is shut down with no resources
Stopped	Service is suspended
Running	Service is active

Table 30 State descriptions

States	Descriptions
Error	There is a problem with the service that must be addressed
Degraded	Service has experienced a heavy load and is being compromised For example: <ul style="list-style-type: none"><li data-bbox="704 611 1179 642">• Several clients trying to use the service<li data-bbox="704 657 1445 720">• An administrator is running other services using a machine with limited resources

Chapter 7 Running Standard Services

E-speak includes three standard services.

Table 31 Standard services

Service	Description
Advertising Service	Locates services that are not present in a local logical machine
Management Services	A suite of services used to manage an e-speak application Note: The <i>E-speak System Management User Guide</i> provides detailed information on the four management services and how to use the Programming Interface to make an e-speak component manageable.
Process Manger	Remotely manages processes running on an e-speak-enabled machine

NOTE: The examples in this section use the Windows NT[®] notation for path names. On Unix platforms, use forward slashes (/) instead of back slashes (\).

The standard services are included in the `<installation_directory>/lib/es.jar` file.

Running the Advertising Service

The Advertising Service locates services that are not on a local machine. The Advertising Services communicate with one another, find services that satisfy a client's requirements, and return them to the client.

NOTE: For more details, please refer to the *e-speak Programmers Guide*.

Depending on how services are stored in backend service directories, an Advertising Service may run in one of three modes.

Table 32 Advertising service modes

Mode	Description
Online, without an external lookup directory	In this mode, also known as SLP mode, Advertising Services use multicast to find and establish connections with one another. Multicast is usually limited by subnet (LAN segment) boundary. Also, online-mode Advertising Services keep service advertisements in memory, which means they are not persistent and less scalable.
Offline, with an external lookup directory	You can use any lookup directory to implement the offline mode. The Advertising Service in this release uses a Lightweight Directory Access Protocol (LDAP) server as its external lookup directory. Offline mode provides persistence and better scalability.
Using a proxy service to access the actual backend directory service	This mode, with a proxy backend service, allows access control to the actual service directory (for example, the advertising proxy of HP sponsored global service directory (ESV)).

Working in Offline Mode

The Advertising Service runs in offline mode using an LDAP server as its external lookup directory. For more information about LDAP, refer to the LDAP World website at <http://www.critical-angle.com/ldapworld/index.html>.

NOTE: The e-speak platform has been tested with the Netscape LDAP server.

Installing a Directory Server

To download and configure a directory server to use with the Advertising Service:

- 1** Obtain a directory server program. You can download a trial version from Netscape at http://www.iplanet.com/downloads/testdrive/detail_8_7.html.
- 2** Ensure that you have the documentation you need, such as an installation guide, to properly install the directory server. If you download the directory server from Netscape, you can download the documentation from:

<http://developer.netscape.com/docs/manuals/index.html>

<http://home.netscape.com/eng/server/directory/4.0>

- 3** Make a note of the following values as they pertain to your installation. You need this information to complete the configuration process:

- Directory suffix (this is the organization setting)
- Directory Manager Distinguished Name (DN)
- Directory Manager password

NOTE: The installation process requires a large amount of configuration data. Except for the values noted in step 3, you can use the default values in most cases.

- 4** Install the LDAP server on your platform. If you are installing on Windows NT®, you must start the system to start the directory server. Refer to the instructions as appropriate for the HP-UX and Linux platforms.
- 5** Create a schema to store information about the connection object. Refer to the *Netscape Directory Server: Administrator's Guide* for instructions on creating object classes and attributes.

Create an Attribute for Connection Objects

In the Directory Server Console:

- 1** Select the **Configuration tab > Database icon > Schema folder > Attributes** tab.
- 2** Click **Create**.
- 3** Enter `cobj` as the Attribute Name.
- 4** Select `binary` as the syntax.
- 5** Click **OK**.

Create an Object Class

In the Directory Server Console:

- 1** Select **Configuration tab > Database icon > Schema** folder.
- 2** Click **Create**.
- 3** Enter `ESCore` as the Name.
- 4** Add the following attributes:
 - `cn`
 - `cobj`
 - `objectclass`
- 5** Click **OK**.

Download the LDAP JDK

An LDAP JDK provides a set of APIs to access an LDAP directory server. You can download a free version of the LDAP JDK from Netscape. The latest version is Netscape Directory SDK 3.1 for Java™, which is downloadable from:

<http://developer.netscape.com/tech/directory/index.html>

Starting the Advertising Service

Before you start `net.espeak.services.advService.server.AdvService`, ensure that the e-speak core is running. Refer to the *e-speak Programmers Guide* for information on J-ESI communities and to determine what kind of configuration is right for your choice. Also refer to [“Configuring the Advertising Service”](#) for detailed descriptions on the configurable properties.

Once you determine the property values, determine the best method to pass them to the Advertising Service.

TIP: If you are using an earlier version of e-speak, you can start the Advertising Service with the old interface using:

```
net.espeak.services.advertise.ypserver.AdvertisingService
```

Command line switches like `-group` are still supported in this release, but are not recommended. Also, you cannot configure new properties like `net.espeak.services.advService.mcast.sleepMillis` when using the old interface.

How the Advertising Service reads Configurable Properties

The Advertising Service reads properties from these locations in this order:

- 1** The command line
- 2** The Advertising Service configuration file, if any, specified by the property `net.espeak.services.advService.config` in the command line
- 3** The e-speak configuration file `espeak.cfg`

If the same property is defined in different places, the value defined in the command line overrides the one defined in the advertising configuration file, which overrides the one defined in `espeak.cfg`.

You may also collect LDAP-related property definitions in an LDAP configuration file and specify the file path for property `net.espeak.services.advService.backend.ldap.config`. The Advertising Service uses this file if you do not define values in the three places listed above. Similarly, you can have an ESVProxy configuration file for ESV proxy properties. The services uses the static default values presented in [“Configuring the Advertising Service”](#) if the property is not defined in the command line, Advertising Service configuration file, or `espeak.cfg`.

Defining Properties in the Command Line

Properties defined in Advertising Service command line take precedence over all other properties. The command line syntax is:

```
<property_name>=<property_value>
```

For example:

```
hostname=localhost portnumber=12345
net.espeak.services.advService.config=adv.cfg
```

NOTE: 12345 indicates an arbitrary port.

Guidelines for Starting the Advertising Services

When starting the Advertising Service:

- In the command line, always specify host name and port number to which the core connects.
- Unless you are using nearly all of the default values, use an Advertising Service configuration file. Use `espeak.cfg` for other non-Advertising Service-specific settings such as security and web proxy.
- If you are creating multiple groups of Advertising Services that share the same backend settings, specify a group in the command line and share other settings with an Advertising Service configuration file.
- If you are creating multiple groups of Advertising Services with *different* backend settings, create different Advertising Service configuration files for each group. Alternatively, consider using LDAP or ESV Proxy configuration files.

Examples

- Start one Advertising Service using all of the default settings:
 - Command line: <empty>
 - Advertising Service configuration file: <not used>

NOTE: The Advertising Service is connected to the default core
localhost:2950.

- **Starting two Advertising Services of the same group in online (SLP) mode:**

```
Command line for adv 1: hostname=localhost portnumber=12345\  
net.espeak.services.advService.config=adv.cfg
```

```
Command line for adv 2: hostname=host2.hp.com portnumber=2950\  
net.espeak.services.advService.config=adv.cfg
```

Common Advertising service configuration file (adv.cfg):

```
net.espeak.services.advService.group=myGroup  
net.espeak.services.advService.backend.protocol=slp  
net.espeak.services.advService.mcast.port=1428  
net.espeak.services.advService.mcast.sleepMillis=7000
```

NOTE: 12345 indicates an arbitrary port.

- **Starting Advertising Services of the two different groups in using the same LDAP server:**

```
Command line for adv 1: hostname=localhost portnumber=12345\  
net.espeak.services.advService.group=myGroup1 \  
net.espeak.services.advService.config=adv.cfg
```

```
Command line for adv 2: hostname=host2.hp.com portnumber=2950\  
net.espeak.services.advService.group=myGroup1\  
net.espeak.services.advService.config=adv.cfg
```

```
Command line for adv 3: hostname=localhost portnumber=12345\  
net.espeak.services.advService.group=myGroup2\  
net.espeak.services.advService.config=adv.cfg
```

```
Command line for adv 4: hostname=host4.hp.com portnumber=2950\  
net.espeak.services.advService.group=myGroup2\  
net.espeak.services.advService.config=adv.cfg
```

Common Advertising service configuration file (adv.cfg):

```
net.espeak.services.advService.backend.protocol=ldap  
net.espeak.services.advService.mcast.sleepMillis=0  
net.espeak.services.advService.backend.ldap.host=myLdap.hp.com  
net.espeak.services.advService.backend.ldap.port=389  
net.espeak.services.advService.backend.ldap.rootDN=myDirMgr  
net.espeak.services.advService.backend.ldap.passwd=myPassword  
net.espeak.services.advService.backend.ldap.organization=hp.com  
net.espeak.services.advService.backend.ldap.proxyHost=myWeb-\  
proxy.hp.com  
net.espeak.services.advService.backend.ldap.proxyPort=8088
```

NOTE: 12345 indicates an arbitrary port. adv 1 and adv 3 are running on different machines; localhost:12345 refers to different cores.

Process Manager

The e-speak Process Manager service is a management agent that remotely manages processes running on an e-speak-enabled machine.

NOTE: The Process Manager should be running on each e-speak-enabled machine.

This service allows the system administrator to remotely start, stop, and monitor the state and output of processes. You can launch processes started in the Process Manager using a command line string. For example, to start the e-speak core, type:

```
java net.espeak.infra.core.startup.StartESCore
```

The Process Manager does not manage e-speak services; e-speak services may be launched in an OS process (for example, a Java™ program that creates an e-speak service or a shell script that launches several Java™ programs).

Since the Process Manager is the management access point for e-speak-enabled machines, it is used to collate information about the processes being managed and handles the registration of services running in those processes. Many other e-speak management applications use these functions.

The e-speak installation includes an executable Process Manager that launches the service as the machine boots. You do not need to explicitly start the service from the command line.

A core and an Advertising Service are launch when the Process Manager starts. These management-specific components allow the Process Manager to be discovered by remote system administration applications.

The Process Manager provides a uniform method of remotely-managing processes. The Process Manager starts and manages e-services and cores on its local host. The Desktop and Process Manager use the SDK to communicate with each other.

NOTE: A process is a general operating system process such as a running Java Virtual Machine™, an active e-speak core, or a database server.

Starting the Process Manager

Once the Process Manager is installed, you can delegate authorizations to enable management with remote machines. Management agents such as the XAMService use the core in the Process Manager VM to communicate with clients.

On Windows NT®, the Process Manager runs as a service that does not interact with the Desktop. Because of this, any process that the Process Manager starts can not display a window. You can modify this setting by changing the `ProcessManagerService` property in the Windows NT® Control Panel. Refer to the [“Working with the Desktop”](#) in Chapter 10, [“Troubleshooting”](#) for details.

The operating system starts the default Process Manager. You can configure the behavior of this Process Manager using the management properties specified in `espeak.cfg`. Refer to [“Basic Concepts”](#) for the details on these properties.

To start the Process Manager from the command line, type:

```
java CLASSNAME= -D espeak_home=<espeak home directory>
net.espeak.services.management.processmanager.
ProcessManager
```

where `espeak_home` is the e-speak installation directory. This command starts the Process Manager, core, and Advertising Service using the default values.

The process manager command also supports five options.

Table 33 Process Manager command line options

Option	Description
<code>-Config <Config File Name></code>	The configuration file overwrites the default values in <code>espeak.cfg</code>
<code>-outputDir <Directory Name></code>	The specified <code><Directory Name></code> overwrites the default value or the value specified for the <code>net.espeak.services.management.processmanager.OutputDir</code> property in <code>espeak.cfg</code>
<code>-noAdv</code>	The specified value overwrites the default value for the <code>net.espeak.services.management.processmanager.NoAdvertisingService</code> property in <code>espeak.cfg</code>

Table 33 Process Manager command line options

Option	Description
<code>-instanceName <InstanceNameValue></code>	The specified value overwrites the default value for the <code>net.espeak.services.management.processmanager.InstanceName</code> property in <code>espeak.cfg</code>
<code>-persistenceList <Persistence List File></code>	The specified value overwrites the default value for the <code>net.espeak.services.management.processmanager.PersistenceListFile</code> property in <code>espeak.cfg</code>

NOTE: The Process Manager has to be started in interactive mode on the NT platform in order to view the applet and other examples, such as calculator.

Chapter 8 Expanding Functionality

E-speak has many advanced features that you configure separately from the core and standard services installation:

- **Gateway Usage** — Activates e-speak over the Internet
- **Security** — Establishes a secure communication environment
- **Integrated Development Environments** — Establishes complex e-speak environments
- **Firewall Gateway** — Mediates access between external clients and e-speak services that reside behind a corporate firewall

Gateway Usage

E-speak Gateway Usage lets you interact with the e-speak core and services through a standard web-based document exchange model. Internally, Gateway Usage uses XML and provides an interface for XML-enabled applications.

It currently provides a subset of e-speak's core functionality for both XML mappings and browser access. Interaction between the browser, XML clients, and Gateway Usage is based on the HTTP protocol.

Implementation Restrictions

Some restrictions apply to the current implementation:

- You must install Gateway Usage services as servlets.
- Gateway Usage has been tested with Apache1.3.12 / Jserv1.1 and Tomcat 3.1
- Only a single e-speak core is currently supported

Features

The e-speak Gateway Usage interface supports all the critical features of e-speak using a document-based interface:

- Vocabulary definition
- Service registration
- Service lookup
- Service invocation (synchronous interactions only)
- Client login session handling
- User account creation and management
- Service URL lookup (HTTP post)

This section describes Apache/Jserv and Tomcat installation and setup.

Installation

The e-speak Gateway Usage interface runs as a servlet within a Java-enabled web server. Gateway Usage requires the packages listed in [Table 34](#).

Table 34 Gateway Usage required packages

Package	Available From
E-speak core	Included in the e-speak release
Apache Web Server version 1.3.12	http://www.apache.org/dist/binaries/
Apache/JServ servlet extension for Apache version 1.1	http://java.apache.org/jserv/dist
Tomcat version 3.1	http://jakarta.apache.org/site/binindex.html
JSDK20, Java™ Servlet Developer's Kit	http://www.sun.com

Table 34 Gateway Usage required packages

Package	Available From
Xerces XML parser	Included in the e-speak release
Webmacro template package	Included in the e-speak release

Conventions

This section follows these conventions:

- `<installation_directory>` is the e-speak installation directory
- `<APACHE>` is the Apache installation directory
- `<JSERV>` is the Apache JServ installation directory
- `<JDK>` is the JDK installation directory
- `<JSDK>` is the Java™ Servlet Development Kit installation directory
- `<TOMCAT>` is the Tomcat installation directory

Changes in Webmacro.properties

1 Open the file:

```
<installation_directory>/extern/webmacro/Webmacro.properties
```

2 Change the `TemplatePath` to point to:

```
<installation_directory>/config/htdocs/templates
```

For example, for Windows NT, the line should be `<installation_directory>/config/htdocs/`. Substitute the actual installation directory path for `<installation_directory>`. The template files are used for internally generating XML messages and for HTML page templates.

Changes in WebAccess.xml

1 Open the configuration file:

```
<installation_directory>\extern\webmacro\webaccess.xml
```

This file contains the settings used to set the password, expiration time, hostname, web proxy name, and database connection information. You can also use it to enable or disable the persistent message queue and debugging.

- 2 Locate `<TokenManager>`. Inside this is a `<TokenPassword>`, which is a key used to encrypt the session tokens given to clients.
- 3 Change the `<TokenPassword>` as needed.

NOTE: To ensure security, each site should have a unique `<TokenPassword>`.
- 4 Locate `<ExpireIntervalSecs>` and make any necessary modifications. This element sets the expiration interval for each token. The default setting is 14400 seconds.
- 5 Update the information for JDBC connection in the configuration file in:
 - The name of the host running the Oracle database
 - The user name and password
- 6 Locate `<webServerInfo>`.
- 7 Set the `hostname` to your hostname.
- 8 Locate `<webProxyInfo>`.
- 9 Set the `hostname` to your proxy.

Setting the Shell Environment for Unix and Linux™ Platforms

Table 35 Set the shell environment

If you are using...	Set the shell environment to...
HP-UX™	<code>SHLIB_PATH=<installation_directory>/lib</code>
Linux™	<code>LD_LIBRARY_PATH=<installation_directory>/lib</code>

Installing Apache Web Server on Windows NT®

You can use Apache Web Server with JServ, Tomcat, or both.

NOTE: If you are using only Tomcat, refer to [“Installing Tomcat”](#) for instructions.

- 1 Download `apache_1_3_12_win32.exe` from www.apache.org.
- 2 Follow the installation procedure to install Apache.
- 3 Open the `<APACHE>/conf/httpd.conf` file.

4 Modify the DocumentRoot and Directory values:

```
DocumentRoot "<installation_directory>/config/htdocs"  
...  
<Directory "<installation_directory>/config/htdocs">  
...  
Order allow,deny  
Allow from all  
</Directory>
```

You may also need to specify the `ServerName` if you are running on NT® or are using a server cluster. For more information, see the Apache server documentation.

NOTE: You can find sample configuration files for Apache and JServ in `<installation_directory>/config/htdocs`.

Installing Apache JServ on Windows NT®

1 Download `ApacheJServ-1.1.exe` from <http://java.apache.org>.

2 Follow the installation procedure to set up JServ.

3 Enter the location of the following components during setup, when prompted:

- JDK
- Apache and its `httpd.conf`
- JSDK

4 Select **Yes** when the JServ installation prompts you to change Apache's `httpd.conf`.

5 Check the Apache `httpd.conf` file to ensure that the line for Apache JServ configuration is present. If not, enter it manually or correct it if necessary:

```
Include "<JSERV>/conf/jserv.conf"
```

6 Modify the following JServ configuration files for e-speak Gateway Usage after Apache JServ installation is complete:

- `<JSERV>\conf\jserv.properties`
- `<JSERV>\servlets\zone.properties`
- `<JSERV>\conf\jserv.conf` (if you want to enable logging)

The following procedures describe how to make the required changes to the two configuration files.

Changes in jserv.properties

- 1 Locate these sections in the `jserv.properties` file and ensure that the paths are valid:

```
# The Java Virtual Machine interpreter.
wrapper.bin=<JDK>\bin\java.exe
# CLASSPATH environment value passed to the JVM
wrapper.classpath=<JSERV>\ApacheJServ.jar
wrapper.classpath=<JSDK>\lib\jsdk.jar
```

- 2 Add the following lines, as appropriate, for the path and classpath:

```
# PATH environment value passed to the JVM (to be added)
wrapper.path=<installation_directory>\lib
# Additional CLASSPATH to be added
wrapper.classpath=<installation_directory>\lib
wrapper.classpath=<installation_directory>\lib\es.jar
wrapper.classpath=<installation_directory>\extern\webmacro\
webmacro.jar
wrapper.classpath=<installation_directory>\extern\webmacro
wrapper.classpath=<installation_directory>\extern\parser\xerces.jar
wrapper.classpath=<installation_directory>\extern\cryptix\cryptix32.jar
wrapper.classpath=<installation_directory>\extern\cryptix
wrapper.classpath=<installation_directory>\extern\ldap\ldapjdk.jar
wrapper.classpath=<installation_directory>\extern\oracle-
lib\816\classes12.zip
```

If you are using Linux™, add this line to the properties text:

```
wrapper.env=LD_LIBRARY_PATH=<installation_directory>/lib
```

Changes in zone.properties

- 1 Locate the `zone.properties` file in the `<JSERV>\servlets` directory.
- 2 Add the following lines as appropriate:

```
# Startup Servlets
servlets.startup=net.espeak.webaccess.WebAccess

# Servlet Aliases
servlet.WebAccess.code=net.espeak.webaccess.WebAccess
servlet.IsItWorking.code=net.espeak.webaccess.htdocs.servlets.
    IsItWorking
servlet.ServiceTest.code=net.espeak.webaccess.ServiceTest
```

When Apache starts, you can access the servlets at `http://<your-machine>/servlets/WebAccess`.

Debug Logging Changes When Using Apache/Jserv

Because Apache servlets do not report messages to the screen, you obtain debug messages generated from servlets from log files. For example:

```
tail -f <JSERV>/logs/jserv.log
tail -f <APACHE>/logs/access.log
```

Setting the debug option for the servlet allows you to trace requests as they pass through the Gateway Usage servlet. This setting is located in the `zone.properties` file. The logging level should be lowered to `info` in the `jserv.properties` file.

To set the debug option:

- 1 Add this line for the Servlet Init Parameters to the `zone.properties` file:

```
servlet.net.espeak.webaccess.WebAccess.initArgs=debug=1
```

- 2 Make these changes to the `jserv.properties` file:

```
log.channel=true
log.channel.info=true
log.channel.debug=true
```

- 3 Make this change to the `jserv.conf` file:

```
ApJServLogLevel info
```

Installing Apache Web Server and JServ on HP-UX™ and Linux™

Refer to “[Installing Apache Web Server on Windows NT®](#)” and “[Installing Apache JServ on Windows NT®](#)”. Use the appropriate download files.

Testing Gateway Usage Configuration

After the components are installed and configured, you can test the Gateway Usage infrastructure.

Testing Gateway Usage Configuration When Using Apache/Jserv

NOTE: Before you begin this procedure, stop all JVMs.

1 Start the core:

```
cd <installation_directory>\bin
.\espeak
```

2 Start the Apache server from the `<installation_directory>\config` directory. For example:

```
cd <installation_directory>\config
"<APACHE>\Apache.exe" -d "<APACHE>" -s
```

NOTE: Use double quotes only if your `<APACHE>` has blanks in it. If you use a Windows NT® shortcut, modify the icons Start In property to `<installation_directory>\config`. You can also start the Apache server by copying these files from the `<installation_directory>/config` directory to `<APACHE>` directory: `espeak.cfg`, `securestore.bin`, `clientcerts.adr`, and `servicecerts.adr`. You can then start Apache the normal way.

3 Launch a web browser, such as Netscape Navigator or Microsoft Internet Explorer.

4 Point the browser to `http://localhost`. Depending on your configuration, you may need to use your actual hostname instead of localhost if the Login screen does not display.

The browser window displays the login window.

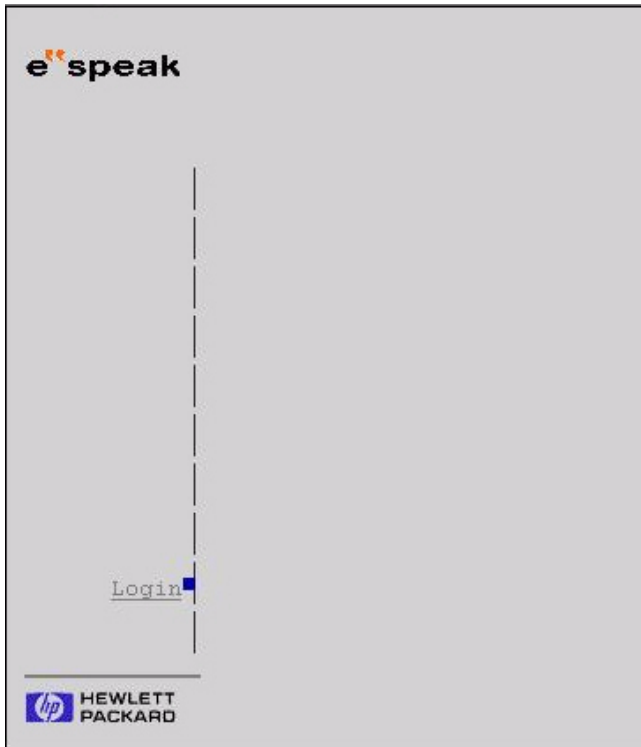


Figure 31 Login window

If you do not see this window, ensure that your:

- Servlet installation is working properly (even without e-speak)
- Apache server is running properly
- Apache server is pointed to the WebAccess `htdocs` directory

5 Click the **Login** link.

6 Enter `esadmin` as both the username and password.

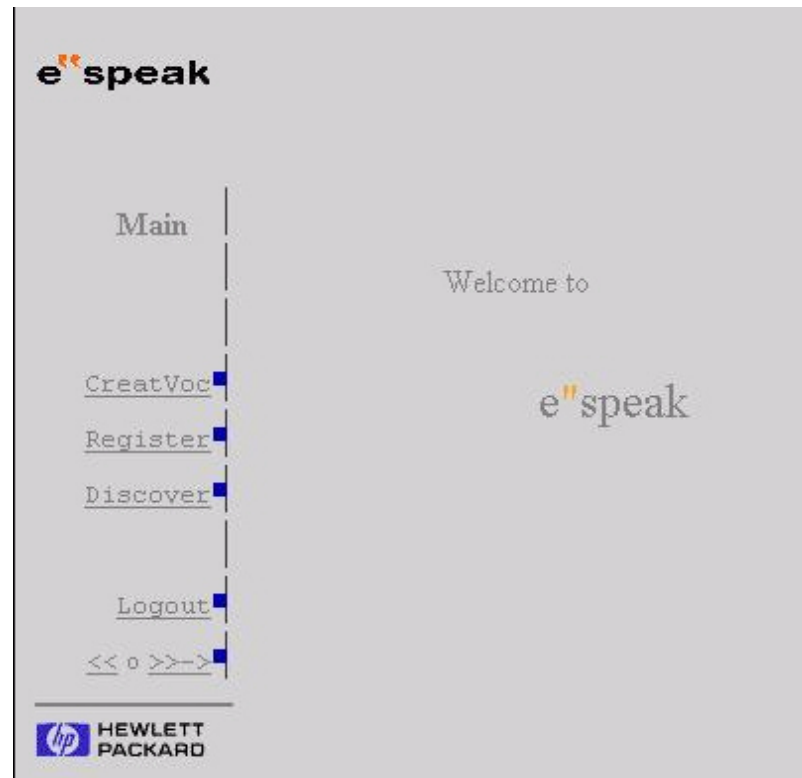
7 Click Login.

Figure 32 Successful login

From this window, you can:

- Register and locate vocabularies and services
- Register users

Troubleshooting Gateway Usage with Apache/Jserv

- Ensure that the configuration files do not have any typing errors or case differences.
- Before you start the core, ensure that no JVMs are running.

- Review the logs under *<APACHE>* and *<JSERV>*:
`<APACHE>/logs/access.log`
`<APACHE>/logs/error.log`
`<JSERV>/logs/jserv.log`
`<JSERV>/logs/mod_jserv.log`
- Most problems in using Gateway Usage are related to faulty configurations in `webaccess.xml`. Make sure you have read the installation and configuration notes on this file.
- Make sure your `webaccess.xml` file is in a directory in the `CLASSPATH`.
- Make sure site-specific plug-ins implement the right interfaces and have default public constructors.
- Make sure the web server is set up to find the `webaccess` directory.
- Make sure the `webmacro.properties` file has been changed to refer to the location of the
`<installation_directory>\webaccess\htdocs\templates` directory.
- Make sure your proxy server is configured correctly so that the `java.net.URL` class can connect to your service.
- Make sure to configure the database connection information for the persistent message manager in `webaccess.xml`.
- Make sure the web server is started in the
`<installation_directory>\config` directory. This directory contains an `espeak.cfg` file that Gateway Usage reads to set up security.

Installing Tomcat

Tomcat requires changes to these configuration files:

- `server.xml` (`<TOMCAT>/conf/server.xml`)
- `web.xml` (`<your_webapp_root>/web-inf/web.xml`)

NOTE: `<your_webapp_root>` is the root directory of your web application. For **e-speak**, this is `<installation_directory>/config/htdocs`.

- `tomcat.bat` or `tomcat.sh` (in `<TOMCAT>/bin/`)

You can modify and use these files or use the sample files included with the release (located in the `<installation_directory>/config/htdocs` directory). If you use the sample files provided, very minimal changes are necessary. The instructions in “[Installing Tomcat on Windows NT®](#)” assume that you are using the sample files.

Installing Tomcat on Windows NT®

NOTE: You must have Win32 tools, such as WinZip™, to extract the downloaded zip file.

To install Tomcat:

- 1** Download `jakarta-tomcat.zip` from <http://jakarta.apache.org>.
- 2** Unzip the file into a directory. The installation creates a new subdirectory `jakarta-tomcat` within the directory.

NOTE: For best results, do not include spaces in the directory name.

- 3** Rename `jakarta-tomcat` to `tomcat`.
- 4** Set a new system environment variable `TOMCAT_HOME` to point to the root directory of your Tomcat hierarchy. For example:

```
TOMCAT_HOME=c:\foo\tomcat
```

- 5** Set the system environment variable `JAVA_HOME` to point to the root directory of your JDK hierarchy. Tomcat requires JDK 1.2.2 or later version. For example, if your JDK 1.2.2 is installed under `c:\pctools\jdk122\`, set your `JAVA_HOME` variable to:

```
JAVA_HOME=c:\pctools\jdk122
```

- 6** Modify the `PATH` environment variable to include the e-speak libraries and webmacro files:

```
set
  PATH=%PATH%;<installation_directory>\lib;<installation_directory>\
  extern\webmacro
```

- 7** Modify the `CLASSPATH` to include all the jar files in the e-speak release:

```
CLASSPATH=.;
  <installation_directory>\lib;
  <installation_directory>\extern\cryptix;
  <installation_directory>\extern\cryptix\cryptix32.jar;
  <installation_directory>\extern\parser\xerces.jar;
  d:\pctools\javatools\jsdk20\lib\jsdk.jar;
  <installation_directory>\extern\webmacro;
```

```

<installation_directory>\extern\webmacro\webmacro.jar;
d:\e-speak\extern\ldap\ldapjdk.jar;
<installation_directory>\extern\oracle-lib\816\classes12.zip

```

8 Modify `<installation_directory>\config\htdocs\tomcat.bat` so that `<TOMCAT>\lib\xml.jar` follows the e-speak CLASSPATH and starts with the provided server.xml file. For example:

```

rem Set up the CLASSPATH that we need; save your existing one

set cp=%CLASSPATH%

rem First, prepend TOMCAT jar files
set CLASSPATH=.
set CLASSPATH=%CLASSPATH%;%TOMCAT_HOME%\classes
set CLASSPATH=%CLASSPATH%;%TOMCAT_HOME%\lib\webserver.jar
set CLASSPATH=%CLASSPATH%;%TOMCAT_HOME%\lib\jasper.jar
set CLASSPATH=%CLASSPATH%;%TOMCAT_HOME%\lib\servlet.jar

rem Add e-speak xml parser xerces.jar and then TOMCAT xml.jar
rem %ESPEAK_HOME% should be your e-speak installation directory.

set CLASSPATH=%CLASSPATH%;%ESPEAK_HOME%\extern\parser\xerces.jar
set CLASSPATH=%CLASSPATH%;%TOMCAT_HOME%\lib\xml.jar

set CLASSPATH=%CLASSPATH%;%JAVA_HOME%\lib\tools.jar
set CLASSPATH=%CLASSPATH%;%JAVA_HOME%\jre\lib\jre.jar

rem append e-speak libraries
rem ESPEAK_HOME is your e-speak installation directory

set CLASSPATH=%CLASSPATH%;%ESPEAK_HOME%\lib\es.jar
set CLASSPATH=%CLASSPATH%;%ESPEAK_HOME%\extern\cryptix
set CLASSPATH=%CLASSPATH%;%ESPEAK_HOME%\extern\cryptix\cryptix32.jar
set CLASSPATH=%CLASSPATH%;%ESPEAK_HOME%\extern\webmacro
set CLASSPATH=%CLASSPATH%;%ESPEAK_HOME%\extern\webmacro\webmacro.jar
set CLASSPATH=%CLASSPATH%;%ESPEAK_HOME%\extern\ldap\ldap.jar

```

```

set CLASSPATH=%CLASSPATH%;%ESPEAK_HOME%\extern\oracle-lib\
  816\classes12.zip

set CLASSPATH=%CLASSPATH%;%ESPEAK_HOME%\extern\rhino\js.jar

set CLASSPATH=%CLASSPATH%;%ESPEAK_HOME%\extern\parser\xalan.jar

rem either JSDK_HOME should be set or give an absolute path here
rem JSDK is not included with the release, you must install it
set CLASSPATH=%CLASSPATH%;%JSDK_HOME%\lib\jsdk.jar
if "%cp%" == "" goto next
rem else
set CLASSPATH=%CLASSPATH%;%cp%

:startServer

echo Starting tomcat in new window
echo Using classpath: %CLASSPATH%

start java %TOMCAT_OPTS% -Dtomcat.home="%TOMCAT_HOME%"
  org.apache.tomcat.startup.Tomcat -f
  %ESPEAK_HOME%\config\htdocs\conf\server.xml

:runServer

rem Start the Tomcat Server
echo Using classpath: %CLASSPATH%

java %TOMCAT_OPTS% -Dtomcat.home="%TOMCAT_HOME%"
  org.apache.tomcat.startup.Tomcat -f
  %ESPEAK_HOME%\config\htdocs\conf\server.xml %2 %3 %4 %5 %6 %7 %8 %9

```

NOTE: The sample tomcat.bat file is in
<installation_directory>\config\htdocs.

- 9** Ensure that the *<installation_directory>\htdocs\conf\server.xml* file includes the ContextManager and the context for e-speak Gateway Usage. Replace *<installation_directory>* with your e-speak installation directory:

```

<ContextManager debug="0" workDir="work"
  home="<installation_directory>/config/htdocs">
...
<!-- 'servlets' for e-speak. -->
<Context path="/servlets" docBase="<installation_directory>/config/
  htdocs" debug="9"
reloadable="false" >

```

```
</Context>

<!-- used for browser access: http://localhost:8080 -->

<Context path="" docBase="<installation_directory>/config/htdocs"
  debug="9"
  reloadable="false" >
</Context>
```

Verifying the Tomcat Installation

To verify the Tomcat installation, start and stop Tomcat from the `<installation_directory>\config` directory using the scripts provided with the installation.

Starting Tomcat

To start Tomcat:

1 Navigate to `<installation_directory>\config`.

2 Enter the start or run command:

```
htdocs\tomcat start
```

or

```
htdocs\tomcat run
```

You will see the message `Starting tomcat` and the classpath. Disregard the `File not found` message for the `tomcat_users.xml` file or the `No webapps` directory message.

Stopping Tomcat

To stop Tomcat, enter the `stop` command from the `<installation_directory>\config` directory:

```
htdocs\tomcat stop
```

You will see the message `Stop Tomcat` and the classpath.

Installing Tomcat on HP-UX™ and Linux™

To install Tomcat:

1 Download `jakarta-tomcat.tar.Z` (tar archive) or `jakarta-tomcat.tar.gz` (GZIP compressed) from <http://jakarta.apache.org>.

- 2 Uncompress or expand the file and untar it into a directory.

NOTE: For best results, do not include spaces in the directory name.

The installation creates a new subdirectory called `jakarta-tomcat`.

- 3 Rename `jakarta-tomcat` to `tomcat`.

- 4 Set a new environment variable `TOMCAT_HOME` to point to the root directory of the Tomcat hierarchy in your `.profile` file (HP-UX™) or `.bashrc` file (Linux™). For example, for `bash/sh`, the commands are:

```
TOMCAT_HOME=/usr/local/tomcat; export TOMCAT_HOME
```

- 5 Set the environment variable `JAVA_HOME` to point to the root directory of the JDK hierarchy in your `.profile` file (HP-UX™) or `.bashrc` file (Linux™). Tomcat requires JDK 1.2.2 or a later version. For example, if your JDK 1.2.2 is installed under `/mnt/tools/jdk/122`, set your `JAVA_HOME` variable as:

```
export JAVA_HOME=/mnt/tools/jdk/122 or
export JAVA_HOME=$ESPEAK_JHOME
```

- 6 Set the `SHLIB_PATH` variable and/or `LD_LIBRARY_PATH` to include e-speak libraries:

```
export SHLIB_PATH=<installation_directory>/lib
```

- 7 Set the `LD_LIBRARY_PATH` variable only if you are using Linux™:

```
export LD_LIBRARY_PATH=<installation_directory>/lib
```

- 8 Modify the `PATH` environment variable to include e-speak libraries and webmacro files:

```
export PATH=$PATH:$ESPEAK_HOME/lib:$ESPEAK_HOME/extern/webmacro
```

- 9 Modify the `CLASSPATH` in the `.profile` file (HP-UX™) or `.bashrc` file (Linux™) to include all `.jar` files in the e-speak release. For example:

```
export J_HOME=/mnt/tools/jdk/122/HP-UX
export JAVAC=/mnt/tools/jdk/122/HP-UX/bin/javac
export JRE=/mnt/tools/jdk/122/HP-UX/jre/bin/java
export JAVA_HOME=/mnt/tools/jdk/122/HP-UX
export TOMCAT_HOME=/usr/local/tomcat
export ESPEAK_HOME=/opt/e-speak
export JSJK_HOME=/mnt/tools/extern/jsdk20
export SHLIB_PATH=/opt/e-speak/lib
```

```

export PATH=$PATH:/mnt/tools/jdk/122/HP-UX/bin:/opt/e-speak/bin:/opt/
e-speak/lib
export PATH=$PATH:/opt/e-speak/config
export PATH=$PATH:/opt/e-speak/extern/webmacro
export CLASSPATH=.
export CLASSPATH=$CLASSPATH:/opt/e-speak/lib
export CLASSPATH=$CLASSPATH:/opt/e-speak/lib/es.jar
export CLASSPATH=$CLASSPATH:/opt/e-speak/extern/cryptix
export CLASSPATH=$CLASSPATH:/opt/e-speak/extern/cryptix/
cryptix32.jar:
export CLASSPATH=$CLASSPATH:/opt/e-speak/extern/webmacro
export CLASSPATH=$CLASSPATH:/opt/e-speak/extern/webmacro/webmacro.jar
export CLASSPATH=$CLASSPATH:/opt/e-speak/extern/parser/xerces.jar
export CLASSPATH=$CLASSPATH:/opt/e-speak/extern/oracle-lib/816/
classes12.zip
export CLASSPATH=$CLASSPATH:/opt/e-speak/extern/ldap/ldapjdk.jar
export CLASSPATH=$CLASSPATH:/opt/e-speak/extern/rhino/js.jar
export CLASSPATH=$CLASSPATH:$JSDK_HOME/lib/jsdk.jar

```

10 Modify the `<installation_directory>/config/htdocs/tomcat.sh` file so that the `<TOMCAT>/lib/xml.jar` follows the e-speak CLASSPATH and starts with the provided `server.xml` file. You may not need to make any changes. For example:

```

oldCP=$CLASSPATH

# Add e-speak libraries and external jar files included in the release
# The order of CLASSPATH is changed for e-speak.
# ${ESPEAK_HOME} should be your e-speak installation directory.
#
# First, prepend TOMCAT jar files
CLASSPATH=.
# Backdoor classpath setting for development purposes when all classes
# are compiled into a /classes dir and are not yet jarred.
if [ -d ${TOMCAT_HOME}/classes ]; then
    CLASSPATH=${CLASSPATH}:${TOMCAT_HOME}/classes
fi
CLASSPATH=${CLASSPATH}:${TOMCAT_HOME}/lib/webserver.jar

```

```
CLASSPATH=${CLASSPATH}:${TOMCAT_HOME}/lib/jasper.jar
CLASSPATH=${CLASSPATH}:${TOMCAT_HOME}/lib/servlet.jar

# Add e-speak xml parser xerces.jar and then TOMCAT xml.jar
CLASSPATH=${CLASSPATH}:${ESPEAK_HOME}/extern/parser/xerces.jar
CLASSPATH=${CLASSPATH}:${TOMCAT_HOME}/lib/xml.jar

# Add jar files included with e-speak release
CLASSPATH=${CLASSPATH}:${ESPEAK_HOME}/lib/es.jar
CLASSPATH=${CLASSPATH}:${ESPEAK_HOME}/extern/cryptix
CLASSPATH=${CLASSPATH}:${ESPEAK_HOME}/extern/cryptix/cryptix32.jar
CLASSPATH=${CLASSPATH}:${ESPEAK_HOME}/extern/webmacro
CLASSPATH=${CLASSPATH}:${ESPEAK_HOME}/extern/webmacro/webmacro.jar
CLASSPATH=${CLASSPATH}:${ESPEAK_HOME}/extern/ldap/ldap.jar
CLASSPATH=${CLASSPATH}:${ESPEAK_HOME}/extern/oracle-lib/816/
  classes12.zip
CLASSPATH=${CLASSPATH}:${ESPEAK_HOME}/extern/rhino/js.jar
CLASSPATH=${CLASSPATH}:${ESPEAK_HOME}/extern/parser/xalan.jar
if [ -f ${JAVA_HOME}/lib/tools.jar ] ; then
    # We are probably in a JDK1.2 environment
    CLASSPATH=${CLASSPATH}:${JAVA_HOME}/lib/tools.jar
fi
if [ -f ${JAVA_HOME}/lib/tools.jar ] ; then
    CLASSPATH=${CLASSPATH}:${JAVA_HOME}/lib/jre.jar
fi
# Either JSDK_HOME should be set or given an absolute path here
# JSDK is not included with the release, you must install it

CLASSPATH=${CLASSPATH}:${JSDK_HOME}/lib/jsdk.jar
if [ "$oldCP" != "" ]; then
    CLASSPATH=${CLASSPATH}:${oldCP}
fi
export CLASSPATH

# We start the server up in the background for a couple of reasons:
```

```
#
# 2) You should use `stop` option instead of ^C to bring down the server
if [ "$1" = "start" ] ; then
    shift
    echo Using classpath: ${CLASSPATH}
    $JAVACMD $TOMCAT_OPTS -Dtomcat.home=${TOMCAT_HOME}
    org.apache.tomcat.startup.Tomcat -f ${ESPEAK_HOME}/config/htdocs/
    conf/server.xml "$@" &
```

NOTE: The sample `tomcat.sh` file is in
`<installation_directory>\config\htdocs.`

11 Ensure that the `<installation_directory>\htdocs\conf\server.xml` file includes the ContextManager and the context for e-speak Gateway Usage:

```
<ContextManager debug="0" workDir="work" home="<installation_directory>/
config/htdocs">
- - -
<!-- 'servlets' for e-speak. -->
<Context path="/servlets" docBase="<installation_directory>/config/htdocs"
debug="9"
reloadable="false"
</Context>
<!-- used for browser access: http://localhost:8080 -->
<Context path="" docBase="<installation_directory>/config/htdocs" debug="9"
reloadable="false" >
</Context>
```

Verifying the Tomcat Installation

To verify Tomcat installation, start and stop Tomcat from the `<installation_directory>/config` directory using the scripts provided with the installation.

Starting Tomcat

To start Tomcat:

1 Navigate to the `<installation_directory>/config` directory:

```
cd <installation_directory>/config/
```

2 Enter the start command:

```
./htdocs/tomcat.sh start
```

You will see the message `Starting Tomcat` and the classpath.

Stopping Tomcat

To stop Tomcat:

1 Navigate to `<installation_directory>/config` directory:

```
cd <installation_directory>/config/
```

2 Enter the stop command:

```
./htdocs/tomcat.sh stop
```

You will see the message `Stop Tomcat` and the classpath.

Testing Gateway Usage Configuration When Using Tomcat

NOTE: Stop all JVMs before starting this procedure

1 Start the core.

Table 36 Testing Gateway Usage — starting the core

Platform	Command
Windows NT®	cd <installation_directory>\bin ./espeak
HP-UX™ and Linux™	cd <installation_directory>\bin .\espeak

2 Start Tomcat in a DOS window.

Table 37 Testing Gateway Usage — start Tomcat

Platform	Command
Windows NT®	cd <installation_directory>/config htdocs/tomcat start or htdocs/tomcat run
HP-UX™ and Linux™	cd <installation_directory>/config ./htdocs/tomcat.sh start

3 Launch a web browser, such as Netscape Navigator or MS Internet Explorer.

4 Point the browser to `http://localhost:8080`.

NOTE: Depending on your configuration, you may need to use your actual hostname instead of localhost if the Login screen does not display.

If the browser window does not display the login window:

- Ensure that the servlet installation is working properly (even without e-speak)
- Ensure that the Tomcat server is running properly
- Ensure that Tomcat started properly

- **Verify the files:**
 - `<installation_directory>/config/conf/server.xml`
 - `<installation_directory>/config/htdocs/server.xml`
 - `<installation_directory>/config/htdocs/tomcat.bat` **or**
`tomcat.sh` file

5 Click the **Login** link.

6 Enter `esadmin` as both the username and password.

7 Click **Login**.

From this window, you can:

- Register and locate vocabularies and services
- Register users

Troubleshooting Gateway Usage When Using Tomcat

- Ensure that the configuration files, `PATH` and `CLASSPATH` do not have any typing errors or case differences.
- Before you start the core, ensure that no JVMs are running.
- Ensure that you have set the system variables `TOMCAT_HOME`, `JAVA_HOME`, `ESPEAK_HOME`.
- Check the logs under `<installation_directory>/config/htdocs/logs`:
`servlet.log`
`tomcat.log`
- Check the `<installation_directory>/config/htdocs/server.xml` file to ensure that the entry for e-speak servlets is as specified in the installation instructions.
- Verify that the `web.xml` file exists in the `<installation_directory>/config/htdocs/web-inf` subdirectory.
- Verify that the servlet and servlet mappings are:

```
<servlet>
    <servlet-name>
        WebAccess
```

```
        </servlet-name>
        <servlet-class>
            net.espeak.webaccess.WebAccess
        </servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>
            WebAccess
        </servlet-name>
        <url-pattern>
            /WebAccess
        </url-pattern>
    </servlet-mapping>
    <servlet-mapping>
        <servlet-name>
            WebAccess
        </servlet-name>
        <url-pattern>
            /WebAccess/
        </url-pattern>
    </WebAccess/>
```

- **Most problems with using Gateway Usage are related to faulty configurations in `webaccess.xml`. Make sure you have read the installation and configuration notes on this file.**
- **Make sure your `webaccess.xml` file is in a directory in your `CLASSPATH`.**
- **Make sure site-specific plug-ins implement the right interfaces and have default public constructors.**
- **Make sure the web server is set up to find the `webaccess` directory.**
- **Make sure the `webmacro.properties` file has been changed to refer to the location of the `<installation_directory>\webaccess\htdocs\templates` directory.**
- **Make sure your proxy server is configured correctly so that the `java.net.URL` class can connect to your service.**

- Make sure to configure the database connection information for the persistent message manager in `webaccess.xml`.

Firewall Gateway

This section describes:

- **Orientation to the firewall gateway** — introduces the e-speak firewall gateway framework and describes the firewall gateway's constraints to clarify standard scenarios and protocols
- **Infrastructure requirements** — specifies hosting parameters and requirements for making a connection (for example, from an intranet to a demilitarized zone (DMZ))
- **Firewall gateway configuration** — describes configuration files for the firewall gateway
- **Internal engine configuration** — defines when to modify the internal engine's e-speak configuration file
- **How an external engine connects to the firewall gateway** — covers associated configuration properties and gives code samples
- **Running the firewall gateway**
- **Reconfiguring the firewall gateway** — lists the procedures for performing administrative tasks while the gateway is running
- **Limitations** — lists currently-known limitations

Orientation

The e-speak firewall gateway (also called the SLS gateway) is software-based and mediates access between external clients and e-speak services that reside behind a corporate firewall (internal services). The clients may be located either on the Internet or on the Intranet of another company. The firewall gateway acts as an access control point between external clients and internal services by authenticating and authorizing external clients based on SLS attribute certificates.

In a typical scenario, the gateway resides between an external client's engine and an internal service's engine. It authenticates and authorizes all requests from the external entities (both clients and engines) based on company-wide access requirements. A company can use the SLS gateway to impose certain basic, or default, access control requirements on external entities before they access any internal e-speak engines or services. These requirements are over and above the access control requirements imposed by the individual internal engines or services.

In e-speak terms, an external client/engine must possess valid certificates to satisfy both the requirements of the internal engine/service and those of the gateway to access the internal engine/service.

While mediating access between an external client and an internal service, the gateway does not compromise end-to-end security properties between the two end points (authentication, confidentiality, and integrity).

Figure 33 illustrates a typical use of the gateway.

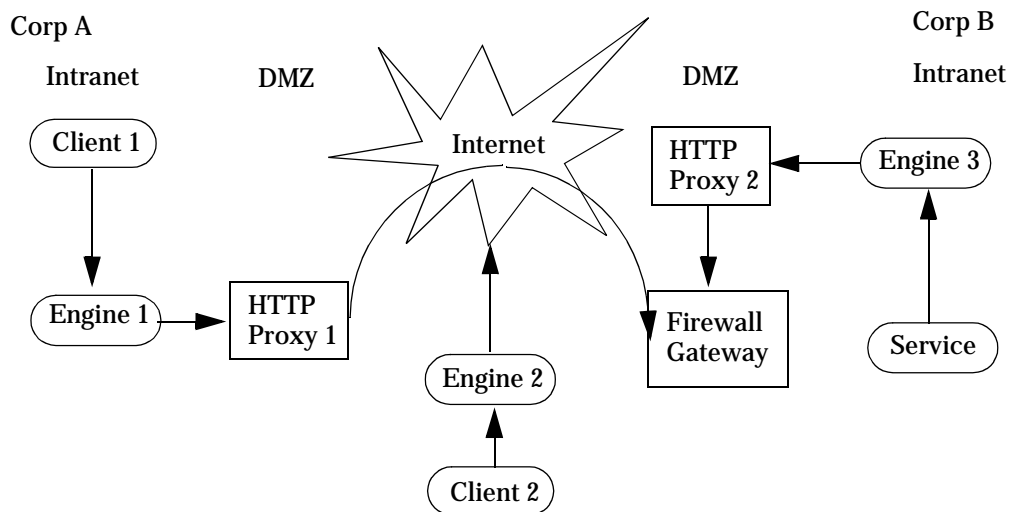


Figure 33 Gateway usage scenario

NOTE:

- The gateway resides between two engines; it is not meant to be used between an e-speak application and an engine.
- The gateway does not act as a mediating point between an internal client and an external service. It protects only internal services and does not prohibit internal clients from accessing external services.
- There is exactly one gateway between two engines. Gateways cannot be cascaded.
- The gateway is neither an e-speak engine nor an e-speak application; it is a plain Java™ application that understands e-speak security and messaging mechanisms (uses e-speak libraries).

Infrastructure Requirements

The gateway is typically hosted in the a company's DMZ. Because the gateway cannot make inbound connections to internal engines, the engines must make an outbound connection to the gateway. This is a connection from the Intranet to the DMZ and requires either an HTTP proxy or a Socks (>= V4) server at the firewall to make the connection, unless the firewall allows for a direct connection from the internal entities to the external/DMZ entities.

Firewall Gateway Configuration

This section lists the configuration files necessary to configure the firewall gateway.

E-speak-Specific Configuration File

The gateway uses the standard e-speak configuration file (by default `$espeak_home/config/espeak.cfg`) to read e-speak specific configuration information. This includes security configuration such as security role, the location of the `pse` file, and the location of certificates. Although the gateway runs as a standalone Java™ application, it re-uses various e-speak libraries (especially the security code). This is the man reason it needs the e-speak configuration file: for the sake of the e-speak libraries that need this file.

Gateway-Specific Configuration File

The gateway executable uses an XML configuration file as a command line parameter. This file must conform to the `dtd`:

```
$espeak_home/config/slsgateway/SLSSConfiguration.dtd
```

You can view a sample XML configuration file at:

```
$espeak_home/config/slsgateway/SLSSConfiguration.xml.
```

Use this sample as a template for writing your own configuration file:

```
<?xml version="1.0"?>
<!DOCTYPE SLSGatewayConfiguration SYSTEM "SLSSConfiguration.dtd">
<SLSGatewayConfiguration>
  <ConfigMode>Online</ConfigMode>
  <InsideAddress>myhost.mycompany.com</InsideAddress>
  <OutsideAddress>myhost.mycompany.com</OutsideAddress>
  <ExternalListeners>
```

```
<ExtListener>
  <Port>2950</Port>
</ExtListener>
<ExtListener>
  <Port>12349</Port>
</ExtListener>
</ExternalListeners>
<InternalListeners>
  <IntListener>
    <Port>22345</Port>
  </IntListener>
  <IntListener>
    <Port>22346</Port>
  </IntListener>
</InternalListeners>
<SocketPool>
  <MaxInboundConnection>10</MaxInboundConnection>
  <MaxOutboundConnection>10</MaxOutboundConnection>
</SocketPool>
<Logger>
  <LogLevel>DEBUG</LogLevel>
  <FileLocation>src/java/net/espeak/security/application/slsgateway/
log/testlog</FileLocation>
</Logger>
<PolicyConfiguration>
<InternalPolicies>
<BaseInternalPolicy>
  <Tags>(net.espeak.method internal (*) (*))</Tags>
</BaseInternalPolicy>
<InternalPolicy>
  <HostPort>testService.rgv.hp.com:2950</HostPort>
<Exposed>true</Exposed>
<Tags>(net.espeak.method pqr (*) (*))</Tags>
</InternalPolicy>
```

```
<InternalPolicy>
    <HostPort>testService.rgv.hp.com:12349</HostPort>
</Exposed>true</Exposed>
<Tags>(net.espeak.method pqr (*) (<!--
    </InternalPolicy>
</InternalPolicies>
<ExternalPolicies>
<BaseExternalPolicy>
    <Tags>(net.espeak.method external (*) (<!--
    </BaseExternalPolicy>
<ExternalPolicy>
    <HostPort>testService.rgv.hp.com:2950</HostPort>
<BaseCoreManagedResTags>(net.espeak.method baz (*) (<!--
    BaseCoreManagedResTags>
<BaseServiceTags>(net.espeak.method xyz (*) (<!--
    </ExternalPolicy>
<ServicePolicy>
    <URL>esp://testService.rgv.hp.com:2950/Core/
    RemoteResourceManager</URL>
</Exposed>true</Exposed>
    <Tags>(net.espeak.method abc (*) (<!--
    </ServicePolicy>
</ExternalPolicies>
</PolicyConfiguration>
</SLSGatewayConfiguration>
```

NOTE: You can only omit the items marked as optional. Currently, the XML file must conform to the dtd specifications for the gateway to work predictably. Any XML information that is not read properly may lead to a lack of security. All XML parsing errors are printed to a standard error, whether you have debug on or not.

Configuration Mode

The current version supports **Online** mode only.

Gateway Inside and Outside Address

The gateway uses these addresses to listen for internal and external connections, respectively. Internal engines establish internal connections to the gateway; external engines establish external connections. Because the gateway can run on a host with multiple transports, you can use this configuration information to specify the specific transport settings to be used for internal and external connections. Always use either a full host name or an IP address. Do not use a localhost for this field.

External and Internal Listeners

These specify the ports on which the gateway should listen for external and internal connections, respectively. You can specify multiple external and internal ports so that the gateway will create multiple server sockets to listen for connections.

NOTE: Since the gateway acts like a proxy on behalf of the internal engines, an internal engine's port must match an external port of the gateway. For example, if the gateway's external ports are 2950 and 2950, then the internal engines must also listen on one of these ports.

SocketPool

The SocketPool specifies the maximum number of internal (inbound) and external (outbound) esip connections the gateway will accept.

Logger

The logger specifies the logging level and the log file location. The directory in which the log file will be created must already exist.

Policy configuration

This is the most important part of the gateway configuration. You must specify which internal entities are exposed and what tags (e-speak sls tags) will be checked for accessing these entities.

Table 38 Policy descriptions

Policy	Description
Internal Policies	<p>Specifies the internal engines that may connect to the gateway to expose their services externally</p> <p>This ensures that unauthorized internal entities do not expose their engines and services externally and compromise the company's security policy. The gateway always authenticates an internal engine and will also check its certificates if so specified in the policy.</p>
BaseInternalPolicy (optional)	The base set of tags that will be checked before any internal engine is allowed to connect to the gateway
InternalPolicy	<p>For each internal engine exposed, there must be an InternalPolicy entry. This entry contains the engine's host:port and specifies:</p> <ul style="list-style-type: none"> • Whether or not the engine is exposed • Optional tags that must be checked before the engine can connect to the gateway
External Policies (optional)	Specifies the tags that the gateway must check against an external entity's certificates
BaseExternalPolicy (optional)	Specifies the set of tags to be checked from external entities for all inbound accesses, regardless of the internal engine or service they are trying to access

Table 38 Policy descriptions

Policy	Description
ExternalPolicy (optional)	Provides, on a per internal engine basis, a set of tags that must be checked when accessing resources on that engine You can use this portion to specify the separate sets of tags to be checked for accessing engine-managed resources and services on that engine.
ServicePolicy (optional)	Specifies, on a per service/engine-managed-resource URL basis, whether a specific URL is exposed and what tags (if any) must be checked before allowing access to the service/resource

Internal Engine Configuration

You must modify the internal engine's e-speak configuration file to specify the gateway host and port. The internal engine uses this host:port to make a connection to the gateway on bootup. The internal engine must possess valid certificates to connect to the gateway (if specified in the gateway policy configuration).

```

#####
! gatewayhost is a single value being the fully qualified hostname
! of the gateway
#####
net.espeak.infra.core.gateway.gatewayhost=gatewayHost.hp.com
#####
! gatewayport is the port on which the gateway listens for internal
! connections
#####
net.espeak.infra.core.gateway.gatewayport=22346
#####
! virtualhostname is the fully qualified virtual name that is being used
! for the
! hostname on which the local engine runs.
#####
net.espeak.infra.core.virtualhostname=testService.rgv.hp.com

```

You can use the last configuration property to specify a virtual host name for the engine. Use this to hide the real host name of the engine from external clients.

You must also specify properties to let the internal engine know how to traverse the firewall (to connect to the gateway). For example, if you want to use an HTTP proxy for firewall traversal, you can specify it as:

```
#####
! webproxyname is a string denoting the fully qualified hostname
! of the http-proxy used to pierce a firewall, if any.
#####
net.espeak.infra.cci.messaging.httpProxyName=web-proxy.rgv.hp.com
#####
! webproxyport is the port on which the proxy listens
#####
net.espeak.infra.cci.messaging.httpProxyPort=8088
#####
! Use these parameters to specify the user name and password if the proxy
! requires password authentication.
#####
net.espeak.infra.cci.messaging.user=www
net.espeak.infra.cci.messaging.passwd=passwd
#####
! Domain names for which direct connection should be done.
! multiple entry are supported, they should be seperated by spaces
! The wildcard ('*') supported and IPv4 is supported
!net.espeak.infra.cci.messaging.noHttpProxy=*.*.hp.com localhost 10.20.*
    1.2.3.4
#####
net.espeak.infra.cci.messaging.noHttpProxy=*.rgv.hp.com
#####
! Whether you want to use socks or http
#####
net.espeak.infra.cci.messaging.proxy=http
```

You can define similar properties using Socks. Socks is preferable over HTTP proxy, because socks-based connections tend to be more reliable. Please refer to the sample `espeak.cfg` (`$espeak_home/config/espeak.cfg`) for details.

If your firewall allows internal entities to connect to external ones directly, you should set the property `net.espeak.mfra.cci.messaging.proxy` to `none`.

Gateway parameters for connection:

- `net.espeak.infra.core.gateway.retryinterval=60000`
- `retryinterval` is the interval after which the internal core repeatedly tries to connect to the gateway and is specified in milliseconds. Default timeout of 1 min. (60000 ms)
- `net.espeak.infra.core.gateway.connection_timeout=60`
- `connection_timeout` is the CoreChannel connection timeout specified in seconds. Default timeout of 1 min. (60 s) is used if the following line is commented out or removed.

How an External Engine Connects to the Firewall Gateway

Once the external client finds a service located behind a corporate firewall, it opens a connection between its local engine and the service's engine using the service's firewall gateway. While the query can be accomplished by searching an Advertising Service accessible to both the client and the service (such as the one located at the e-services Village portal), opening a connection through the firewall gateway is different. There are two approaches:

- **DNS Address Modification** — This assumes that a service's engine runs on the machine `testService.rgv.hp.com` and the company's firewall gateway runs on the machine `gateway.hp.com`. In the external address, add a CNAME entry to map `testService.rgv.hp.com` to `gateway.hp.com`:

```
testService.rgv.hp.com CNAME gateway.hp.com
```

When the external engine tries to resolve `testService.rgv.hp.com`, it:

- Resolves to `gateway.hp.com`
- Further resolves it to the gateway's IP address
- Makes an `esip` connection to the gateway

Because the gateway already has a connection open to the service's engine, it can forward messages between the two engines.

NOTE: Changing the `hosts` file on the external engine's machine to map `testService.rgv.hp.com` to the gateway's IP address has the same effect as adding the `CNAME` entry in the external address. Use this strategy only for testing, however.

- **Using special advertising service group format** — This strategy uses a special e-speak advertising service group format to access a service behind a firewall. To use this scenario, the internal engine must run an advertising service and then advertise its internal service to this advertising service. The client accesses the service by first finding it in the internal advertising service. Consider that the internal engine runs on `host1:port1` and the gateway runs on `host2` with its external listener port as `port1` (as mentioned earlier, the gateway's external port and the internal engine's port must match). If the internal advertising service has a group name of `group1`, then the client can use the following e-speak community specification to access the internal advertising service (refer to E-speak Programmer's Guide for details on advertising service groups and communities and their uses):

```
community = host1:port1;host2:port1/group1
```

This specification allows the client's engine to establish a connection to the internal engine via the firewall gateway. The client will be able to do a J-ESI "find" on the internal advertising service and to find the internal service.

You can specify the community either via the client's properties file or programmatically using J-ESI calls. The server side does not need any gateway specific code. Also, if you specify the community via the properties file, no gateway specific code will be required on the client side.

This strategy requires that the client knows the gateway host used by the internal advertising service and the internal service. In order to make the interaction more dynamic, use the next strategy, albeit at the cost of extra programming efforts.

- **Using a special vocabulary**— If you don't want to change address entries or the `hosts` file, the server and client sides must do some extra work at the application level to achieve the same result.

Assume that the client finds the service URL through an Advertising Service accessible to both the client and the service.

For a service provider:

- 1 Describe the service in a vocabulary with a string attribute called `firewallGateway` (the name of this attribute is not important as long as it is known both to the client and the service provider. The name `firewallGateway` is used just as an example).
- 2 Set the value of the attribute to the `hostname` of the gateway machine.

3 Advertise the vocabulary and the service in the Advertising Service.

NOTE: This means that the service can not use the base vocabulary. Also, if the service was already using a non-base vocabulary, an extra string attribute will need to be added.

For a client:

- a** Find the vocabulary and the service using the Advertising Service.
- b** Find the value of the `firewallGateway` attribute for the service.
- c** Open a connection through the gateway to the remote engine that hosts the service.

Example:

```
// Get ESConnection to the local engine.
ESConnection es = new ESConnection("localhost", 2950, "esip");
// Find the vocabulary. We are assuming that the advertising service
// group is already included in the e-speak "community".
ESVocabularyFinder vf = new ESVocabularyFinder(es);
// These two flags need to be set for using services located behind
// a firewall gateway. They ensure that J-ESI does not try to open a
// connection to the remote engine on its own. This is required as
// we need to use a special openConnection call that understands how
// to open a connection through the firewall gateway.
vf.setInterfaceVerificationFlag(false);
vf.setOpenConnectionFlag(false);
ESVocabulary v =
    vf.find(new ESQuery("Name == 'FirewallGatewayVocab'"));
// Find the service based on this vocabulary
String interfaceName = EchoServiceIntf.class.getName();
ESServiceFinder sf = new ESServiceFinder(es, interfaceName);
sf.setInterfaceVerificationFlag(false);
sf.setOpenConnectionFlag(false);
ESQuery query = new ESQuery(v, "Name == 'InternalService'");
EchoServiceIntf echoService = (EchoServiceIntf)sf.find(query);
// Get the value of the "firewallGateway" attribute for the
// service.
String attrs[] = new String[1];
```

```

attrs[0] = "firewallGateway";
ESCommunity community =
    es.getServiceContext().getCurrentCommunity();
ESAccessor srvcAcc =
    ((ESBaseServiceStub)echoService).getAccessor();
String [][] values =
    community.findAttributes(srvcAcc, query, attrs);
String gatewayHost = value[0][0];
// Get the gateway port. This is the same as the port on which the
// remote engine listens. So we get this from the service accessor.
StringTokenizer st =
    new StringTokenizer(srvcAcc.getESName().getServerString(), ":");
String temp = st.nextToken();
int gatewayPort = Integer.parseInt(st.nextToken());
// The openConnection call.
String coreId = "esp://" + srvcAcc.getESName().getServerString();
ESRemoteConnectionManager conMgr = es.getConnectionManager();
String conId = conMgr.openConnection(coreId, gatewayHost,
    gatewayPort);
// Make calls on the service.
String reply = echoService.echo("Hi there!");

```

Running the Firewall Gateway

You must have all of the appropriate e-speak libraries in your class path:

```

cd $espeak_home
java net.espeak.security.application.sls.gateway.gateway.Gateway
<appropriate dir>/SLSSConfiguration.xml

```

The file `SLSSConfiguration.dtd` must be located in the same directory as the file `SLSSConfiguration.xml`.

You can also start the gateway through standard e-speak mechanisms by running the `$espeak_home/bin/espeak.pl` script. The `gw.ini` is located in the `$espeak_home/config/slsgateway` directory. Please modify the `.ini` file to reflect the location of your XML configuration file.

If you run the gateway with the `useGui` parameter, a GUI window provides a live summary of the gateway usage statistics. This includes the number of:

- Messages/bytes sent inbound/outbound by the gateway

- Connections opened
- Messages dropped

Click **Reload** to refresh the interface with the most current statistics.

If you run the gateway without the `useGui` parameter, the gateway will run without the GUI statistics.

To enable debug messages from the gateway, enable debug for `slsgateway` in the gateway's e-speak config file (typically `espeak.cfg`). For example:

```
ESTrace=default
default= net.espeak.util.ESPrintWriter()=\
*-EMERG:*-ALERT:*-CRIT:*-ERROR:*-WARNING:*-NOTICE:*-INFO:slsgateway-
debug1\
;Severity+Invoker
```

Reconfiguring the Firewall Gateway

While the gateway is running, you can modify the XML configuration file and reconfigure the gateway by clicking on `reconfigure`. This may include adding new policy requirements or changing the connection listener ports. It will not affect existing connections and sessions through the gateway, but any new connection requests will follow the new configuration. Note that the `reconfigure` option is available only when the gateway is used with the `useGUI` command line option.

Limitations

- The `reconfigure` option is not completely thread-safe. Receiving new connection requests while reconfiguration is in progress may cause rejected requests.
- Reconfiguration does not always stop existing connection listeners. The expected behavior is that the gateway will stop all connection listeners and restart them based on the new configuration. Based on how server sockets function, you may get an `address already in use` exception. This will not interfere with the gateway's functions.
- The gateway currently does not support security for the XML configuration file and the log file. Use operating system mechanisms to protect these files.
- You cannot use the gateway in conjunction with a three-engine scenario. In an earlier e-speak version, an e-speak engine residing in the DMZ acted as a router between an engine in the corporate Intranet and another one outside the

corporate Intranet (may be inside the Intranet of another corporation). This is the schema is the three-engine scenario. Refer to the *E-speak Programmers Guide* for details.

Although the three-engine scenario will still work independently, it may not work appropriately when a gateway is added. For example, you may not have a scenario such as the one shown in [Figure 34](#).

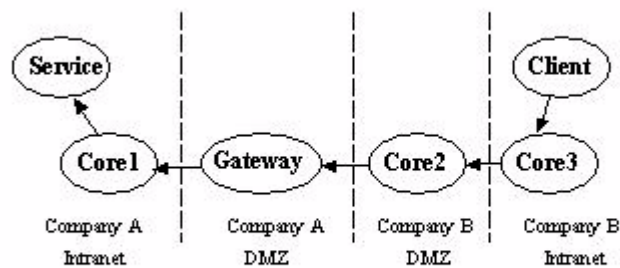


Figure 34 Three-engine scenario

Chapter 9 **espeak Utility**

Use the **espeak** utility, located in `<espeak_installation_directory>/bin`, when starting the e-speak Core and other components.

Table 39 espeak Utility

Operating System	File Name
Linux™ or HP-UX™ Note: You must have perl 5.003 or later to run the espeak utility.	espeak.pl (perl5 script)
Windows NT®	espeak.exe

NOTE: The default version included with the standard HP-UX™ installation is Perl4, not Perl5.

You can use the **espeak** utility to both start basic e-speak components and load user-defined `.ini` files.

NOTE: An experimental utility **espeak3** is also available in the `<espeak_installation_directory>/bin` directory that allows you to start advanced e-speak components.

Displaying Help

To display help:

- 1** Type `espeak -h`.
- 2** Press **Enter**.

Help Text

Usage : Execute E-speak component(s).

```
Syntax: espeak [ --version | -v ]
        espeak [ --help | -h | /? ]
        espeak [ -c ] [ -j(opt) <JVM arguments> ]
                [ configfile=<config file> ] [ configpath=<config path> ]
                [ esport=<port number> ] [<name>=<value> ...]
                [-i inifile | <E-speak components> ]
```

Details:

-h, --help, /?	Print this screen.
-d, --debug	Sets debug mode ON
-j, -jopt <JVM Arguments>	Extra arguments to the JVM. Quotes needed if contains spaces.
-i <ini file>	User supplied Ini file.
-v, --version	Prints version number and exits.
configfile=<config file>	Specifies the e-speak's core configuration file. The default file looked for is 'espeak.cfg'.
configpath=<config path>	Path to config file. The file is looked for in the following places: config directory under e-speak home as defined by property 'espeak_home'(defined in this script), directory specified by option 'configfile', or current directory(from system property 'user.dir') if the property is not set directory specified by 'user.home' system property as a system resource from the classpath
esport=<port number>	TCP Port number where e-speak Core is started.
<name>=<value>	Assign 'value' to JVM Property 'name'. Typically used in user supplied ini file which contains JVM properties (specified by %<name>%)

<E-speak components> One or more E-speak components and their arguments
 These are ignored if -i is specified. Following
 are currently valid E-speak components and their
 arguments. By default Core, AdvertisingService,
 ServiceDistributor are started.

Arguments that can be specified with any of the components:

run=newjvm|injvm Optional. This will start the component in a new
 JVM, or same JVM. Default is injvm

Valid E-speak Component(s) and their respective arguments :

 c | Core

 p=<Protocol info> Is esip:<tcpport>
 user=<user> User name to access backend DB
 password=<Password> Password to access backend DB
 load=<plugin class> Load a plugin class.
 restart Restart the core purging the repository.

 ads | AdvertisingService

advconfig=<config file> Specifies the e-speak Advertising Service
 configuration file
 hostname=<host name> Specifies the host name of the core the
 Advertising Service should connect to.
 portnumber=<port number> Specifies the port number of the core the
 Advertising Service should connect to.
 advgroup=<group name> Specifies the group the Advertising
 Service belongs to.
 advprotocol=<esvProxy|slp|ldap|esvProxyOnly>
 Specifies the type of backend service.
 directory to be used by the Advertising
 Service.

ldapconfig=<config file>	Specifies the LDAP configuration file.
esvconfig=<config file>	Specifies the esvProxy configuration file.
mcastport=<port number>	Specifies the port number for multicasting.
mcastpriority=<positive integer>	Sets a value for e-speak's multicast lookup feature.
mcastsleep=<milliseconds>	Specifies the time, in milliseconds, an Advertising Service in slp mode waits after multicasting before finishing initialization.
mcastgroupip=<IP address>	Specifies the multicast GroupIP address.
ldaphost=<host name>	Specifies the hostname of the LDAP server.
ldapport=<port number>	Specifies the port number of the LDAP server.
ldaprootdn=<name>	Specifies the LDAP directory manager username.
ldappasswd=<password>	Specifies the LDAP directory manager password.
ldaproxyhost=<host name>	Specifies the web proxy hostname used to traverse a service's firewall.
ldaproxyport=<port number>	Specifies the web proxy port number used to traverse a service's firewall.
ldaporg=<Root entry>	Specifies the root entry of the LDAP directory tree
ldapbranch=<branch name>	Specifies the branch of the LDAP server directory tree used to keep Advertising Service advertisements.
ldapoolmax=<integer>	Specifies the number of simultaneous connections to open with the LDAP server.
ldapoolperiod=<milliseconds>	Specifies the time, in milliseconds, to wait before closing an idle LDAP server connection.
esvhostport=<host:port/count[,...]>	Specifies alternate hosts and ports to use to connect to the e-services Village proxy. For example: esvhostport=www.foo.com/43210/2

```

esvmaxtrials=<positive integer>    Specifies the maximum number of failed
                                     connection attempts to the ESV Proxy.
esvusername=<username>             Specifies the e-services Village username.
esvpassword=<password>             Specifies the e-services Village password.
    
```

For more information, see SystemAdministration.pdf.

sd | ServiceDistributor

```

CORE_PORT=<port>                    Port number where core was started.
CORE_HOST=<host>                    Host name where the core was started.
CORE_URL=<URL>                      CORE_HOST:CORE_PORT instead of specifying
    
```

rc | RunClass

```

class=<class>                       Class (fully qualified name) to run
arg=<arguments>                     Arguments to the class
    
```

dt | desktop

```

[schemafilename]                   The file name of a alternative schema file.
    
```

Environment Variables:

Following Environment variables should be defined.

```

ESPEAK_HOME                        Location where E-speak is installed (mandatory).
ESPEAK_JHOME                        Full path to your JVM Installed location.
                                     If not set, 'java' from PATH is attempted.
    
```

Examples:

- o To start E-speak PSE Manager
espeak RC class=net.espeak.security.pse.manager.Run

- o To start the E-speak Desktop
espeak desktop

- o To run the default E-speak components
espeak

- o To run only Core at TCP port 12390.
espeak C p=esip:12390

- o Execute Core and Advertising Service at specified group.
espeak C Ads advgroup=myPrivateGroup

Note:

In order to terminate the program, you must hit <CTRL>C

Caution:

All JVM's that get started do not always get terminated, particularly on Windows NT. Please kill those JVM processes, else the port numbers remain occupied and you hit exceptions when starting the services again.

Chapter 10 Troubleshooting

If you encounter a question or issue with e-speak, ensure that:

- You have installed the proper versions of all required products
- Your environment variables are pointing to the correct locations

If, after performing these tasks, you have not resolved the issue:

- 1** Check the e-speak `readme` file and *Release Notes* for the latest product updates and notes.
- 2** Check the on-line manuals most appropriate for your issue.
- 3** Check the *e-speak Programmers Guide*.
- 4** Contact the e-speak support team at:
 - www.e-speak.hp.com
 - www.e-speak.net

Installation and Configuration

This section discusses issues that can occur due to improper installation and configuration.

Working with the (PSE) tool

Scenario	Running the PSE tool results in the following error: <pre>net.espeak.security.pse. ProtectedDataObjectException: BinaryFilePDO.constructor> OS file protection library "Osprotect" required - but could not be loaded.</pre>
Reason	The <code><installation_directory>\lib</code> is not in the PATH variable.
Solution	OSProtect.dll needs to be in the variable, which is referenced by PSE libraries. It is available under <code><installation_directory>\lib</code> . This path has to be included in the system PATH setting.

Working with JServ

Scenario	JServ cannot find the CLASSPATH variable for WebAccess and other servlets.
Solution	Go to <code>http://localhost/jserv/</code> (assuming the port is 80) > Mapped servlet engines > ApacheJServ/1.1b3 Servlet Engine Status. Apache1.3.12/Jserv1.1 is the supported version. Verify that the CLASSPATH variable displays all the files mentioned with wrapper.classpath. If it doesn't, you need to shutdown the client and bring it up again to have the CLASSPATH available for JServ.

Determining which Components Have Started

<p>Scenario</p>	<p>E-speak starts and displays these messages: ESCore starting with In-Memory Repository Creating Factories Loading Plugins Core Initialized Starting ESCore Server with Rendezvous of "tcp:2950" Started Espeak core The default .ini file includes: [Tasks] Start=Core,ClientCF,AdvertisingService,CoreDistributor, ServiceDistributor,ManagementDistributor</p>
<p>Solution</p>	<p>If a service fails during start up, it throws an exception. However, it can still fail during run-time because there is a dependency on the ServicePack for some e-speak functions, such as messaging. The latest e-speak code is available free of charge to registered developers at http://www.e-speak.hp.com/developers.</p>

J-ESI

Using Non-Serializable Classes in a ParameterList

<p>Scenario</p>	<p>A user creates a service and uses <code>InputStream</code> as part of a <code>ParameterList</code>. A <code>NotSerializableException</code> is thrown.</p>
<p>Reason</p>	<p>The class used is not a serializable class.</p>
<p>Solution</p>	<p>Only objects that have the <code>java.io.Serializable</code> interface, can be sent as a parameter to the <code>addObject</code> method. In the present case, <code>InputStream</code> doesn't implement the <code>java.io.Serializable</code> interface. Its instance cannot be serialized; therefore, the <code>NotSerializableException</code> is thrown.</p>

An Unexpected Exception when Returning User-defined Datatype

Scenario	A user wants to know if it possible to return user-defined types when calling the stub. It works properly when a string is returned, but when he tries to return a user-defined type, he receives a run-time exception: <code>Unexpected exception or Invalid call sequence</code>
Reason	E-speak uses the default Java serialization.
Solution	The method parameters in the interface, which any e-speak service exposes, are sent through the e-speak core. When the user-defined types are not explicitly registered with the e-speak message registry, e-speak uses the default Java serialization. One or more data types may not be <code>java.io.Serializable</code> .

J-Kernel

Resolving a CorePanicException at the Client Side

Scenario	A CorePanicException is thrown on the client side: <code>net.espeak.infra.cci.exception.CorePanicException</code> <code>Core panic: java.io.EOFException</code>
Reason	The core has panicked because the connection no longer exists.
Solution	This issue indicates that the socket between the client and core is dead. The core may have crashed or it may have decided to close the connection. One reason the connection was dropped between the client and the core might be that the client was supposedly stopped and unregistered during the scavenging process.

Client API

Adding a Community through a Property File

Scenario	Unable to add a community through a property file other than through e-speak API ESCommunity.
Reason	A community or complete machine name need to specified with a specific format.
Solution	<p>A community can be specified through a property file in the following way:</p> <pre>community = localhost:2950/HPgroup, localhost:12347/HPgroup, localhost:12348/SUNgroup, localhost:12349/SUNgroup</pre> <p>Alternatively, a complete machine name should be specified in the following way:</p> <pre>community = pc1.rgv.hp.com:12347/grp3_csp056,pc1.rgv.hp.com:2950/grp2_csp056</pre>

Using the ESDebug class, Routing Output to Stdout

Scenario	A user cannot set up and use the ESDebug class and route the debug1() method's output to stdout.
Reason	The user must set a specific ESDebug class.

Using the ESDebug class, Routing Output to Stdout

<p>Solution</p>	<p>To set up an ESDebug class, use the following:</p> <pre> classpublic class MyClient { private static final ESDebug debug = new ESDebug("intercorecom"); private static final String CLASS_NAME = "tests.java.net.espeak.infra.intercorecom.TestClient "; // your class name debug.debug1(CLASS_NAME, "*Here is your Debug Message* "); throw ffel; </pre> <p>To route the debug1() method's output to stdout, turn the debug flag to on in the espeak.cfg file and then view the debug1() method's output in your console.</p>
-----------------	---

Finding the Advertised Service

<p>Scenario</p>	<p>In a multi-core environment, the user registered and advertised a service, but the client cannot find the advertised services.</p>
<p>Reason</p>	<p>The server and the client may be in different groups.</p>
<p>Solution</p>	<p>If the server is started on core A and the client is started on core B, the Advertising Services started on core A and core B should be in the same group.</p>

Sample Application

LookupFailedException: while running Client application

Scenario	While running the client application, this error message is displayed: <code>Connected to e-speak net.espeak.infra.cci. exception.LookupFailedException: Lookup failed</code>
Reason	The client could not found the Service.
Solution	The possible reasons could be: <ul style="list-style-type: none">• The service is not registered and/or started.• The client looks from a different core; that is, the service is not advertised properly.• The service is registered in a different vocabulary than the one the client would normally use to find the service.• Clients do a <code>find()</code> before the service registers.

WebAccess

Working with DOS or Other Legacy Applications

Scenario	A user wants to distribute an existing DOS application. The application performs complex calculations that cannot be migrated easily.
Solution	<p>There are several ways to do this. You can do this through a servlet, you can access the DOC application (or other legacy applications) through JNI. For more information, see this URL:</p> <p>http://java.sun.com/j2se/1.3/docs/guide/jni/index.html</p> <p>You can also call *.exe by using runtime methods. For example:</p> <pre> Process p = Runtime.getRuntime().exec(new String("your DOS command here")); InputStream is = p.getInputStream(); int r; while ((r = is.read()) != -1) { char c = (char)r; //whatever you want to do with the output of the DOS command } </pre> <p>The services, implemented using servlet can be e-speak enabled using web access.</p>

Logging in as the e-speak Administrator

Scenario	The e-speak core is up and running and the e-speak login page is displayed. Apache JServ 1.1.2 is also up and running. The user is unable to login as esadmin.
Solution	<p>The problem may be because some settings are not in place. Consider the following:</p> <ul style="list-style-type: none"> • JServ may not be able to find the CLASSPATH for WebAccess. Make sure all the settings mentioned for all the config files and properties files are set. • Go to <code>http://localhost/jserv/</code> (assuming the port is 80) > Mapped servlet engines > ApacheJServ/1.1b3 Servlet Engine Status. Apache1.3.12/JServ1.1 is the supported version. • Verify that CLASSPATH displays all the files mentioned with wrapper.classpath. If it doesn't, you need to shutdown the client and bring it up again to have the CLASSPATH available for JServ.

Setting the IP Address

Scenario	<p>A user assumes that his IP address is incorrect and that is why he is unable to ping it or trace to it and that is why I am getting a time-out noroutetohost exception. The user has these values set in the proxy.cfg file:</p> <pre>host: web-proxy.reliance.com port: 8088</pre>
Solution	Update the proxy.cfg file to reflect your proxy server and port number.

Security

Understanding the Resource Mask

Scenario	In reading an example in the <i>e-speak Programmers Guide</i> , it appears that the resource mask is used to disable the need for certificates in order to use certain methods. The notes indicate that because a mask has been created for <code>checkStatus</code> , one does not need a certificate to use this method.
Solution	<p>If masking is off, you must have an appropriate certificate to connect to the core which in turn acts as a mediator for the service. This means the client should have a certificate with a tag name (<code>net.espeak.*</code>) After a connection is established between both the parties. Any of the interface's methods at the service end could be invoked even if you do not have certificates. Note that both parties should have certificates to invoke methods in the core.</p> <p>If masking is turned on, and the service provider doesn't set any authorization tags during the registration phase, any client would be able to invoke methods.</p> <p>If masking is turned on, and the service provider sets authorization tags during the resource registration phase, any client wanting to invoke methods on the service should provide appropriate certificates issued by the service provider.</p>

Working with Certificates

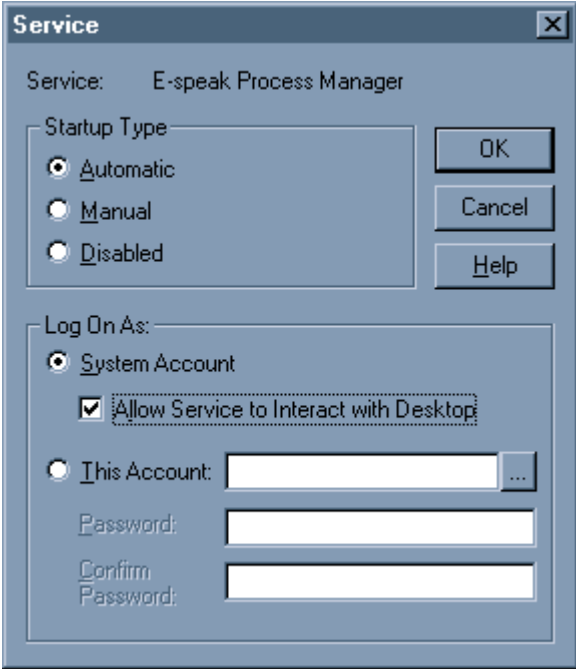
Scenario	A developer is writing a service in which clients can access some methods only if they present tags that match the methods. The developer would like to know how clients present these certificates in the J-ESI model.
Solution	For more information, refer to the examples at http://wray-m-4.hp1.hp.com:8008/public/security2000-06-14/examples/ .

Working with the client.prop File

Scenario	A developer wants to know how to create user credentials in the client.prop file.
Solution	<p>This is the sample entry in the client.prop file:</p> <pre> General e-speak properties username : Cards password : passwd accountname : Games hostname : localhost portnumber : 2950 protocol : TCP sessionname : MySession community : ntxx.india.hp.com:22020/ Games,ntxx.india.hp.com:22021/Game </pre>

Working with the Desktop

Displaying Windows

Scenario	When running services and processes from the Desktop, the appropriate windows do not display.
Solution	<p>Configure the e-speak Process Manager service to interact with the desktop.</p> <p>In Windows NT:</p> <ol style="list-style-type: none"> 1 Select Start>Settings>Control Panel. 2 Select Services. 3 Select E-speak Process Manager. 4 Click Startup. 5 Select Allow Service to Interact with Desktop.  <ol style="list-style-type: none"> 6 Click Ok. <p>In Linux and HP-UX, ensure that you have set the environment variable to the proper display.</p>

Appendix A Advanced E-speak Configuration Parameters

Tracing

Tracing Security Components

Security

net.espeak.security.trace

Purpose	Traces the e-speak security component
Default	off
Options	on off <ul style="list-style-type: none">• on – Starts tracing• off – Stops tracing
Modify when	Modify this parameter to turn tracing on or off
Impact	Tracing generates additional text messages; a large number of extra messages may effect e-speak performance

net.espeak.security.sls.SessionManager.trace

Purpose	Traces low-level, secure socket layer (SLS) operations such as connect , send , and receive Note: These operations are performed during secure communications between clients, services, and cores.
Default	off

`net.espeak.security.sls.SessionManager.trace`

Options	<code>on</code> <code>off</code> <ul style="list-style-type: none">• <code>on</code> – Starts tracing• <code>off</code> – Stops tracing
Modify when	If you are debugging a problem, tracing session operations let you determine if communication sessions are being successfully created and whether messages are being sent and received.
Impact	Tracing generates additional text messages. Session level tracing is detailed and may effect e-speak performance.
See also	net.espeak.security.trace

Security

Security Properties

Table 40 lists properties supported by the security code.

Table 40 Security properties

Property	Description
Properties	<p><code>net.espeak.security.connectOnContact</code>, default <code>off</code></p> <p>This controls whether secure sessions are established with newly-encountered resources. When it is <code>off</code>, sessions are not established unless required by <code>SessionRequiredException</code> or created explicitly.</p>
PSE mode	<p><code>net.espeak.security.pse.mode</code>, values: <code>gui</code>, <code>passphrase</code>, <code>passfile</code>, default <code>gui</code></p> <p>If the mode is <code>gui</code>, an interface retrieves the PSE passphrase. If the mode is <code>passphrase</code> the property <code>net.espeak.security.pse.passphrase</code> retrieves the passphrase (default <code>null</code>). If the mode is <code>passfile</code> the property <code>net.espeak.security.pse.passfile</code> (default <code>passfile.txt</code>) retrieves the name of a file that must contain a <code>net.espeak.security.pse.passphrase</code> property defining the passphrase.</p>
PSE key file	<p><code>net.espeak.security.pse.storefile</code>, default <code>securestore.bin</code></p> <p>This defines the name of the file containing public-private key pairs.</p>
PSE role	<p><code>net.espeak.security.pse.role</code>, default <code>client</code></p> <p>This defines the default role (symbolic PSE key name).</p>
PSE file protection mode	<p><code>net.espeak.security.pse.OSfileprotection</code>, default <code>true</code></p> <p>This property specifies whether local OS file protection is applied as an aid for testing purposes only. For full security protection, set this option to <code>true</code>.</p>
Certificate file suffix	<p><code>net.espeak.security.pse.certfile</code>, default <code>certs.adr</code></p> <p>The value of this property is appended to the role name to retrieve the name of the certificate file to load. For example, if the role is <code>client</code>, the certificate file is <code>clientcerts.adr</code>.</p>

Table 40 Security properties

Property	Description
ACL file suffix	<p><code>net.espeak.security.pse.aclfile</code>, default <code>acl.adr</code></p> <p>The value of this property is appended to the role name to retrieve the name of the ACL file (trust assumptions) to load. For example, if the role is <code>client</code>, the certificate file is <code>clientcerts.adr</code>.</p>
Cipher suites	<p><code>net.espeak.security.cipherSuites</code></p> <p>The value is a list of cipher suites in ADR syntax. The default is <code>hmac, sha-1</code>, and 128-bit <code>blowfish</code>.</p>

Sample `espeak.cfg` file

```

=====
! E-speak security properties file.
=====

net.espeak.security.activate=on
user.name="John Doe"

! Example time value.
  foo.timeout = 12h 3m .0001s

! Gui mode runs a dialog for the passphrase.
  !net.espeak.security.pse.mode=gui

! Passphrase mode looks for the passphrase property.
  !net.espeak.security.pse.mode = passphrase

! Passfile mode looks for a file containing the passphrase
!property.
  !net.espeak.security.pse.mode = passfile

! Define the passphrase.
  !net.espeak.security.pse.passphrase = default passphrase

! Define the default role (PSE key name).
  !net.espeak.security.pse.role = foo

```

NOTE: Refer to [“Security”](#) for detailed information about setting security properties.

Determining Whether Stub Class Files are Downloaded

net.espeak.security.allow_stub_class_download

Purpose	<p>Determines whether the <code>stub</code> class file is downloaded</p> <p>To use an e-speak service, the <code><servicename>Stub.class</code> file must reside on the client machine. Typically, when you write a client, this file already exists; you do not need to download the file from the server. Because of this, the parameter is commented out and defaults to <code>false</code>.</p> <p>If you need to download the stub class file from the server, set this parameter to <code>true</code> on both the client and the server.</p>
Default	<code>false</code>
Options	<p><code>true</code> <code>false</code></p> <ul style="list-style-type: none"> <code>true</code> - Turns off security <code>false</code> - Turns on security
Modify when	You want to download the stub class file
Impact	Use this parameter with caution. Downloading a file exposes the client machine to a security risk. The server can download and run any code on the client, making the client vulnerable to attack. Only set this parameter to <code>true</code> if you trust the server from which you are downloading.

Specifying the Security Setup Duration

net.espeak.security.sls.setup_duration

Purpose	Specifies the maximum time, in milliseconds, to create a Secure Socket Layer (SLS) connection
Default	120000 milliseconds (2 minutes)
Options	A positive integer value
Modify when	An unacceptably large number of SLS connections time out
Impact	Setting too small a value may cause a larger number of SLS connections to time out.

net.espeak.security.pse

Purpose	Sets the property prefix to properties starting with a dot For example, the prefix <code>.pse.mode=passphrase</code> is equivalent to <code>net.espeak.security.pse.mode=passphrase</code> .
Default	N/A
Options	A valid e-speak property beginning with a dot

net.espeak.security.pse.mode.gui

Purpose	Sets the PSE mode to <code>gui</code> and causes a window to prompt you for the passphrase
Default	<code>gui</code>
Options	<code>gui</code> <code>passphrase</code> <code>passfile</code>
Modify when	You want to change the setting of <code>.pse.mode</code>
Impact	This is the most secure setting, but requires attended operation; you must supply a passphrase.
See also	net.espeak.security.pse.mode=passphrase

net.espeak.security.pse.mode=passphrase

Purpose	Determines the method the e-speak Core uses to request the secret passphrase to decrypt the PSE file
Default	<code>passphrase</code> Note: The passphrase must be defined in <code>net.espeak.security.pse.passphrase</code> .
Options	<code>gui</code> <code>passphrase</code> <code>passfile</code> <ul style="list-style-type: none"> • <code>gui</code> - e-speak requests the passfile when needed by a GUI • <code>passfile</code> - must be defined in <code>net.espeak.security.pse.passfile</code>

net.espeak.security.pse.mode=passphrase

Modify when	Modify the value depending on the use of the program. Note: The <code>gui</code> option is the only option that does not require that the passphrase be stored as plain text.
Impact	You cannot use <code>gui</code> mode for batch programs because it requires user interaction.
See also	net.espeak.security.pse.mode=passfile net.espeak.security.pse.mode.gui

net.espeak.security.pse.mode=passfile

Purpose	Sets the PSE mode to <code>passfile</code> The <code>passfile</code> provides the passphrase property. Use this parameter for unattended operation; in this mode, you do not have to supply a password.
Default	<code>gui</code>
Options	<code>gui passphrase passfile.</code>
Modify when	You create a new PSE with a different <code>passfile</code> .
Impact	You must protect the password to prevent it from being stolen.
See also	net.espeak.security.pse.mode=passphrase

net.espeak.security.pse.passphrase=default passphrase

Purpose	Specifies a key to decrypt the PSE file
Default	<i><default passphrase></i>
Options	A case-insensitive string value
Modify when	You create a new PSE with a different passphrase.
Impact	You must also set <code>net.espeak.security.pse.mode</code> .
See also	net.espeak.security.pse.mode=passfile

`net.espeak.security.pse.storefile = securestore.bin`

Purpose	The PSE file contains one or more key pairs (one key pair for each role in the PSE). This file is located in the <code>/config</code> directory.
Default	<code>securestore.bin</code>
Modify when	The PSE name changes

`net.espeak.security.pse.certfile`

Purpose	<p>An entity issues a certificate to allow a second entity to perform operations on the first entity. These certificates are kept in a certificate file; <code>espeak</code> reads and forwards them as needed.</p> <p>The file must be in ADR format. The actual file name differs from the name given here. The file has a <code>role</code> used in the PSE. For example, if the value is <code>certs.adr</code> and the role is <code>core</code>, the file used is <code>corecerts.adr</code>.</p>
Default	<code>certs.adr</code>
Modify when	The certificate file name changes
See also	net.espeak.security.pse.role

`net.espeak.security.pse.acl.adr`

Purpose	<p>A trust assumption is a certificate used by a conveyed trust. An entity decides whether or not to accept another entity's certificate based on whether it is signed by entity in the trust assumptions.</p> <p>Trust assumption certificates are kept in a certificate file. They must always be self-signed certificates.</p> <p>The file must be in ADR format. The actual file name differs from the name given here. The file has a <code>role</code> used in the PSE. For example, if the value is <code>certs.adr</code> and the role is <code>core</code>, the file used is <code>corecerts.adr</code>.</p>
Default	<code>acl.adr</code>
Modify when	The certificate file name changes
See also	net.espeak.security.pse.role

net.espeak.security.pse.role

Purpose	Determines which key pair to use in the PSE There can be many roles defined in each PSE. A key pair with this label must exist in the PSE.
Default	client
Modify when	The role using e-speak changes
Impact	The role name is used to construct the file name for the certificate file.
See also	net.espeak.security.pse.certfile net.espeak.security.pse.acl.adr

Setting Threads for Polling

The e-speak Core contains a thread pool that processes client requests. When a client sends a message to the Core, a thread is assigned from `WorkerThreadFactory` to process the request. The `WorkerThreadFactory` increases or decreases, based on the number of client messages. When the load of incoming messages from clients decreases, threads in the pools decrease until the number of threads equals the minimum number of threads set in the `WorkerThreadFactory.min` parameter.

net.espeak.util.thread.WorkerThreadFactory.min

Purpose	Specifies the minimum number of threads available for polling
Default	10
Options	The value to specify depends on the client activity on your network. For example, if the Core supports a large number of clients (e.g., 1000) that are active for a brief period, set this value to a high number. If you do not set the value high enough, responses to clients are delayed.
Modify when	There are not enough threads available for the amount of clients attempting to send messages
Impact	Denial of service may occur if the number of clients trying to contact the Core exceeds the number of available threads.

net.espeak.util.thread.WorkerThreadFactory.max

Purpose	<p>Specifies a maximum number of threads for polling (the maximum number present in the Core)</p> <p>Typically, define this parameter when you define the <code>WorkerThreadFactory.min</code> parameter. Together, these parameters define a range of threads that the Core's pool requires to receive messages from clients.</p>
Default	500
Options	This value depends on client activity on your network. For example, if the Core supports 1000 active clients and each thread can serve an estimated 5 clients, set the maximum number of threads to 250 for the shared connections, plus the number of dedicated connections.
Modify when	There is a relatively high number of threads defined compared to the active number of clients trying to send messages
Impact	You must set this parameter to a large enough value to ensure adequate system resources.

net.espeak.util.thread.WorkerThreadFactory.min_channel_writers

Purpose	<p>Specifies the minimum number of threads inside the Core that write messages to clients</p> <p>The Core creates and starts the specified number of <code>min ChannelWriter</code> threads at startup. This parameter indicates the minimum number of <code>ChannelWriter</code> threads that are always present in the e-speak Core.</p>
Default	8
Options	This value depends on client activity on your network. For example, if the Core supports a large number of clients continuously, set this parameter to a high value. A too-low value may cause response delays.
Modify when	Reduce this parameter if a relatively high number of threads has been defined compared to the active number of clients trying to read messages from the Core.
Impact	Too high of a value allows inactive <code>ChannelWriter</code> threads to remain in memory, using system resources.

net.espeak.util.thread.WorkerThreadFactory.max_channel_writers

Purpose	<p>This value specifies the maximum number of <code>ChannelWriter</code> threads (threads that write messages to clients) the Core can create</p> <p>The Core dynamically increases the number of <code>ChannelWriter</code> threads to the upper limit of <code>max_channel_writers</code>. The Core creates a new <code>ChannelWriter</code> thread only if it must send a message and all the existing <code>ChannelWriter</code> threads are busy.</p>
Default	8
Options	<p>This value depends on the client activity on your network. For example, if the Core supports a large number of clients intermittently, set this parameter to a large value and set <code>min_channel_writers</code> to a value sufficient for a normal load. The Core dynamically increases the number of <code>ChannelWriters</code> during a heavy load and reduces them when the load subsides.</p>
Modify when	<p>Reduce this parameter if a relatively high number of threads has been defined compared to the active number of clients trying to read messages from the Core.</p>
Impact	<p>If you do not set the value high enough, clients will experience response delays during periods of heavy load.</p>

Configuring the Engine to the Firewall Gateway

```
net.espeak.infra.core.gateway.gatewayhost
net.espeak.infra.core.gateway.gatewayport
```

Purpose	<p>These parameters make Intranet e-speak services (internal services) available to external clients through an e-speak firewall gateway. These two parameters specify the e-speak firewall gateway's internal host and port, which are used by engines hosted in the Intranet (internal engines) to connect to the gateway.</p> <p>A firewall gateway makes e-speak services deployed on an Intranet assessable to external clients. An internal engine, at boot time, must connect to its company's e-speak firewall gateway before it can make its services accessible to external clients.</p> <p>Because the gateway is designed to run on a host that has two network interfaces, it runs with two different host and port settings:</p> <ul style="list-style-type: none"> • An internal host and port setting to be used by internal engines • An external host and port setting to be used by external clients and engines <p>Point these two parameters to the internal host and port on which your company's firewall gateway is running.</p> <p>Because the firewall gateway is deployed in the corporate firewall, an engine in the Intranet cannot make a direct <code>esip</code> connection; you must also specify e-speak firewall traversal related parameters such as Web proxy configuration or SocksV4 proxy configuration.</p>
Options	A valid host name and port number
Modify when	When you change your host name and post number
Impact	<p>When you specify these parameters, the engine tries connection to the gateway at boot time. If a connection cannot be established, it does not affect booting, but services on that engine are inaccessible to external clients.</p> <p>If you do not specify the parameters, the engine does not attempt to connect to the gateway.</p>

net.espeak.infra.core.virtualhostname

Purpose	<p>Conceals Intranet host names from external clients</p> <p>E-speak services are identified by a URL with the format:</p> <p>esp://host:port/service-path</p> <p>where the <code>host:port</code> refers to the machine on which the service's engine runs.</p> <p>When you make services accessible to external clients using a URL, you also make Intranet host names available. If you specify this parameter, all service URLs are based on a virtual hostname rather than the internal engine's IP address.</p> <p>You can also use this parameter to specify more meaningful URLs for services than is possible through the use of real host names in URLs. For example, if the real name of the machine running an engine is rgelpc158.rgv.hp.com, a service URL may be:</p> <p>esp://rgelpc158.rgv.hp.com:12345/service-name</p> <p>If the engine hosts services that deal with various aspects of customer relationship management (CRM), you could specify the virtual hostname as crm.hp.com, with a URL of:</p> <p>esp://crm.hp.com:12345/service-name</p> <p>Set this parameter when the hostname of the machine on which the Core is running changes, such as when the machine is a DHCP client or in a high-availability cluster, to allow a Core with a given URL to start independently of the hostname its machine.</p>
Options	Any virtual host name
Modify when	You want to create or modify a virtual host name
Impact	If you do not specify this parameter, the engine uses the real host name of the machine. All service URLs contain this host name.

net.espeak.infra.core.gateway.GWconnector.java

Purpose	Specifies the interval (in milliseconds) after which the service engine tries to connect to the gateway. The service engine tries to make this connection regardless of whether the gateway is up or down.
Default	60000 milliseconds (1 minute)
Options	A positive integer value.

Preventing Denial of Service Attacks

net.espeak.infra.core.mailbox.Timeout

Purpose	<p>Specifies the timeout (in seconds) to read a single packet.</p> <p>If the Core does not receive the packet in the specified amount of time, the connection closes to prevent denial of service attacks by deliberately-slow transmissions.</p> <p>Typically, a denial of service attack occurs when a client (or clients) sends a large number of packets to a server in a short amount of time, overwhelming server resources. However, denial of service can also occur when a client sends packets to the server too slowly (for example, one packet per minute). Core resources are significantly compromised in both cases.</p>
Default	300 seconds (5 minutes)
Options	Calculate a timeout value based on your network's typical performance. As a guideline, use 1 MB as the maximum packet size and divide that by the optimal speed of your connection. For example, if most of your connections are over a 10 Mb/sec LAN then, ideally it takes 0.8 seconds to receive a packet. Pad this optimal transmission rate to take slower performance into consideration. In this example, you could set the timeout value to 2 seconds.
Modify when	Users complain of excessive time-outs

Configuring the Advertising Service

net.espeak.services.advService.config

Purpose	<p>Specifies the e-speak Advertising Service configuration file, allowing multiple, related Advertising Services to share common settings to make configuration easier and ensure consistency</p> <p>Specify the path to the config file on a command line:</p> <pre>net.espeak.services.advService.config=<adv_cfg_path></pre>
Options	The name of the Advertising Service configuration file
Modify when	Typically, do not modify this file.
Impact	If you do not set this property, it is not used.
See also	Chapter 7, "Running Standard Services"

hostname
portnumber

Purpose	<p>Specifies which Core the Advertising Service is connecting to. The Core <hostname>:<portnumber> parameters connect to the Advertising Service. These are typically specified in Advertising Service command lines as:</p> <pre>hostname=<host> portnumber=<port></pre> <p>Each Advertising Service must connect to a different Core. In general, specify these parameters on the Advertising Service command line as:</p> <pre>hostname=<host> portnumber=<port></pre> <p>If the Core <hostname>:<portnumber> uses a virtual host name, ensure that the you set the Advertising Service with the same property value.</p> <p>Always set this parameter if the hostname of the machine on which the Core is running may change, such as when the machine is a DHCP client or in a high-availability cluster to allow a Core with a given URL to start independently of the hostname of the machine on which the Core.</p> <p>Note: This is equivalent to the <code>-eshost</code> and <code>-esport</code> options in previous versions of the Advertising Service.</p>
Default	localhost:2950

hostname
portnumber

Options	A valid host name and port number
See also	net.espeak.infra.core.virtualhostname Chapter 7, "Running Standard Services"

net.espeak.services.advService.backend.ldap.config

Purpose	Specifies the path of a file containing LDAP configurations, allowing multiple, related Advertising Services to share common settings to make configuration easier and ensure consistency
Options	Any valid pathname
Modify when	This property has no effect unless you specify the LDAP mode. LDAP-related properties in a file are used if they are not defined in the command line, advertising configuration file (if any), or <code>espeak.cfg</code> . Other properties in this file have no effect.

net.espeak.services.advService.backend.esvProxy.config

Purpose	Specifies the path of a file containing <code>esvProxy</code> configurations This parameter enables Advertising Services that use the e-services Village proxy to share common setting to make configuration easier and ensure consistency.
Options	Any valid pathname.
Modify when	Typically, do not change this parameter once it is set.
Impact	This proxy has no effect unless you specify <code>esvProxy</code> or <code>esvProxyOnly</code> mode. <code>esvProxy</code> -related properties in a file are used if they are not defined in the command line, advertising configuration file, or <code>espeak.cfg</code> . Other properties in this file have no effect.
See also	"E-services Village (ESV) Proxy Settings"

Multicast Settings

SLP-mode Advertising Services rely on multicast to discover one another. Core discovery is also attached to an Advertising Service's multicast functions.

`net.espeak.services.advService.mcast.port`

Purpose	Specifies the port number for multicasting, enabling an Advertising Service to use an alternative multicast port and avoid conflict with other applications This parameter is equivalent to the <code>-mport</code> option in previous versions of the Advertising Service.
Default	427
Options	Any valid multicast port number
Modify when	Modify this parameter so that it matches the port number of existing multicast applications.
Impact	SLP-mode Advertising Services must use the same multicast port to see one another.

net.espeak.services.advService.mcast.priority

Purpose	<p>Sets a numeric value for e-speak's multicast lookup feature, allowing an installation to set up multiple cores:</p> <ul style="list-style-type: none"> • One as a primary Core • One or more as secondary cores <p>E-speak clients use the <code>ESConnection.findCores()</code> method or set the <code>hostname</code> property to <code>multicast</code> to use this feature. In both cases, the Core's priority determines to which Core to connect. In the <code>findCores</code> method, the priority is returned to the client for examination. When <code>hostname = multicast</code>, the Core with the highest priority is automatically used. In this way, the primary Core is set up with the highest value and each secondary Core is set up with lower values.</p> <p>Normally, when the client tries to connect through the multicast hostname, the primary Core is used. If it is not available, the highest secondary Core is used, and so on.</p> <p>This parameter is equivalent to the <code>-priority</code> option in previous versions of the Advertising Service.</p>
Default	0
Options	<p>The appropriate values depend on your environment. Typically, the values are small numbers such as:</p> <ul style="list-style-type: none"> • 10 for the primary • 9, 8, 7, and so on for the secondaries <p>You can only assign positive numbers.</p>

net.espeak.services.advService.mcast.sleepMillis

Purpose	<p>Specifies the amount of time, in milliseconds, that an Advertising Service waits after multicasting itself before it finishes initialization</p> <p>This wait allows time for an Advertising Service to discover and establish a connection.</p>
Default	30,000 milliseconds (30 seconds)

`net.espeak.services.advService.mcast.sleepMillis`

Options	Set this parameter to 0 if the Advertising Service is not running in SLP-mode or if there is minimal time elapsing when Advertising Services discover one another.
Modify when	Modify this parameter according to the network speed in your environment.

`net.espeak.services.advService.mcast.groupIp`

Purpose	Specifies the multicast GroupIP address This parameter allows the Advertising Service to use an alternative multicast group IP to avoid conflict with other applications.
Options	Any valid IP address
Modify when	Typically, do not change this parameter once it is set.
Impact	To discover one another, SLP-mode Advertising Services must use the same group IP.

LDAP Settings

The parameters in this section only have effect when you set the

`net.espeak.services.advService.backend.protocol` **parameter to LDAP.**

`net.espeak.services.advService.backend.ldap.host`

`net.espeak.services.advService.backend.ldap.port`

Purpose	Specify the host name and port number of the LDAP server The Advertising Service uses this information to connect to the LDAP server. These properties are equivalent to <code>-behost</code> and <code>-beport</code> options in previous versions of the Advertising Service.
Default	The default LDAP host name is <code>speakto.me.e-speak.hp.com</code> ; the default LDAP port is 389.
Options	Any valid host and port number for your LDAP server
Modify when	Typically, do not change these parameters once they are set.

net.espeak.services.advService.backend.ldap.rootDN
 net.espeak.services.advService.backend.ldap.passwd

Purpose	Specifies the LDAP directory manager user name and password The Advertising Service uses this information to access the LDAP directory. These parameters are equivalent to the <code>-root</code> , <code>-passwd</code> , and <code>-base</code> options in previous versions of the Advertising Service.
Options	Any valid distinguished name and password
Modify when	You must set the parameters to conform to the LDAP server installation settings.
See also	Chapter 7, "Running Standard Services"

net.espeak.services.advService.backend.ldap.proxyHost
 net.espeak.services.advService.backend.ldap.proxyPort

Purpose	Specifies the web proxy host name and port number used to traverse a service's firewall The Advertising Service uses this information to contact the LDAP server through the firewall.
Options	Any valid host name and port number
Modify when	Specify these properties when LDAP server is outside the firewall.
Impact	Typically, do not change these parameter once they are set.

net.espeak.services.advService.backend.ldap.organization

Purpose	The root entry of the LDAP directory tree for an Advertising Service This string must match the <code>organization</code> setting of the LDAP server.
Default	e-speak.hp.com
Options	The name of root entry for the LDAP directory tree
Modify when	Typically, do not change this parameter once it is set.

net.espeak.services.advService.backend.ldap.branch

Purpose	Specifies the branch of the directory tree in the LDAP server to use to keep advertisements visible to the Advertising Service This allows you to partition the LDAP directory so that it can be shared.
Default	oso
Options	The appropriate branch of the directory tree in the LDAP server
Modify when	You do not want to dedicate the LDAP server for the Advertising Service's exclusive use, but instead want to control location of the ads All the ads visible to this Advertising Service are located under the subtree of <code>organizationUnit=<branch value>, o=<organization></code> .
Impact	Advertising Services that share the same LDAP server can only see one another's ads if this property is the same for all of them.

net.espeak.services.advService.backend.ldap.poolmax

net.espeak.services.advService.backend.ldap.poolPeriodMillis

Purpose	Fine-tunes LDAP performance, particularly to avoid too many simultaneous Advertising Services to LDAP server connections that can slow the LDAP server's performance When started in LDAP mode, the Advertising Service can dynamically increase the number of simultaneous connections it opens to LDAP, up to the maximum number of <code><poolmax></code> . If a connection becomes idle for <code><poolPeriodMillis></code> milli-seconds, it automatically closes the connection. These parameters are optional.
Default	2 for poolmax; 60,000 milliseconds (60 seconds) for poolPeriodMills
Modify when	LDAP server performance is slow

E-services Village (ESV) Proxy Settings

net.espeak.services.advService.backend.esvProxy.hostPort

Purpose	<p>Lists alternative hosts and ports to connect to the e-services Village proxy</p> <p>When the Advertising Service starts in <code>esvProxy</code> mode, it randomly chooses a host/port to use to connect to the Advertising proxy of E-services Village. If the Advertising Service fails to connect to the ESV Proxy through one host/port, it tries another. The format of the list is:</p> <pre><host>:<port>/<n>[,<host>:<port>/<n>]*</pre> <p><n> is the range of ports. For example, <code>www.foo.com:43210/2</code> includes two ports:</p> <ul style="list-style-type: none"> • <code>www.foo.com:43210</code> • <code>www.foo.com:43211</code>
Options	A list of valid hosts and port numbers
Modify when	<p>The value in <code>espeak.cfg</code> is the host/port supported by ESV, the HP sponsored global service directory</p> <p>To use your own customized proxy service (for example, to provide access control to your private LDAP server):</p> <ol style="list-style-type: none"> 1 Start your own version of the proxy service that implements the <code>net.espeak.services.advService.agents.ESVProxyIntf</code> interface. 2 Specify this list with the host names and port numbers of the Cores on which your customized proxy services are running.

net.espeak.services.advService.backend.esvProxy.maxTrials

Purpose	Specifies the maximum number of failed connection attempts to the ESV Proxyhost/ports
Default	2
Options	A non-zero, positive integer
Modify when	Typically, do not change this parameter once it is set.

Configuring the Core to Gateway Connection

net.espeak.infra.core.gateway.gatewayhost
net.espeak.infra.core.gateway.gatewayport

Purpose	Make Intranet e-speak services (internal services) available to external clients through an e-speak firewall gateway Point these parameters to the internal host and port on which your company's firewall gateway is running.
Options	Any valid gateway host and gateway port name
Modify when	Typically, do not change these parameters once they are set.
Impact	If you specify these parameters, the engine tries to connect to the gateway when it boots. If it cannot establish a connection, the engine boots normally but services on that engine are inaccessible to external clients. If you do not specify these parameters, the engine does not attempt to connect to the gateway.

net.espeak.infra.core.virtualhostname

Purpose	<p>E-speak services are identified by a URL with the format: tcp://host:port/service-path</p> <p>host:port is the machine on which the service's engine runs. Because services are identified by a URL, when you make services accessible to external clients using an e-speak firewall gateway, you make Intranet host names also available.</p> <p>Use this parameter to specify a virtual host name for an engine that hosts services accessible outside your network. If you specify this parameter, all service URLs are based on the virtual host name rather than the real name (or IP address) of the internal engine's machine. If you do not specify this parameter, the engine uses the real host name of the machine, and all service URLs contain this host name.</p> <p>You can also use this parameter to specify more meaningful URLs for services than is possible through the use of real host names in URLs. For example, if the real name of the machine running an engine is:</p> <p>rgelpc158.rgv.hp.com</p> <p>You could set the service URL as:</p> <p>tcp://rgelpc158.rgv.hp.com:12345/service-name</p> <p>If the engine hosts services that deal with various aspects of customer relationship management (CRM), you could specify the virtual host name as:</p> <p>crm.hp.com</p> <p>The URL would be:</p> <p>tcp://crm.hp.com:12345/service-name</p>
Default	The real host name of the machine
Options	Any virtual host name
Modify when	Typically, once you create a virtual host name, do not modify it.

Configuring Repository Variables

net.espeak.infra.core.repository.RepositoryParams.CacheSize

Purpose	<p>The cache provides a fast associative lookup of information based on <code>RepositoryHandles</code>. All information loaded into the cache remains there unless the cache reaches its maximum size.</p> <p>When you delete information in the cache, e-speak ensures that the information is stored in the repository database before discarding the cache. Usually, the repository database only stores data for persistent resources; however, if the cache reaches its limit, both persistent and transient data may be written to the database to make room in the cache.</p> <p>The cache must be large enough to hold all resource-specific data currently in use. For an in-memory repository, all resource-specific data must fit in the cache since there is no repository store backing up the cache.</p>
Default	1000000 bytes
Options	<p>Only the digits 0-9 are allowed.</p> <p>The minimum size is 100000 bytes.</p>
Impact	<p>The memory availability on the machine on which the Core is running must have sufficient physical memory for cache memory support. If the cache size is not large enough to hold all the resource data for the in-memory repository or if all the resource data is currently in use for a JDBC repository, you will receive a <code>RepositoryFullException</code>.</p>

net.espeak.infra.core.repository.RepositoryParams.StoreType

Purpose	<p>Specifies whether or not the e-speak Core is persistent.</p> <p>If you want the resources in the Core to be persistent, specify the JDBC option. All persistent resource data is written to this database through the JDBC store so that it is not lost if the Core shuts down. The JDBC store also takes advantage of the database server query facilities to perform lookup operations on the repository database.</p>
Default	INMEMORY

net.espeak.infra.core.repository.RepositoryParams.StoreType

Options	INMEMORY JDBC
Impact	<p>If you specify the JDBC option, you must also set these parameters to point to the correct database:</p> <pre> net.espeak.infra.core.repository.JDBCGLue.driverName net.espeak.infra.core.repository.JDBCGLue.connectionString net.espeak.infra.core.repository.JDBCGLue.JDBCBlobType (optional) net.espeak.infra.core.repository.JDBCGLue.stringLength (optional) net.espeak.infra.core.repository.JDBCGLue.tblExists net.espeak.infra.core.repository.JDBCGLue.noTbl net.espeak.infra.core.repository.JDBCGLue.dupRec net.espeak.infra.core.repository.JDBCGLue.dbFull </pre>

net.espeak.infra.core.repository.QuotaParams.SoftWorkspace

Purpose	<p>The e-speak Core quota system allows you to limit resource consumption on a protection-domain basis. You can configure the Core to restrict each protection domain to a certain number of total bytes occupied by all resources owned by the domain.</p> <p>About 40% of the Core's total storage is soft workspace which is divided among all the protection domains.</p>
Default	8000000 bytes
Options	The parameter value must be greater than 0. Only numeric digits are allowed.
Modify when	You must calculate the quota requirement for the estimated protection domains and the amount of space each protection domain requires. Because the protection domains are guaranteed a soft limit (set in the <code>espeak.cfg</code> file), set the Core total soft workspace value to the protection domain soft limit multiplied by the number of protection domains you expect to have.

net.espeak.infra.core.repository.QuotaParams.SoftWorkspace

Impact	The protection domain soft limit quota size and the Core soft workspace quota size determine how many protection domains you can create in the system when they all use the soft workspace limit.
See also	net.espeak.infra.core.repository.Quota_Params.Hard_Workspace net.espeak.infra.core.repository.Quota_Params.InitialPDHardLimit net.espeak.infra.core.repository.Quota_Params.InitialPDSofLimit

net.espeak.infra.core.repository.Quota_Params.Hard_Workspace

Purpose	<p>The e-speak Core quota system allows you to limit resource consumption on a protection-domain basis. You can configure the Core to restrict each protection domain to a certain number of total bytes occupied by all resources owned by the domain</p> <p>About 60% of the Core's total storage is hard workspace, which is available to protection domains that exceed their soft workspace limits. A protection domain may allocate more storage as long as its byte consumption remains below its hard limit and the hard workspace of the Core is not exhausted.</p>
Default	8000000 bytes
Options	The parameter value must be greater than 0. Only numeric digits are allowed.
Modify when	Tuning the hard workspace size is important. Because the storage allocation for a protection domain is removed from the Core's hard workspace, you must set the hard workspace parameter value so that the difference between the hard and soft workspaces can be shared by all the protection domains if they exceed the soft limit.
See also	net.espeak.infra.core.repository.QuotaParams.SoftWorkspace net.espeak.infra.core.repository.Quota_Params.InitialPDHardLimit net.espeak.infra.core.repository.Quota_Params.InitialPDSofLimit

net.espeak.infra.core.repository.Quota_Params.
InitialPDHardLimit

Purpose	<p>The e-speak quota system provides the ability to limit resource consumption on a protect-domain basis. Specifically, the Core can be configured to restrict each protection domain to a certain number to total bytes occupied by all resources owned by that protection domain.</p> <p>If a protection domain has allocated more storage than its soft limit, it may allocate more storage so long as its byte consumption remains below its hard limit and the hard workspace of the Core is not exhausted.</p> <p>This parameter specifies the initial setting of the hard limit (the maximum number of bytes occupied by all resources owned by a protection domain) when a protection domain is created.</p>
Default	10000000 bytes.
Options	<p>Any non-zero integer value. The person installing the Core determines how to configure the division between the hard and soft workspaces in the Core and the values of the hard and soft limits for each protection domain.</p> <p>This parameter has no upper limit. If the parameter is set to a very large value, each protection domain can allocate its share of the Core's soft workspace and as much of the hard workspace as is available.</p>
See also	<p>net.espeak.infra.core.repository.QuotaParams.SoftWorkspace</p> <p>net.espeak.infra.core.repository.Quota_Params.Hard_Workspace</p> <p>net.espeak.infra.core.repository.Quota_Params.InitialPDSofLimit</p>

net.espeak.infra.core.repository.Quota_Params.
InitialPDSofLimit

Purpose	<p>The e-speak Core quota system allows you to limit resource consumption on a protection-domain basis. Specifically, you can configure the Core to restrict each protection domain to a certain number of total bytes occupied by all resources owned by that protection domain.</p> <p>Each protection domain is guaranteed to be able to allocate resources up to the total byte limit of its soft limit. To guarantee this, the sum of the soft limits of all protection domains must be less than the soft workspace of the Core.</p> <p>A protection domain may always allocate storage so long as its total byte consumption remains below its soft limit. If a protection domain has allocated more storage than its soft limit, it may allocate more storage only so long as its byte consumption remains below its hard limit and the hard workspace of the Core is not exhausted.</p>
Default	30000 bytes
Options	If you specify this parameter, it must be greater than 0; otherwise, the default is value is used.
Modify when	Because protection domains are guaranteed soft limit space, consider the impact carefully before changing this parameter.
See also	<p>net.espeak.infra.core.repository.QuotaParams.SoftWorkspace</p> <p>net.espeak.infra.core.repository.Quota_Params.Hard_Workspace</p> <p>net.espeak.infra.core.repository.Quota_Params.InitialPDSofLimit</p>

Configuring JDBC-related Parameters

Configure the parameters in this section if you set the `net.espeak.infra.core.repository.Repository_Params.StoreType` parameter to `JDBC` so that the Core resources are persistent (written to the JDBC database).

These parameters define important JDBC information for the e-speak Core repository module, including:

- The location of the JDBC driver
- The string used to establish a connection to the database
- Important JDBC error codes, such as whether or not database tables and records exist

When you configure e-speak parameters related to JDBC error codes, refer to your database documentation for the correct codes to specify.

Connecting to the Database

net.espeak.infra.core.repository.JDBCGlue.driverName

Purpose	Specifies the JDBC driver name used to connect to the database
Options	<p>This parameter must include the full path to the database driver. For example, in Oracle8, the JDBC driver name is:</p> <pre>oracle.jdbc.driver.OracleDriver</pre> <p>For mysql:</p> <pre>org.gjt.mm.mysql.Driver</pre> <p>NOTE: Note : The Driver Class files should be download before running the Application.</p> <p>For example :</p> <p>Oracle : Classes12.zip (available at <espeak dir>/extern/oracle-lib/816/classes12.zip</p> <p>MySql:mysqljdbc.jar (available at <espeak dir>/extern/mysql/MySqljdbc.jar</p>
Modify when	Typically, do not change this parameter once it is set
See also	For more information, refer to your database documentation

net.espeak.infra.core.repository.JDBCGLue.connectionString

Purpose	<p>Specifies the connection string the Core repository uses to establish a JDBC connection to the database. You can configure any SQL-based database with a JDBC driver for the repository. For example, if you use an Oracle8 database, you can use:</p> <pre>jdbc:oracle:thin:@(DESCRIPTION=(ADDRESS=(PROTOCOL=TCP) (HOST=myhost.com)(PORT=1521))(CONNECT_DATA=(SID=ORCL)))</pre> <p>To use this driver, you must include the Oracle8 shared library in your PATH command. Be sure to change the HOST, PORT, and SID parameters to match your Oracle database parameters.</p> <p>If you use MySQL as your Database, the connection string should be like this:</p> <pre>jdbc:mysql://localhost or <machine id>/<database name></pre> <p>ex : jdbc:mysql://localhost/test</p> <p>database name = Mysql provides mysql , test as default databases .You can use any one of them or create your own database using mysqladmin tool.</p>
Options	This parameter is database dependent.
Modify when	Typically, do not change this parameter once it is set
See also	<p>Refer to the Oracle8 documentation for information on how to use the oci8 driver.</p> <p>For Mysql documentation , refer to documents available at http://www.mysql.com</p> <p>For the JDBC driver name, refer to your database documentation.</p>

Specifying Error Codes

The e-speak Core repository module must be aware of all possible error codes returned by the database engine in the following cases:

- a table or a view already exists in the database
- a table or a view does not exist
- a table or a view has a duplicate record

NOTE: You must specify these error codes in the parameters listed below. Refer to your database error codes documentation and locate these values.

net.espeak.infra.core.repository.JDBCGLue.tblExists

Purpose	Specifies the database error codes returned if a table or view already exists . For example, the Oracle database returns this error code: <code>ORA-00955 name is already used by an existing object</code> In this case, set the value of this parameter to 955. MySQL uses an error code such as this: <code>net.espeak.infra.core.repository.JDBCGLue.tblExists=0</code>
Options	A single unsigned integer value
Modify when	Your database vendor changes the error codes
Impact	If this parameter is not set properly and there is a JDBC error, the repository module may not be able to determine the "table or database view exists" condition.

net.espeak.infra.core.repository.JDBCGLue.noTbl

Purpose	Specifies the database error codes returned if a table or a view does not exist in the database. For example, the Oracle database returns this error code: <code>ORA-00942 name is already used by an existing object</code> In this case, set the value of the parameter to 942. For MySQL, e-speak works fine with 0 as value of this error code
Default	0
Options	You can only set this parameter to a single unsigned integer.
Modify when	Your database vendor changes the error codes
Impact	If this parameter is not set properly and there is a JDBC error, the repository module may not be able to determine the "table or database view does not exist" condition.
See also	Refer to your database documentation for error codes.

net.espeak.infra.core.repository.JDBCGLue.dupRec

Purpose	<p>Specifies the database error codes that are returned if the repository module attempts to create a duplicate record on a unique key table. For example, the Oracle database returns the following error code if it attempts to insert or update a duplicate record:</p> <pre>ORA-00001 unique constraint (string.string) violated</pre> <p>For Mysql e-speak works fine with 0 as value of this error code</p>
Options	This parameter must be one or more valid database error codes.
Modify when	Typically, do not change this parameter once it is set.
Impact	If you do not set this parameter properly and there is a JDBC error, the repository module may not be able to determine if it is creating or updating duplicate records.
See also	<p>Refer to your database documentation for error codes. For example</p> <p>Oracle : 1</p> <p>Mysql : 0</p>

net.espeak.infra.core.repository.JDBCGLue.dbFull

Purpose	<p>Specifies the error codes returned if the data space is full and unable to extend. For example, the Oracle database returns the following error codes if the tablespace command failed:</p> <p>ORA-01650 unable to extend rollback segment string by string in tablespace string</p> <p>ORA-01651 unable to extend save undo segment string by string for tablespace string</p> <p>ORA-01652 unable to extend temp segment string by string in tablespace string</p> <p>ORA-01653 unable to extend table string.string by string in tablespace string</p> <p>ORA-01654 unable to extend table index.string by string in tablespace string</p> <p>ORA-01655 unable to extend cluster string.string by string in tablespace string</p> <p>ORA-01656 max # extents (string) reached in cluster string.string</p> <p>ORA-01657 invalid SHRINK option value</p> <p>ORA-01658 unable to create INITIAL extent for segment in tablespace string</p> <p>ORA-01659 unable to allocate MINEXTENTS beyond string in tablespace string</p> <p>In this case, set the parameter to 1650, 1651, 1652, 1653, 1654, 1655. For Mysql, e-speak works fine with following values of these error codes (-1,-1,-1,-1,-1,-1,-1,-1,-1,-1), i.e</p> <p>net.espeak.infra.core.repository.JDBCGLue.dbFull=-1,-1,-1,-1,-1,-1,-1,-1,-1,-1</p>
Default	0
Options	One or more valid database error codes separated by commas
Modify when	This parameter is only valid if the Storage_Type parameter is JDBC.
Impact	If this parameter is not set properly and there is a JDBC error, the repository module may not be able to determine the database full condition.
See also	Refer to your database documentation for error codes.

net.espeak.infra.core.repository.JDBCGLue.sqlStringSize

Purpose	Specifies the maximum length of strings (VARCHARs) supported by the repository database. This parameter will determine the maximum size of string attributes and URLs that can be stored in the database. The JDBC configuration parameter allows to configure the size of VARCHAR fields for the particular database.
Default	255 bytes NOTE: Many databases support much higher upper limits for variable-length character strings. For example, Oracle supports up to 4000 bytes; other types of databases may not even support 255 bytes.
Options	If you do not specify this parameter, the attribute names in the vocabulary tables are created with the default size of strings set to 255.
Modify when	You need longer attribute names or URLs and your database supports long strings
Impact	The parameter determines the size of the: <ul style="list-style-type: none"> • URL field in the Resource Specification table (RESSPEC) when you create the repository database schema • String attribute fields in vocabulary tables that are created when a new vocabulary is registered Changing the parameter has no effect on existing tables.

net.espeak.infra.core.repository.JDBCGLue.DBBlobtype

Purpose	The e-speak Core repository must create the internal system tables with the parameter Binary Large Object (BLOB) Fields. This parameter specifies the actual BLOB datatype name used by the database associated with the JDBC BLOB datatype specified in the parameter: <code>net.espeak.infra.core.repository.JDBCGLue.JDBCBlobType</code> For example, in an Oracle8 database, the LONG VARCHAR type is called LONG VARCHAR, but the LONG VARBINARY type is called LONG RAW. For MySql Database, you need to use LONG VARBINARY type: <code>net.espeak.infra.core.repository.JDBCGLue.JDBCBlobType=LONG VARBINARY</code>
Default	If you do not specify a value for this parameter, it defaults to the same value as JDBCBlobType.

net.espeak.infra.core.repository.JDBCGLue.DBBlobtype

Modify when	<p>This parameter is only valid if the Storage_Type parameter is JDBC . You must specify the JDBCBlobType if the database uses a name of the BLOB type other than LONG VARCHAR or LONG VARBINARY.</p> <p>For MySql Database, you need to use LONG VARBINARY type:</p> <pre>net.espeak.infra.core.repository.JDBCGLue.DBBlobType=LONG VARBINARY</pre>
See also	Refer to your database's user guide to find out about the BLOB support data type in the database.

net.espeak.infra.core.repository.JDBCGLue.primarykeysize

Modify when	<p>This parameter depends on the used database:</p> <ul style="list-style-type: none"> • Oracle allows a primary key size of more than 500 • mysql has a limit on the primary key size depending on the operating system: for NT and HP-UX the maximum size is 500, for Linux it is only 256
Default	256 (for mysql) and 500 (for Oracle)
See also	The documentation provided by database vendors on supported primary key size.

net.espeak.infra.core.repository.Repository_Params.JDBSConnectionRetrials

Purpose	Specifies the number of times the service engine should try to re-establish a connection with the database if the database is not available. The retry can be linked to receiving an error code from JDBC.
Default	4
Options	Valid JDBC error codes.
Modify when	You want to re-establish a connection based on an error code being generated.
See also	net.espeak.infra.core.repository.Repository_Params.JDBCConnection Retrials

Configuring Logging

loggerConfig.<loggerName>.formatter

<p>Purpose</p>	<p>Specifies the name of the <code>formatter</code> class. This class formats the log messages. The default formatting of messages is based on XML and is enclosed in tags.</p> <p>For example:</p> <pre><logEntry> <timestamp>...</timestamp> <logLevel>...</logLevel> <message>...</message> </logEntry></pre> <p>You can specify your own formatters. For example, the <code>ErrorLogger</code> uses a different formatter:</p> <pre>loggerConfig.ErrorLogger.formatter=net.espeak.util. logger.ESErrorFormatter</pre>
<p>Default</p>	<pre>net.espeak.util.logger.ESLogFormatter</pre>
<p>Options</p>	<p>You can specify a fully-qualified class name for the formatter; if not, the default value is used.</p>
<p>Modify when</p>	<p>You want to change the format of the messages</p>
<p>Impact</p>	<p>All the log messages are logged with a different format.</p>

loggerConfig.<loggerName>.handler

Purpose	<p>Defines the name of the log handler</p> <p>This class writes the log messages to an output device, which may be a file or a database depending on its configuration. You can select the <code>logHandler</code> or write your own. For example, the <code>ErrorLogger</code> uses a custom log handler:</p> <pre>loggerConfig.ErrorLogger.logHandler=net.espeak.util.logger.ESBaseLogHandler</pre> <p>This <code>logHandler</code> creates a well-formed XML log file by adding the document root tags.</p>
Default	<p><code>net.espeak.util.logger.ESLogHandler</code></p> <p>Note: The default log handler writes to a file.</p>
Options	You can specify a fully-qualified class name for the formatter; if not, the default value is used.
Modify when	You want to change the output device to which messages are logged

loggerConfig.<loggerName>.maxNumFiles

Purpose	<p>Specifies the maximum number of log files to create</p> <p>The log files names depend on the <code>logFileName</code> property. For example, if the name specified is <code>error.log</code>, the log files created are <code>error##.log</code>, where <code>##</code> is the number of a sequential log file, starting at 0 and increasing to $(\text{maxNumFiles} - 1)$. A new log file is created when the current log file size equals the <code>maxFileSize</code> specified.</p> <p>The logs are written so that no log record is split between files. When the last log file is full, the number of the log file is reset to 0 and that log file is overwritten.</p> <p>For example:</p> <pre>loggerConfig.ErrorLogger.maxNumFiles=5</pre>
Default	2
Options	Specify the maximum number of log files appropriate to your needs.
Modify when	The number of log files must be increased or decreased
See also	loggerConfig.<loggerName>.maxFileSize

loggerConfig.<loggerName>.maxFileSize

Purpose	Defines the maximum log file size Once this limit is reached, a new file is created. For example: <code>loggerConfig.ErrorLogger.maxFileSize=4000K</code>
Default	1000K
Options	You can specify any file size that is appropriate to your needs.
Modify when	The log file size must be changed
See also	loggerConfig.<loggerName>.maxNumFiles

loggerConfig.<loggerName>.logging

Purpose	Determines if the logger records messages This is the preferred property to use to turn logging on or off (rather than editing the <code>loggerManager.loggerList</code> property): <code>loggerConfig.ErrorLogger.logging=true</code>
Default	false
Options	true false Note: Any other value will be treated as false.
Modify when	You want to turn logging on or off

loggerConfig.<loggerName>.messageFile

Purpose	<p>The logging framework uses property files for messages to help localize messages so that Japanese users see Japanese messages and French users see French messages. This property specifies the name of the message file, if one is used. Not every logger uses a message file.</p> <p>This property is optional. If you do not specify the parameter, it is not used.</p> <p>The <code>ErrorLogger</code> uses the message:</p> <pre>loggerConfig.ErrorLogger.messageFile=ESErrorMessageProperties</pre> <p>This file must be in a directory or a jar file that is in the application CLASSPATH.</p>
Options	Specify any valid <code>.properties</code> file.
Modify when	You want to use a messages file

loggerConfig.<loggerName>.logFilters

Purpose	<p>Specifies an extra filter for the logger</p> <p>Along with the <code>logLevel</code>, the filters specify the information that is logged. You can configure the logger to log only warning messages coming from the e-speak J-ESI library.</p> <p>The list and the format for specifying the filters is:</p> <pre>loggerConfig.<loggerName>.logFilters.<index>=<log filter class name></pre> <p><loggerName> is the name of the logger.</p> <p><index> is the index of the filter in the list and starts from 0.</p> <p><log filter class name> is the fully-qualified class name of the log filter.</p> <p>These filters are specified for the <code>ErrorLogger</code>:</p> <pre>loggerConfig.ErrorLogger.logFilters.0=net.espeak.util.logger.ESWebAccessFilter loggerConfig.ErrorLogger.logFilters.1=net.espeak.util.logger.ESJesiFilter loggerConfig.ErrorLogger.logFilters.2=net.espeak.util.logger.ESSecurityFilter loggerConfig.ErrorLogger.logFilters.3=net.espeak.util.logger.ESCoreFilter</pre>
Options	You can use any fully-qualified class name. If an error occurs when loading the class, the filter is ignored.
Modify when	You want to add or remove logger filters
Impact	If you do not specify a filter, only the <code>logLevel</code> filters messages. Specifying even one filter works with the <code>logLevel</code> property to ensure that only messages passing the specified filter are logged.

loggerConfig.<loggerName>.external

Purpose	<p>Provides a bridge between the e-speak logging framework and a third party logging system</p> <p>You can connect a third party logging system to the logging framework without making changes in the third party logging framework.</p> <p>It can have two values: true or false. The loggers specified with the e-speak distribution have an external logger called <code>SecurityLogger</code>. The definition of this property for the logger is:</p> <pre>loggerConfig.SecurityLogger.external=true</pre> <p>The users of external loggers are responsible for writing the code that invokes the third party logger. The logging framework provides only a mechanism to start the external logger.</p>
Default	false
Options	<p>true false</p> <p>Note: Specifying anything else is treated as false.</p>
See Also	Refer to the API documentation for more information.

Appendix B Configuring Security

This appendix discusses:

- Private Secure Environment
- Certificates
- Managing Keyrings
- Bootstrap Process for Testing

E-speak security centers around access control to certain services and resources based on a PKI-based certificate system. When a client machine requests a service, the authorization engine within e-speak's security service looks for a valid certificate on the client-side for authorization. Clients with valid certificates are "trusted."

E-speak also uses cryptographic technology to prevent unauthorized access of messages along the client-server connection. Messages are encrypted and authenticated to prevent interception and alterations. This means that a copy of a message cannot be captured and used again at a later date because the attempt is ignored.

E-speak security uses:

- Private Secure Environment
- Certificates

Private Secure Environment (PSE) Overview

E-speak security is an example of a Public Key Infrastructure (PKI). The public key approach requires at least one pair of keys for each participant:

- A private key that you keep to yourself
- A public key that can be widely known

Typically, you use a private key to digitally sign data; others use the corresponding public key to authenticate the data origin.

In the e-speak model:

- Entities are distributed and interact with one another through secure sessions using the SLS protocol. This includes firewall traversal technology. All entities can both use services offered by others and also provide services to others. All parties in secure sessions must be authenticated to each other. SLS secure sessions authenticate both parties involved using challenge-response negotiations based on public-key cryptography.
- Exchanging digitally signed certificates provides access control to services. These certificates act as **tickets** that grant entities the authorization to access and make use of services. Issuing entities (or principals) issue certificates to subject principals who may use them. These certificates can also be chained together to give composite authorizations.

For more details concerning the security model, please consult the J-ESI documentation and the *e-speak Architectural Specification*.

Private Secure Environment

Keys must be protected from unauthorized disclosure and tampering. A PSE provides a secure store containing labeled public/private key pairs.

The file is encrypted, using the key derived from the passphrase. Only those who know the correct passphrase can decrypt the file and access the key to perform operations. The passphrase is not normally held on the system. The knowledge of the public key, together with the capacity to produce valid signatures with the corresponding private key, forms an e-speak identity.

Keys identify the roles held by a particular e-speak entity. By using the private key to sign certificates, the entity can prove the origin and that it was not modified. E-speak entities need at least one key pair and, in general, have more than one.

Any entity that must verify a signature needs the corresponding public key of the private key that was used to sign. Fortunately, the certificate format that e-speak uses generally includes the public key of the issuer signing the certificate, as well as that of the subject. Alternatively, you can distribute the keyring containing your public key to make it available.

To prevent disclosure of a private key, there is no access method provided by the PSE that directly exposes private keys as data. The PSE's API only provides a way to sign particular certificates, through some labeled key pair within the current PSE, without directly accessing the private key. The PSE itself can be stored as a binary file (typically `securestore.bin`) in the local file system. This data is encrypted and a passphrase is required to decrypt the data it contains.

Delegation and Trust Assumptions

E-speak allows authorization to be delegated using certificates. A sequence of delegated certificates can be processed and combined, or chained, to produce an authorization supported by all of the contributing certificates. Validating a chain of delegated certificates must be rooted in something that the verifying entity trusts implicitly. All entities must have their own set of trusted certificates within which all security verifications that they authorize will be grounded. This set of particularly trusted certificates form the Trust Assumptions of an e-speak entity. Each verifying entity selects or defines their own set of Trust Assumption certificates for this expressed purpose.

Each verifier's Trust Assumption represents the root of secure access control; their entire function is to base all authorization decisions derived from them.

PSE Manager

The PSE Manager:

- Creates, stores, and retrieves PSEs
- Manages passphrases for PSEs
- Generates key pairs and is used to assign labels to them
- Creates/imports and edits individual certificates
- Signs and validates individual certificates
- Saves and retrieves collections of certificates to and from text files

- Creates/imports, edits, and retrieves keyrings
- Signs and exports keyrings

End-user services implemented using e-speak security must deploy their own security model tailored to their own needs and requirements. An e-speak API manages and manipulates PSEs.

All users of e-speak-enabled services must have their own PSE to store their own keys, plus their own collections of service-specific certificates issued by those services against some of the users' own public keys. These certificates need not be kept secret (although users may wish to do so for privacy reasons) and can be kept in a public place, even accessible by URL.

Starting the PSE Manager

To start the PSE Manager, type:

```
espeak RC class=net.espeak.security.pse.manager.Run
```

You can also invoke PSE Manager from a shell script.

Creating a PSE

To create a PSE:

- 1 Start the PSE Manager.
- 2 Select the **Key Management** tab.
- 3 Select **File > New**.
- 4 Specify the PSE file name or select an existing file.
- 5 Specify a passphrase. The PSE data is only accessible to those who know this passphrase

Generating Key Pairs

To generate key pairs:

- 1 Select the **Key Management** tab.

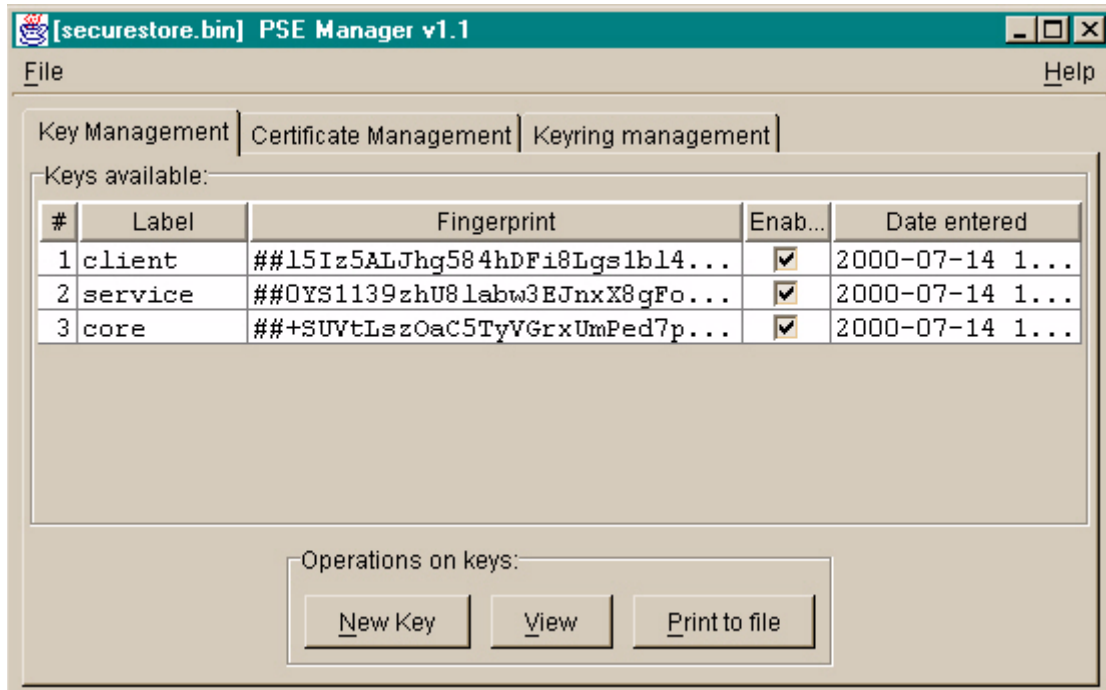


Figure 35 Key Management tab

- 2 Click **New Key**.
- 3 Enter a key label.

NOTE: Only the current default type of key is generated. Labels must be unique. The PSE Manager does allow you to import and export of keyrings, which are files containing public keys.

- 4 Click **OK**.
- 5 Repeat steps 2 - 4 for each key you want to generate.
- 6 Select **File>Save**.

7 Click **Yes** to confirm saving the PSE using the current passphrase.

NOTE: You can change the current passphrase by selecting **File>Passphrase**.

Whenever you overwrite a PSE file, PSE Manager creates a numbered backup file (to a maximum 10 files).

Certificates

There are two basic types of certificates:

- **Attribute certificates** can be chained together via delegation to create indirect authorizations, often based on roles that entities possess.
- **Name certificates** replace symbolic names by particular Principals. Use name certificates to create groups of users so that each member of the group inherits all of the permissions granted to that group as a whole.

Managing Certificates Using PSE Manager

The PSE is essentially a secure keystore and does not contain certificate data. To use key pairs held within a PSE for signing and verification purposes, the PSE Manager allows you to load, process, and save certificate text files.

The **Certificate Management** tab provides several functions.

Table 41 Certificate management functions

Function	Description
New	Create a new certificate
Edit/browse	Edit the selected certificate
Sign	Sign the selected certificate(s)
Validate	Verify that the selected certificates are well-formed, their validity period is meaningful, and (if signed) that the signature is valid
Expand	Displays fingerprints of public keys Note: The fingerprints are useful to compare public keys
Unexpand	Displays public key labels only

Creating Certificates

In the PSE Manager:

- 1 Click **New** in the **Certificate Management** tab.
- 2 Select **Attribute certificate**.

Creating new certificate ...

Type of Certificate:

Name certificate Attribute certificate

Issuer

Issuer key: ##PHyfcVY+ggDFgoV1haKn8jiqwm0=#SHA-1

Issuer names:

Subject

From:

Location: <none> <none>

Key:

Names:

Attribute (tag)

Attribute: (* set)

Derive from interface

May delegate?

Validity

Not before (UTC): 2001-01-04_23:24:33

Not After (UTC): 2001-01-04_23:24:33

View/Edit Text Sign Cancel

Figure 36 Certificate editor

- 3 Select a subject in the **From** drop list.

Once you select the subject, the Location field displays the location of the PSE file. The right side of the field contains the label of the key selected as the subject. After you sign the certificate, the Label field displays the fingerprint of the subject key. The Key field displays the fingerprint of the subject, which is SHA-1 digest of the subject. You can use the Names field to assign a group or a role name to the subject.

- 4 Select **Derive from interface** to create standard authorization tags, with correct SPKI syntax, automatically.
- 5 Click **Add Interface**.
- 6 Locate the interface for which you are generating the tag.
- 7 Click **Open**.
- 8 Select **Set Authorization** to display details on the interface methods.
- 9 Select the specific interface methods to which the certificate will grant access.

or

Select **All** to grant access to all interface methods.

- 10 Click **Ok**.
- 11 Click **Ok** a second time. The **Attribute** field now displays the selected tag.
- 12 Select the **May delegate?** check box to allow the subject to delegate authorization.
- 13 Enter the **Not before** and **Not after** values to define the certificate's validity period.
- 14 Click **View/Edit Text** to make changes to the certificate text, if necessary.
- 15 Click **Sign** to sign the certificate with your private key.

NOTE: Until you sign it, the certificate is in text format with no signature. An unsigned certificate is not valid.

- 16 Click **eXpand** and **Unexpand** in the **Certificate Management** tab to modify the information displayed in the Issuer and Subject panels for a selected certificate.

17 Select File > Save.

NOTE: In standard practice, a certificate file has the suffix `.certs`.

Trust Assumptions

Follow the steps outlined “[Creating Certificates](#)” to create a Trust Assumption certificate, except:

- 1** Enter your own key pair label in the Issuer field
- 2** Enter the issuer you trust (for the tag you create) in the Subject field.
- 3** Save the file as `<your label>trust.acl`. For example, if your label is `translation`, the trust assumptions file is `translationtrust.acl`.

NOTE: You can substitute any name using the configuration property `net.espeak.security.pse.acl` file.

Using PSE Key Labels as Symbolic Principals

Figure 37 shows signed certificates. Because PSE key labels are locally bound names, they cannot occur within signed certificates; any such labels must be expanded to literal Principals (for example, public keys). As a convenience, the signing operation attempts to expand any symbolic Principals (for example, PSE key labels) before actually signing the certificate and fails if there are any symbolic Principals remaining.

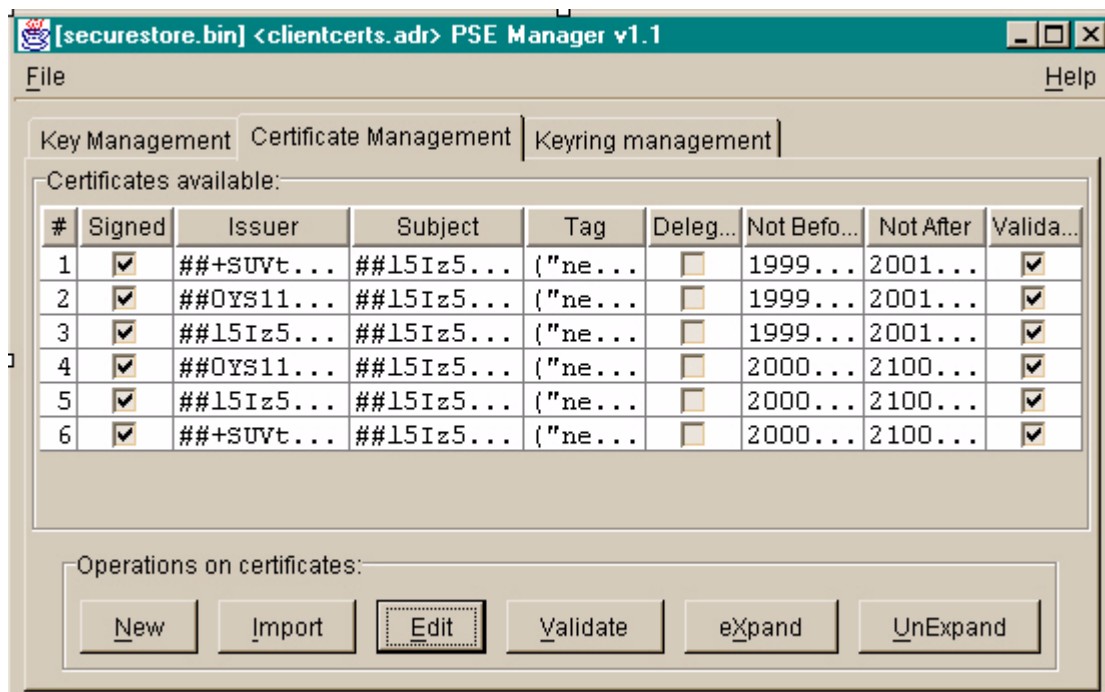


Figure 37 Certificate Management tab

Editing and Browsing Certificates

In the PSE Manager:

- 1 Select the certificate(s) you want to edit in the **Certificate Management** tab.
- 2 Click **Edit**.

Refer "[Creating Certificates](#)" for information on specific fields.

Managing Keyrings

Because the public keys authenticate entities, those entities must both distribute their own public keys to others and store other entities' public keys. Keyrings provide a convenient method to store keys. If an entity wants to distribute a key, it can do so easily by simply distributing the keyring containing its key. The keyring is always signed.

Table 42 Keyring options

Option	Description
New	Creates a new keyring
Open	Opens an existing keyring
Remove	Removes a keyring
Properties	Adds/modifies keyring properties
Name current keyring	Changes the name of the current keyring
Remove entry	Removes an existing element from the current keyring
Rename entry	Renames the current element in the keyring Note: When you rename an element, the previous nickname is displayed in the Description field.
Import KeyRing	Imports a keyring into the current keyring
Export Value	Exports an ADR definition to a file

Creating a Keyring

In the PSE Manager:

- 1 Select the **Keyring Management** tab.
- 2 Select **File>New**.
- 3 Ensure that the relevant keystore is open in the **Key Management** tab, otherwise you will receive a `no keystore open` message.

- 4 Enter the file name (usually `<filename>.keys`).
- You have just created a keyring. For it to be valid, though, it must contain at least one key.
- 5 Select the new file in the **Keyring Management** tab.
- 6 Click **Properties**.
- 7 Select **Add Entry** to add a key.
- 8 Enter a nickname for the key in the **Description** field.
- 9 Select a **Subject**.
- Once you select a subject, the Location field displays the subject's location. The Key displays the fingerprint of the selected subject.
- 10 Enter a subject **Name**, if desired.
- 11 Click **OK** to store the key in the keyring.
- 12 Select **File>Save**. You can view the new keyring in the **Keyring Management** tab.
- 13 Select the appropriate signer from the **Select signer label** window.

Passing a Keyring to a Remote Client

NOTE: For clarity, the two machines in this example are named Computer A and Computer B.

Before you begin, ensure that both Computer A and Computer B both have a role of `client` (`net.espeak.security.pse.role=client` exists in their `espeak.cfg` file).

- 1 Follow the steps outlined in [“Creating a Keyring”](#) to create a signed keyring on Computer A.
- You can now pass this keyring to Computer B.
- 2 Start the PSE Manager on Computer B.
- TIP:** Refer to [“Starting the PSE Manager”](#) if you need instructions.
- 3 Select the **Keyring Management** tab.

- 4 Select **File>Open**.
- 5 Enter the **passphrase**.
- 6 Click **Ok**.
- 7 Select **File>New**.
- 8 Name the key `keystosign.keys`.
- 9 Select the keyring.
- 10 Select **Import Keyring**.
- 11 Locate `clientaccess.keys`.
- 12 Verify the fingerprint.
- 13 Click **Yes**.
- 14 Click **New** in the **Certificate Management** tab.
- 15 Select **Attribute certificate**.
- 16 Select remote client from the **Issuer Key** drop list.
- 17 Select **KeyRing** as the subject from the **From** drop list.
- 18 Specify `keystosign.keys` as the key.
- 19 Specify `client public key` as the Name.
- 20 Set the Attribute tag as:
`(net.espeak.method(*)*)(*)`
- 21 Modify the Validity dates as necessary.
- 22 Select **File>Save**.

Computer B can now pass `clientaccess.certs` to Computer A.
- 23 Ensure that the PSE Manager is still running on Computer A.
- 24 Select the **Certificate Management** tab.
- 25 Select **File>Open**.
- 26 Select `clientcerts.adr`.

27 Click **Import**.

28 Select `clientaccess.certs`.

29 Select **File>Save**.

The certificate is now stored and can be used to request access.

Using PSE Manager as an Attribute Certificate Issuer

You can use the PSE Manager to issue certificates when two separate entities are involved. For example, Client A uses the PSE Manager, loaded with her own PSE. She can pass her public key to Client B to issue her a certificate in one of two ways:

- She can create a self-signed certificate (selecting one of her Principals in both the issuer and subject field) and send the resulting certificate (called `clientaRequest.adr`) to Client B through e-mail.

or

- She can simply send the keyring containing her public key to Client B.

Client B uses his own PSE to import Client A's self-signed certificate as a text file from his e-mail. If he received a keyring, he can import Client A's key from her keyring.

Next, Client B determines whether the certificate really was sent by Client A. To do this, Client B verifies that the fingerprint associated with the Issuer and Subject Principal are identical. Also, Client A could offer additional authentication information that Client B could verify.

Using the PSE Manager, Client B constructs a new attribute certificate containing Client A's Principal as a Subject. Client B adds the appropriate attributes, a validity period, delegation rights, and uses one of his own key pairs as the Issuer. He then signs the resulting certificate, saves it to a text file called `issuedCerts.adr` and sends it to Client A through email.

Client A now possesses a Service Access certificate that she can use to access Client B's service by using the SLS secure session protocol provided within e-speak security.

Sample Certificates Generated by Client A and Client B

This section contains edited versions of the text files, `clientaRequest.adr` and `issuedCerts.adr`, as generated by PSE Manager in the example:

```
clientaRequest.adr
```

```
!!! Certificate Data File : C:\ ... \clientaRequest.adr
!!! Written by PSE Manager v0.5 : $Revision: 1.1.2.2 $ $Date: 2000/05/
    12 14:12:24 $
!!! Dated 2000-05-30_20:32:58

((signed (cert
  (issuer (public-key elgamal-pkcs1 "\003\017v\245\235b\004 ...
    \177.\312\357"))
  (subject (public-key elgamal-pkcs1 "\003\017v\245\235b\004 ...
    \177.\312\357"))
  (tag ("Please issue me a service access certificate - Client A"))
  (not-before 2000-05-30_20:28:38)
  (not-after 2000-05-31_20:28:38)
  ) (signature (hash SHA-1 "a\210i\330\263o\303\336j;q\000\037 ...")
    (public-key elgamal-pkcs1 "\003\017v\245\235b\004 ... \177.\312\357")
      "\003\017l\314\254\227 ..."))
  )
```

```
!!! Summary:
!!! =====
!!! Signed objects      = 1
!!! Unsigned objects    = 0
!!! Name Certificates   = 0
!!! Attr. Certificates  = 0
```

```
issuedCerts.adr
```

```
!!! Certificate Data File : C:\ ... \issuedCerts.adr
!!! Written by PSE Manager v0.5 : $Revision: 1.1.2.2 $ $Date: 2000/05/
    12 14:12:24 $
!!! Dated 2000-05-30_21:00:21
```

```

((signed (cert
  (issuer (public-key elgamal-pkcs1 "\003\017v\245\235b\004 ...
\177.\312\357"))
  (subject (public-key elgamal-pkcs1 "\003\017v\245\235b\004 ...
\177.\312\357"))
  (tag ("Please issue me a service access certificate - Client A"))
  (not-before 2000-05-30_20:28:38)
  (not-after 2000-05-31_20:28:38)
  ) (signature (hash SHA-1 "a\210i\330\263o\303\336j;q\000\037 ...")
    (public-key elgamal-pkcs1 "\003\017v\245\235b\004 ...
\177.\312\357")
      "\003\017l\314\254\227 ..."))

(signed (cert
  (issuer (public-key elgamal-pkcs1 "\003\017v\245\235b\004 ...
E\024\252B{ [e"))
  (subject (public-key elgamal-pkcs1 "\003\017v\245\235b\004 ...
\177.\312\357"))
  (tag (net.espeak.method Trigonometric (* set sine cosine tangent)
(Mathematical)))
  (not-before 2000-05-30_20:56:52)
  (not-after 2002-05-30_20:56:52)
  ) (signature (hash SHA-1 " \b\253\225\037W\"0\351\352 ...")
    (public-key elgamal-pkcs1 "\003\017v\245\235b\004 ... E\024\252B{ [e")
      "\003\017PZ\031\235g!a ..."))
)

!!! Summary:
!!! =====
!!! Signed objects      = 2
!!! Unsigned objects   = 0
!!! Name Certificates  = 0
!!! Attr. Certificates = 0

```

Security Examples

From a security point of view, your main concern is the metadata and resource masks you set up for your resources. When security is enabled, authorization is required for all operations. Masks control this behavior. You set a mask, operations matching the mask are permitted whether the requestor is authorized or not.

Two masks are available:

- Metadata mask for metadata operations
- Resource mask for resource-specific operations

Masks are specified as tags. The basic method tag format is:

```
(net.espeak.method <interface name> <method name>)
```

In the metadata mask, the interface name is the specified core interface; the method name is the operation in that interface. For metadata, the interface is always `ResourceManipulationInterface` and the method name is one of its methods.

In the resource mask for a J-ESI service, the interface name is the fully-qualified name of the interface class. The method name is the name of the interface method plus the argument types. This allows you to distinguish overloaded methods.

The in-core metaresource uses the metadata mask when performing metadata operations. The Core passes the resource mask to the service handler to use when performing operations on the service.

The masks are general tags, so the mask tag or any of its fields may use the tag-matching features such as sets, prefixes, and ranges. The interface and method names, for example, do not have to be string literals; they can be sets or prefixes. Refer to the *e-speak Architectural Specification* for information on tag formats.

This tag masks method `foo()` in interface `net.espeak.examples.ExampleIntf`:

```
(net.espeak.method net.espeak.examples.ExampleIntf foo())
```

This tag masks all methods beginning with `foo`:

```
(net.espeak.method net.espeak.examples.ExampleIntf (* prefix foo))
```

This tag masks methods `foo` and `bar`:

```
(net.espeak.method net.espeak.examples.ExampleIntf (* set foo()  
bar()))
```

Methods with prefix `foo` or `bar`:

```
(net.espeak.method net.espeak.examples.ExampleIntf
 (* set (* prefix foo) (* prefix bar)))
```

All methods in the interface:

```
(net.espeak.method net.espeak.examples.ExampleIntf )
```

Because missing trailing elements match anything, this is equivalent to:

```
(net.espeak.method net.espeak.examples.ExampleIntf (*))
```

Methods `foo()` in `InterfaceA` and `bar()` in `InterfaceB`:

```
(* set (net.espeak.method InterfaceA foo())
 (net.espeak.method InterfaceB bar()))
```

All methods:

```
(net.espeak.method) OR (*)
```

The full form of the method tag is:

```
(net.espeak.method <interface name> <method name> <service>)
```

When the service handler receives a message invoking an operation, it:

- Extracts the interface and method from the message
- Retrieves the service identifier from the information passed to the handler by the core
- Constructs a method tag using this data
- Queries the service authorizer to verify that the tag is authorized

The authorizer checks to see if the tag matches the resource mask, and if it does, permits the operation. If the tag does not match the resource mask, the authorizer uses the current security session to see if the tag is authorized.

Normal tag matching rules apply, which is why the service part of a mask tag was omitted in the examples. A tag is authorized at a server in the current security session if a client presents a valid certificate (or certificates) that contain the tag. A certificate is valid if it has not expired and has a valid signature. Refer to the *e-speak Architectural Specification* for more information on certificate validity and tag matching.

You can construct general tags using this method in `ESSecurityEnv`:

```
ADR createTag(String s) throws IOException
```

The `IOException` subclass `net.espeak.security.adr.ADRParseException` is thrown on a parse error. The parameter `s` is a string containing the input syntax for the tag.

You can construct method tags using:

```
ADR createMethodTag(String interfaceName, String methodName, ADR
    service)
```

For resource masks, tags normally contain `(*)` as the service parameter. In advanced applications, the service can optionally set the service parameter to its service ID.

The `ESAbstractElement` methods use mask tags:

```
void setResourceMask(ADR tag) throws ESEException
void setMetadataMask(ADR tag) throws ESEException
```

For non-registered services, these simply affect the local state. After registration, the methods set the local state and update the service metadata.

Turn masking on or off using `ESAuthorizer`:

```
void setMasking(Boolean x)
```

When masking is off, the service authorizer ignores the resource mask, even if it was previously set. Turning masking off in the authorizer has no effect on the resource metadata or the in-core metaresource handling metadata operations. You can turn masking off completely in the core and handler by setting a mask to null.

`ESConnection` includes methods for controlling the default resource and metadata masks used when services are registered:

```
void setDefaultResourceMask(ADR mask)
ADR getDefaultResourceMask()
void setDefaultMetadataMask(ADR mask)
ADR getDefaultMetadataMask()
void setMasks(ADR metadataMask, ADR resourceMask)
```

After you set a default mask, all registered resources use it until it is changed. Unless you set the default masks explicitly, `ESConnection` considers them null and requires authorization for all operations.

Bootstrap Process for Testing

NOTE: When writing and deploying secure applications, refer to the J-ESI documentation and the *e-speak Architectural Specification*.

Use this configuration for testing purposes only. In a live deployment, all entities (users) or services that must be distinguished for access control must have separate PSEs. Sharing a PSE is comparable to sharing a login password to a Unix or Windows NT system.

In this section, the subject of the certificate is the entity being authorized. For example, if the issuer is the Core, the subject is the client, and the certificate contains the tag attribute (`net.espeak.method (*) (*)`), the core is issuing a certificate authorizing the client to access any method within any service in the Core.

In the PSE Manager:

- 1** Generate a keystore object (for example, a Private Secure Environment) called `securestore.bin`.

Because it is shared by all participants — the core, the client, and the service — this configuration is *not* distributed.

- 2** Generate three different key pairs, one for each participant, within the same PSE with the labels:

- `client`
- `core`
- `service`

- 3** Generate a basic attribute certificate for each pair of distinct participants (for example, the client as issuer, core as subject, and so on for all nine distinct combinations). This includes those roles generating certificates for themselves (the issuer is the same as the subject that gives each participant arbitrary permission to perform operations). Each certificate contains the e-speak tag attribute:

```
(net.espeak.method (*) (*)
```

- 4** Set the validity dates of the certificates to span the entire testing period.
- 5** Issue the certificate. It must be signed by the issuer (client, core, or service) using the PSE Manager.

To operate the core with security enabled, load a security configuration file that contains the appropriate attributes.

The default configuration file is `espeak.cfg`, which may be located in the:

- Config directory in `<e-speak home>` as defined by the property `espeak_home`
- Directory specified by property `net.espeak.util.config.path`
- Current directory (from the system property `user.dir`), if the property is not set
- Directory specified by the `user.home` system property as a system resource from the classpath

Simplified `espeak.cfg` file

```

!=====
! $Id: espeak.cfg,v 1.1.2.1.4.10 2000/05/16 07:17:42 ks Exp $
!=====
! E-speak security properties file.
!=====

!-----
! Security properties.
!-----

! Master flag controlling whether security is on or off.
! net.espeak.security.activate=on
! This property has been deprecated.

! Default name of the keystore file
!net.espeak.security.pse.storefile=securestore.bin

! Gui mode runs a dialog for the passphrase.
!net.espeak.security.pse.mode=gui
! Passphrase mode looks for the passphrase property.
!net.espeak.security.pse.mode = passphrase
! Passfile mode looks for a file containing the passphrase property.

```

```
!net.espeak.security.pse.mode = passfile

! Define the passphrase.
!net.espeak.security.pse.passphrase = default passphrase

! Define the default role (i.e. the default PSE key label).
!net.espeak.security.pse.role = client
```

Appendix C Uninstalling E-speak

Windows NT

To uninstall e-speak:

- 1** Select **Start > Programs > e-speak**.
- 2** Select **Uninstall e-speak**.
- 3** Restart the computer.
- 4** Verify that the e-speak reference has been removed from **Programs** in the **Start** menu.
- 5** Verify that the `\Program Files\e-speak` directory is empty.
- 6** Verify that you cannot start e-speak.
- 7** Verify that the **EspeakProcessManager** is no longer listed in the Windows **Services** window.
- 8** Verify that none of these processes exist in the Task Manager:

`java.exe`

`espeak.exe`

`ESPM.exe`

NOTE: You may have Java programs other than e-speak running.

Linux

To uninstall e-speak:

- 1 Log in as `root`.
- 2 Query all of the installed packages using:

```
rpm -qa
```
- 3 Select the package that you want to uninstall.
- 4 Type this rpm command to uninstall e-speak:

```
rpm -e <e-speak package name>
```
- 5 Check for any e-speak directories or files that were not deleted and remove them manually.

HP-UX

To uninstall e-speak:

- 1 Log in as `root`.
- 2 Type `swremove`.
- 3 Select **e-speak** from the de-installation menu.

NOTE: For more information, refer to the *HP-UX™ Administrator's Guide*.