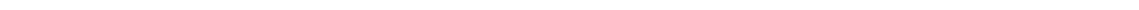


Red Hat Fortitude 1.0.0

Deployment Guide



Red Hat Fortitude 1.0.0: Deployment Guide

Copyright © 2007 Red Hat, Inc.

This manual covers all aspects of installing and configuring Red Hat Fortitude 1.0.0 to run and manage secure web services.

1801 Varsity Drive
Raleigh, NC 27606-2072
USA
Phone: +1 919 754 3700
Phone: 888 733 4281
Fax: +1 919 754 3701
PO Box 13588
Research Triangle Park, NC 27709
USA

Documentation-Deployment

Copyright © 2007 by Red Hat, Inc. This material may be distributed only subject to the terms and conditions set forth in the Open Publication License, V1.0 or later (the latest version is presently available at <http://www.opencontent.org/openpub/>).

Distribution of substantively modified versions of this document is prohibited without the explicit permission of the copyright holder.

Distribution of the work or derivative of the work in any standard (paper) book form for commercial purposes is prohibited unless prior permission is obtained from the copyright holder.

Red Hat and the Red Hat "Shadow Man" logo are registered trademarks of Red Hat, Inc. in the United States and other countries.

All other trademarks referenced herein are the property of their respective owners.

The GPG fingerprint of the security@redhat.com key is:

CA 20 86 86 2B D6 9D FC 65 F6 EC C4 21 91 80 CD DB 42 A6 0E

Table of Contents

Introduction	vi
1. Document Conventions	vi
2. Send in Your Feedback	vii
1. What is Fortitude?	1
2. Installation	2
1. Prerequisites	2
1.1. Required Libraries	2
1.2. Required JRE	2
1.3. Required JDK	4
2. The Fortitude Installation	5
3. Modes of Operation	6
4. Starting and Stopping the Server	8
5. Configuration	10
1. Directory Layout	10
1.1. Fortitude Directory Listing	10
2. Apache HTTP Server SSL Configuration	10
2.1. Introduction	10
2.2. Certificate Generation	10
2.3. Configuration Directives	11
2.4. Environment Variables	20
2.5. Database Management	20
2.6. Importing OpenSSL Certificates	20
2.7. Why SSLv2 is Disabled	21
3. Setting up CRL Revocation	21
3.1. CRLFile Attribute Examples	23
4. Example CRL Revocation File	23
5. Tomcat SSL Configuration	23
5.1. Introduction	23
5.2. Setting up a Connector in Tomcat	23
6. Troubleshooting and FAQ	26
6. Migrating from NES	27
1. Introduction	27
2. Migrating the NES Configuration	27
2.1. Unprocessed Directives	27
2.2. A Simple Migration Session	28
2.3. A More Complex Migration Session	29
3. Migrating Web Applications	30
4. Migrating Access Controls	30
4.1. A Sample ACL Migration	31
5. Migrating CGIs	34
6. Other Manual Migration Tasks	35
7. Known Issues	36
8. Migration Warning Messages	36
7. Additional Resources	39
I. Appendixes	40

A. Fortitude 1.0.0 Installation Components	41
B. Unsupported Apache mod_ssl Directives	42
C. List of Available Ciphers	43
D. CGI and SSI Environment Variables	44

Introduction

Welcome to the *Fortitude Deployment Guide*.

This guide contains information on how to install and configure Fortitude on Red Hat Enterprise Linux and Solaris.

This guide assumes you have a basic understanding of your Red Hat Enterprise Linux system. If you need help installing Red Hat Enterprise Linux, refer to the *Red Hat Enterprise Linux Installation Guide*.

1. Document Conventions

Certain words in this manual are represented in different fonts, styles, and weights. This highlighting indicates that the word is part of a specific category. The categories include the following:

Courier font

Courier font represents `commands`, `file names` and `paths`, and `prompts`.

When shown as below, it indicates computer output:

```
Desktop      about.html   logs         paulwesterberg.png
Mail         backupfiles  mail         reports
```

Courier font

Bold Courier font represents text that you are to type, such as: `service jonas start`

If you have to run a command as root, the root prompt (`#`) precedes the command:

```
# gconftool-2
```

italic Courier font

Italic Courier font represents a variable, such as an installation directory: `install_dir/bin/`

bold font

Bold font represents **application programs** and **text found on a graphical interface**.

When shown like this: **OK**, it indicates a button on a graphical application interface.

Additionally, the manual uses different strategies to draw your attention to pieces of information. In order of how critical the information is to you, these items are marked as follows:





Note

A note is typically information that you need to understand the behavior of the system.



Tip

A tip is typically an alternative way of performing a task.



Important

Important information is necessary, but possibly unexpected, such as a configuration change that will not persist after a reboot.



Caution

A caution indicates an act that would violate your support agreement, such as recompiling the kernel.



Warning

A warning indicates potential data loss, as may happen when tuning hardware for maximum performance.

2. Send in Your Feedback

If you find an error in the *Fortitude Deployment Guide*, or if you have thought of a way to make this manual better, we would like to hear from you! Submit a report in Bugzilla (<http://bugzilla.redhat.com/bugzilla/>) against the Product `Red Hat Fortitude` and the component `Documentation`.

If you have a suggestion for improving the documentation, try to be as specific as possible. If you have found an error, include the section number and some of the surrounding text so we can find it easily.

Chapter 1. What is Fortitude?

Fortitude is an overarching term that describes a set of web-based, cryptographic security products that use the Network Security Services (NSS) libraries. These products, delivered as plugins for the Apache HTTP Server and the Tomcat servlet container, allow the operator to replace existing cryptographic services using OpenSSL with NSS.

The *Secure Sockets Layer* (SSL) is provided by NSS instead of OpenSSL. In Apache, this functionality is provided by `tomcat_jss` and Network Security Services for Java (JSS) for Tomcat.

Supported Platforms. Fortitude 1.0.0 is currently supported on the following platforms:

- Red Hat Enterprise Linux AS 4 for i386
- Red Hat Enterprise Linux ES 4 for i386
- Red Hat Enterprise Linux AS 4 for AMD64 and Intel EM64T
- Red Hat Enterprise Linux ES 4 for AMD64 and Intel EM64T
- Sun Solaris SPARC 9 64-bit

Features. Fortitude is designed to provide an easy migration path from an existing secure web service. It is an open-source project and supports a wide range of standards and features:

- Federal Information Processing Standards (FIPS) 140-1 certified cryptography (140-2 certification is in progress)
- SSLv3 and TLSv1
- RSA support
- Client certificate authentication
- Hardware accelerators
- Certificate Revocation Lists (CRLs)
- Online Certificate Status Protocol (OCSP)
- Public Key Cryptography Standard (PKCS) #11

For more information on the Apache HTTP Server, refer to <http://httpd.apache.org/>.

For more information on the Tomcat servlet container, refer to <http://tomcat.apache.org/index.html>.

Chapter 2. Installation

The Fortitude Web Server is best installed from RPMs using a specially-prepared script. This script is prepared on a per-platform basis and ensures that the required packages, including the Tomcat components, are installed in the correct order.



Note

The Fortitude installation is not relocatable. You cannot move the Fortitude files to a directory other than where they were installed.

1. Prerequisites

To install Fortitude you must have a working Apache and Tomcat installation. Apache and Tomcat are included as part of Red Hat Enterprise Linux. On Solaris they are provided separately.

Refer to [Chapter 7, Additional Resources](#) for more information on these applications.

1.1. Required Libraries

Fortitude relies on the following libraries:

1.2. Required JRE

Java 1.5.0 Java Runtime Environment (JRE), Update 8 or later. Fortitude does not support earlier versions of the JRE. This JRE is required for running Tomcat.

On 32-bit Red Hat Enterprise Linux 4 platforms, Fortitude requires the 32-bit version of the IBM JRE. A pre-packaged binary distribution of this package is available through either of the following Red Hat Network channels:

- Red Hat Enterprise Linux AS (v. 4 for x86) Extras
- Red Hat Enterprise Linux ES (v. 4 for x86) Extras

This binary is available as `java-1.5.0-ibm-1.5.0.0-1jpp_2rh:0.i386`.

On 64-bit Red Hat Enterprise Linux 4 platforms, Fortitude requires the 64-bit version of the IBM JRE. A pre-packaged binary distribution of this package is available through either of the following Red Hat Network channels:

- Red Hat Enterprise Linux AS (v. 4 for AMD64/EM64T) Extras
- Red Hat Enterprise Linux ES (v. 4 for AMD64/EM64T) Extras

1.2. Required JRE

This binary is available as `java-1.5.0-ibm-1.5.0.0-1jpp_2rh:0.x86_64`.



Note

The JRE is not included as part of the distribution. You must download and install the appropriate JRE before you install Fortitude.

As root, run `/usr/sbin/alternatives --config java` to ensure that the IBM Java 1.5.0 JRE is selected.



Warning

Both the 32-bit xSeries (Intel-compatible) and 64-bit AMD/Opteron/EM64T versions of the IBM J2SE JRE 5.0 RPM packages available through the IBM download site are packaged in a format which is incompatible with Fortitude.

On 64-bit Solaris 9 (SPARC) platforms, Fortitude requires the latest version of the 64-bit Sun J2SE Java Runtime Environment 5.0 (Update 8 or later). This is available from the Sun JDK 5 download site, http://java.sun.com/javase/downloads/index_jdk5.jsp.



Important

The 64-bit Solaris version of Fortitude requires the 32-bit version of the JRE as well as the 64-bit version. The 32-bit version is used for the applet and Java Web Start support. Before you install Fortitude, you should refer to the following:

- <http://java.sun.com/j2se/1.5.0/README.html>
- <http://java.sun.com/j2se/1.5.0/ReleaseNotes.html>
- <http://java.sun.com/j2se/1.5.0/jre-install-solaris-64.html>

Under the section Java Runtime Environment (JRE) 5.0 Update 9, Sun only makes this JRE available through a self-extracting file. This file is incompatible with Fortitude because the format does not use the native Solaris packaging utility database.

To obtain the Sun 5.0 JRE in a compatible format, use the following procedure:

1. Click **Download** under the JDK 5.0 Update 9 section
2. Under Solaris SPARC Platform - J2SETM Development Kit 5.0 Update 9, **select** Solaris SPARC 32-bit packages - tar.Z (`jdk-1_5_0_09-solaris-sparc.tar.Z`) **and** Solaris SPARC

1.3. Required JDK

64-bit packages - tar.Z (use 32-bit version for applet and Java Web Start support)
(jdk-1_5_0_09-solaris-sparcv9.tar.Z).

3. Uncompress the files and extract the contents to a suitable directory..

The 32-bit file, `jdk-1_5_0_09-solaris-sparc.tar.Z`, contains the following:

- COPYRIGHT
- LICENSE
- README.html
- SUNWj5cfg
- SUNWj5dev
- SUNWj5dmo
- SUNWj5jmp
- SUNWj5man
- SUNWj5rt

The 64-bit file, `jdk-1_5_0_09-solaris-sparcv9.tar.Z`, contains the following:

- SUNWj5dmx
- SUNWj5dvx
- SUNWj5rtx

Because only the JRE is needed on Solaris 9 systems, use the `pkgadd` utility to add the 32-bit package, `SUNWj5rt`, and then add the 64-bit package, `SUNWj5rtx`.

1.3. Required JDK

A Java Development Kit (JDK) is required if you plan to use uncompiled Java Server Pages (JSPs). The JDK should match the installed JRE.

Refer to http://kbase.redhat.com/faq/FAQ_54_4667.shtm for more information.



Note

Fortitude has not been tested with JDK 6.0.

On 32-bit Red Hat Enterprise Linux 4 platforms, a pre-packaged binary distribution of the 32-bit version of the IBM JDK 1.5.0 is available through either of the following Red Hat Network channels:

- Red Hat Enterprise Linux AS (v. 4 for x86) Extras
- Red Hat Enterprise Linux ES (v. 4 for x86) Extras

This binary is available as `java-1.5.0-ibm-devel-1.5.0.0-1jpp_2rh:0.i386`.

On 64-bit Red Hat Enterprise Linux 4 platforms, a pre-packaged binary distribution of the 64-bit version of the IBM JDK 1.5.0 is available through either of the following Red Hat Network channels:

- Red Hat Enterprise Linux AS (v. 4 for AMD64/EM64T) Extras
- Red Hat Enterprise Linux ES (v. 4 for AMD64/EM64T) Extras

This binary is available as `java-1.5.0-ibm-devel-1.5.0.0-1jpp_2rh:0.x86_64`.

After installing the JDK, run `/usr/sbin/alternatives --config javac` as root to insure that a JDK is available.

Solaris 9 systems do not require that you download and install a JDK. You may, however, need to download and install the Sun JDK 5.0 package in order to obtain a compatible Sun JRE 5.0 package.



Note

In order to use the JDK instead of the JRE on Solaris, you need to modify the `/etc/java/java.conf` file. Any references to `/usr/java/jre` should be replaced with `/usr/java`.

2. The Fortitude Installation

The Fortitude installation creates a separate instance of Apache and Tomcat using customized configuration files. The new configuration uses `mod_nss` as the SSL provider. These configuration files are discussed further in [Section 1, "Directory Layout"](#).



Tip

The `mod_nss` package generates a self-signed server certificate with the default password `httpstest`. This password is automatically set in `/opt/fortitude/conf/password.conf` to allow for unattended startup.

Chapter 3. Modes of Operation

Fortitude can be used in a variety of ways depending on your needs. You can run Fortitude in any of the configurations listed below, using secured listeners, unsecured listeners, or both:

- With Apache, with `mod_nss` running in front of Tomcat (the default configuration)
- With Apache on its own, with `mod_nss`
- With Tomcat on its own, with `tomcatjss`



Note

If you only want to run Apache and/or Tomcat unencrypted, and you are not migrating an application from NES, you are probably better off using the default Apache and Tomcat instances that ship with Red Hat Enterprise Linux. This option is not available on Solaris.

On Red Hat Enterprise Linux, Fortitude runs as the user "apache". On Solaris, it runs as the user "nobody". In both cases, Fortitude listens on ports 80 and 443 by default. Tomcat listens only on the AJP connector on port 8009.



Note

You cannot run both the `httpd` service and the `fortitude` service at the same time without changing the port of at least one of the services.

To change the port number for the `httpd` service, edit the `Listen` entry in the `/etc/httpd/conf/httpd.conf` file.

To change the port number for the `fortitude` service, edit the `Listen` entry in the `/opt/fortitude/conf/httpd.conf` file.

To run Tomcat as a stand-alone service, uncomment the JSS entry in `/opt/fortitude/conf/server.xml` and set the port as desired. You also need to modify the start and stop functions in the `/etc/init.d/fortitude` file.

Procedure 3.1. To modify the start and stop functions for Apache:

1. Open the `/etc/init.d/fortitude` file and locate the `start()` function.
2. Comment the following line (precede it with a hash (#) mark): `daemon ${fortitude_root}/start`

3. Locate the `stop()` function, and comment the following line: `daemon ${fortitude_root}/stop`
4. Save and close the file.

Refer to [Section 5, “Tomcat SSL Configuration”](#) for more information on configuring Tomcat to run as a stand-alone service.

Chapter 4. Starting and Stopping the Server

This section describes how to start and stop the Fortitude server.

Starting the Web Server. The procedure for starting a Fortitude server is similar to that for starting a standard Apache Web server. To start the server, you need to authenticate yourself to the security token; that is, you need to enter the password for the associated key.

Use the following command to start the Fortitude server:

```
[root@server93 ~]# service fortitude start
```

If you want to start the Apache Web server and the Tomcat servers separately, use the following commands:

```
[root@server93 ~]# /opt/fortitude/start  
[root@server93 ~]# /opt/fortitude/dtomcat5
```

The first command only starts the web server. The second command only starts the Tomcat server.



Tomcat can take some time to start. Allow sufficient time for the Tomcat server to start before performing other operations that rely on this server.

If you require that the PIN be entered each time the server is started, you need to modify the NSS configuration file. Locate the following section in the `/opt/fortitude/conf.d/nss.conf` file:

```
NSSPassPhraseDialog file:/opt/fortitude/conf/password.conf
```

Change it to read as follows:

```
NSSPassPhraseDialog builtin
```

If you have additional hardware tokens, you are prompted for each token password.



Note

If you make this change, you cannot use the init scripts to start the server. Instead, you must use the start script in `/opt/fortitude/`. This also prevents unattended startup.



Note

There are currently no specific SELinux policy rules defined for use with Fortitude. On machines with SELinux enabled, the web server runs in the `unconfined_t` domain. This is achieved by prepending the web server start command with the following SELinux directive:

```
runcon -t unconfined_t
```

Stopping the Web Server. Use the following command to stop both the Apache Web server and the Tomcat server:

```
[root@server93 ~]# service fortitude stop
```

If you want to stop the Apache Web server and the Tomcat servers separately, use the following commands:

```
[root@server93 ~]# /opt/fortitude/stop  
[root@server93 ~]# /opt/fortitude/dtomcat5 stop
```

The first command only stops the web server. The second command only stops the Tomcat server.

For a list of available options for the `fortitude service` command, type `service fortitude` in a command shell.

Chapter 5. Configuration

This section provides configuration details for Fortitude. The basic configuration of the Apache HTTP Server and Tomcat are not covered here. Refer to [Chapter 7, Additional Resources](#) for more information on these topics.

1. Directory Layout

Fortitude is installed in `/opt/fortitude/`. The configuration files are stored in the `/opt/fortitude/conf/` subdirectory. Additional configuration files, used by the Apache Web server for third party modules such as PHP, are stored in `/opt/fortitude/conf.d/`.



Important

If you install a new Apache Web server plugin that includes a configuration file, that file is not automatically copied to the appropriate Fortitude directory. You must manually copy the new configuration file from `/etc/httpd/conf.d/` to the `/opt/fortitude/conf.d/` directory.

1.1. Fortitude Directory Listing

The Fortitude installation directory contains the following sub-directories:

2. Apache HTTP Server SSL Configuration

2.1. Introduction

The standard Apache distribution uses `mod_ssl`, which is based on the `OpenSSL` library. The `mod_ssl` package was created in April 1998 by Ralf S. Engelschall (<mailto:rse@engelschall.com>). This was originally derived from the Apache-SSL package developed by Ben Laurie (<mailto:ben@algroup.co.uk>) and is available under the Apache 2.0 license.

Fortitude uses `mod_nss`, an Apache module that uses NSS instead of `OpenSSL`. It is a direct conversion from the standard `SSL` module that Apache uses, `mod_ssl`.

The configuration file can be found in `/opt/fortitude/conf.d/nss.conf`.

2.2. Certificate Generation

Fortitude includes a shell script, `/opt/fortitude/bin/gencert`, which can automatically generate a self-signed *Certificate Authority* (CA) plus one server certificate. This is adequate for testing purposes but it is strongly recommended that a genuine server certificate be obtained from a recognized CA before moving a `mod_nss` server into production. Users should be expected to cancel any request to a secure server signed by an unknown issuer.

2.3. Configuration Directives

The command `gencert` takes one argument; the path to the location of the certificate database. This command displays the output of the operation, but this is not essential information for most administrators.



Tip

The `mod_nss` package generates a self-signed server certificate with the default password `httpptest`. This password is automatically set in `/opt/fortitude/conf/password.conf` to allow for unattended startup.

The following is an example output from the above command. (Excess output is trimmed for display purposes.)

```
mkdir /opt/fortitude/alias
./gencert /opt/fortitude/alias
#####
Generating new server certificate and key database. The
password is httpptest
#####
#####
Generating self-signed client CA certificate
#####
#####
Generating key. This may take a few moments...
...
```

Example 5.1. Output from `gencert`

The following files should now exist:

- `/opt/fortitude/alias/cert8.db`
- `/opt/fortitude/alias/key3db`
- `/opt/fortitude/alias/secmod.db`

These files make up an NSS certificate database.

2.3. Configuration Directives

You need to specify a number of directives to properly configure `mod_nss`. Many of these directives are similar to or replace the `mod_ssl` directives used by Apache, while others have no meaning in the `mod_nss` environment.

Refer to [Appendix B, Unsupported Apache `mod_ssl` Directives](#) for a list of directives not supported by `mod_nss`:

The following sections describe directives that apply to `mod_nss`. The format of these directives is

2.3. Configuration Directives

`name value.`

2.3.1. NSSPassPhraseDialog

Authentication is required in order to use the private key in an NSS certificate database. Use the `NSSPassPhraseDialog` directive to specify the method of this authentication. This directive accepts one of the following arguments:

`builtin`

Specifies that the system should prompt for the token password for each cryptographic device. This works seamlessly with any hardware tokens used. The default device is the "internal" token provided by the NSS Certificate database itself.



Note

It is not mandatory to use a password for an NSS database. To modify an existing NSS database to have no password, use the following command:

```
[root@dhcp-91 ~]# modutil -dbdir /opt/fortitude/alias -changepw <NSS Certificate Database>
```

This command prompts you to enter a password. Press **Enter** for no password. Press **Enter** again to confirm no password.

If you are not using a password with your NSS database, the `builtin` option is recommended.

`file:/path/to/passwordfile`

Specifies that the system should read the password from the specified file. The token passwords can be stored in an ASCII text file which is read during startup. This enables the server to start without user intervention. You need to specify the full path to the file, in the following format: `file:/path/to/passwordfile`.

The format of the password file is as follows:

```
token:password
```

For example, the internal token might be:

```
internal:secret12
```

2.3.2. NSSPassPhraseHelper

When the Apache Web server is started, it performs some preliminary housekeeping tasks, detaches itself from the terminal and then creates several child processes.

During this preliminary housekeeping, all of the modules specified by the `LoadModule` directive in `httpd.conf` are loaded. Each module is initialized as it is loaded, and any of the module's configuration directives are identified.

At this point, Apache detaches itself from the terminal. That is, `stdin`, `stdout`, and `stderr` are all closed. This prevents the system from prompting for the NSS token passwords. Because the NSS module is loaded and unloaded, the NSS certificate database also needs to be loaded and unloaded. This means that any previously entered PINs are lost, which leaves Apache in a non-working state.

The solution to losing these PINs is the `PassPhraseHelper`. This is a stand-alone program that also opens the NSS certificate database and stores a copy of the encrypted token password that was entered during the first load of the NSS module. When `mod_nss` needs to open the certificate database during subsequent reloads, it queries the `PassPhraseHelper` for the token password.

The format of this directive is as follows:

```
NSSPassPhraseHelper /opt/fortitude/bin/nss_pcache
```

2.3.3. NSSCertificateDatabase

Specifies the path to the required NSS certificate database. An NSS certificate database consists of three files:

- `cert8.db` — stores certificates and *Certificate Revocation Lists* (CRLs)
- `key3.db` — stores keys
- `secmod.db` — stores information about available PKCS#11 modules

```
NSSCertificateDatabase /opt/fortitude/alias
```



Note

This directive specifies a path, not a filename.

2.3.4. NSSSessionCacheSize

Specifies the number of `SSL` sessions that can be cached. The default value is 10000. There is no upper limit.

2.3.5. NSSSessionCacheTimeout

Specifies the number of seconds `SSLv2` sessions are cached. The valid range is 5 - 100 seconds. A setting outside the valid range is silently constrained. The default value is 100.

2.3.6. NSSSession3CacheTimeout

Specifies the period for which `SSLv3` sessions are cached. This value is in seconds, and the valid range is from 5 to 86400. A setting outside the valid range is silently constrained. The default value is 86400 (24 hours).

2.3.7. NSSRandomSeed

Configures sources to seed the NSS *Random Number Generator* (RNG) upon activation. This is the only seeding of the RNG currently available.



Note

NSS has its own RNG that it automatically uses when a random number is needed. Consequently, seeding is not explicitly required.

In contrast to `mod_ssl`, you cannot set up per-connection seeding with NSS because NSS already does this for you.

The following sources are available:

`builtin`

Combines the current system time, the current process ID and a randomly chosen 128-byte extract of the process stack. This is not a particularly strong source of entropy.

```
NSSRandomSeed startup builtin
```

`file:/path/to/source`

Reads from the specified file. You can also specify the number of bytes to read. The byte value has no minimum or maximum constraints, but it must be a positive integer.

Be aware that `/dev/random` is a blocking read. If you request more bytes than what are currently available, the server will pause or hang until that data is available.

An alternative to `/dev/random` is the non-blocking device `/dev/urandom`. This device returns whatever data it has available and allows the startup process to continue. The disadvantage of using this device is that it may not provide the required level of entropy.

```
NSSRandomSeed startup /dev/urandom 512
```

2.3. Configuration Directives



Warning

If you use the `file:` method to specify the seed source for the RNG and you do not specify a `bytes` argument, `NSSRandomSeed` reads the entire file. If you specify `/dev/*random` as the input file, then the read operation will continue indefinitely.

For more information on these devices and random number generation, refer to www.linux.com/howtos/Secure-Programs-HOWTO/random-numbers.shtml [<http://www.linux.com/howtos/Secure-Programs-HOWTO/random-numbers.shtml>].

`exec: /path/to/program`

Executes the given program and uses `stdout` as the entropy. If the `bytes` argument is included, it reads that many bytes, otherwise it reads until the program exits.

```
NSSRandomSeed startup /usr/bin/makerandom
```

2.3.8. NSSEngine

Enables or disables the `SSL` protocol. This is usually used within a `VirtualHost` tag to enable `SSL` for a particular virtual host.

`SSL` is disabled by default.

2.3.9. NSSFIPS

Enables or disables `FIPS 140` mode.

This is the FIPS-certified module and replaces the standard internal PKCS#11 module. It also forces the enabled protocols to `TLSv1` and disables all except FIPS ciphers. You can still specify which ciphers to use, but only FIPS-certified ciphers are enabled. Any non-FIPS ciphers that are included in the `NSSCipherSuite` entry are automatically disabled.

FIPS is disabled by default.

The available ciphers are as follows:

- `rsa_3des_sha`
- `rsa_des_sha`
- `fips_3des_sha`
- `fips_des_sha`
- `rsa_des_56_sha`
- `fortezza`

2.3. Configuration Directives

2.3.10. NSSOCSP

Enables or disables *Online Certificate Status Protocol* (OCSP). This allows the server to check the validity of a client certificate before accepting it.

OCSP is disabled by default.

Refer to <http://www.ietf.org/rfc/rfc2560.txt?number=2560> for more information on OCSP.

2.3.11. NSSCipherSuite

A comma-separated list of the available `SSL` ciphers. To enable a particular cipher, prefix it with a plus sign (+). To disable a cipher, prefix it with a minus sign (-).

All ciphers are disabled by default.

Refer to [Appendix C, List of Available Ciphers](#) for a list of available ciphers.

In the following example, the backslash character (\) is used to break lines for printing purposes; the entire entry should be on one line.

```
NSSCipherSuite +rsa_3des_sha,-rsa_des_56_sha,+rsa_des_sha,-rsa_null_md5,\
-rsa_null_sha,-rsa_rc2_40_md5,+rsa_rc4_128_md5,-rsa_rc4_128_sha,\
-rsa_rc4_40_md5,-rsa_rc4_56_sha,-fortezza,-fortezza_rc4_128_sha,\
-fortezza_null,-fips_des_sha,+fips_3des_sha,-rsa_aes_128_sha,-rsa_aes_256_sha
```



Note

`SSLv2` ciphers are not supported in `mod_nss` and cannot be enabled.

This option differs from `mod_ssl` in that you can not add or subtract protocols.

2.3.12. NSSProtocol

A comma-separated list of the basic protocols that the server can use (and that clients may connect with). This list does not actually enable a cipher, but provides permissions to use the specified ciphers for that protocol.

The available options are:

- `SSLv3`
- `TLSv1`
- `All`

`SSLv2` is currently not supported.

2.3.13. NSSNickname

Specifies the nickname to use for this server certificate.

Certificates stored in an NSS database are referred to using nicknames, which makes accessing a specific certificate much easier. It is possible to specify the certificate *Distinguished Name* (DN), but it is easier to use a nickname. If you use a nickname that includes spaces then you need to enclose the nickname in double quotes.

Refer to the following examples:

```
NSSNickname Server-Cert
NSSNickname "This contains a space"
```

2.3.14. NSSEnforceValidCerts

By default `mod_nss` does not start if the server certificate is not valid. This means that if the certificate has expired or is signed by a *Certificate Authority* (CA) that is not trusted in the NSS certificate database, the server does not start. If you want the server to start even if the certificate is not valid, add this directive to the `nss.conf` file. This causes the server to start but to also issue a warning about the invalid certificate.



Caution

It is not good security practice to allow invalid certificates. You should always enforce valid server certificates, especially on production systems.

2.3.15. NSSVerifyClient

Determines whether *Client Certificate Authentication* will be requested or required. You can set this on a per-server or per-directory context. At the server level, the certificate is requested during the initial `SSL` handshake. In the per-directory context, an `SSL` renegotiation is required and a certificate requested from the client.

The available options are:

`none`

No client certificate is required or requested.

`optional`

A certificate is optional. A client certificate is requested but if one is not available, the connection may continue.

`require`

A valid client certificate is required for the connection to continue.

The `mod_ssl` option `option_no_ca` is not supported.



Note

There is no `NSSVerifyDepth` directive. NSS always verifies the entire certificate chain.

2.3.16. NSSUserName

Defines the field in the client certificate that sets the user field in the request. The `FakeBasicAuth` option must also be set for this to work correctly. (Refer to [Section 2.3.17, “NSSOptions”](#).)

```
NSSUserName SSL_CLIENT_S_DN_UID
```

2.3.17. NSSOptions

Control various options on a per-server or per-directory context.

`FakeBasicAuth`

When this option is enabled and `NSSUserName` is set, the certificate attribute defined in `NS-UserName` is used to populate the value of `r->user` in the Apache request object. This equates to the environment variable `REMOTE_USER`.

`StdEnvVars`

A standard set of `SSL` environment variables is created.

`CompatEnvVars`

A `no-op`, that is, a computer operation that does nothing. This is allowed for backwards compatibility to `mod_ssl`.

`ExportCertData`

Several additional environment variables are created: `SSL_CLIENT_CERT`; `SSL_CLIENT_CERT_CHAIN[0..n]`; and `SSL_SERVER_CERT`. This provides additional certificate information on the client and server to the environment, plus every CA certificate in the client certificate.

`StrictRequire`

Absolutely forces the connection to be forbidden when `NSSRequireSSL` or `NSSRequire` aren't met.

`OptRenegotiate`

Allows the `SSL` connection to be renegotiated using a different configuration. This is designed for a per-directory and is CPU-intensive. For example, it can be used to force very strong ciphers in particular directories.

All options are disabled by default.

2.3. Configuration Directives

```
NSSOptions +FakeBasicAuth
<Files ~ "\.(cgi|shhtml)$">
NSSOptions +StdEnvVars
<Files>
```

2.3.18. NSSRequireSSL

The request is forbidden unless the connection is using `SSL`. Only available in a per-directory context. This directive takes no arguments.

2.3.19. NSSRequire

Provides a regular expression-based, access-control mechanism. Access may be restricted or allowed based on any number of variables, such as components of the client certificate, the remote IP address, etc.

The following example is from a typical `mod_nss.conf` file:

```
<Location />
NSSRequire (    %{SSL_CIPHER} !~ m/^(EXP|NULL)/ \
    and %{SSL_CLIENT_S_DN_O} eq "Snake Oil, Ltd." \
    and %{SSL_CLIENT_S_DN_OU} in {"Staff", "CA", "Dev"} \
    and %{TIME_WDAY} >= 1 and %{TIME_WDAY} <= 5 \
    and %{TIME_HOUR} >= 8 and %{TIME_HOUR} <= 20      ) \
    or %{REMOTE_ADDR} =~ m/^192\.76\.162\.[0-9]+$/
</Location>
```

2.3.20. NSSProxyEngine

Enables or disables `mod_nss` HTTPS support for `mod_proxy`.

Refer to [Section 6, "Troubleshooting and FAQ"](#) for more information on proxy requirements.

2.3.21. NSSProxyProtocol

Specifies the `SSL` protocols that may be used in proxy connections. The syntax is identical to that used for `NSSProtocol`.

Refer to [Section 6, "Troubleshooting and FAQ"](#) for more information on proxy requirements.

2.3.22. NSSProxyCipherSuite

Specifies the `SSL` ciphers available for proxy connections. The syntax is identical to that used for `NSSCipherSuite`.

Refer to [Section 6, "Troubleshooting and FAQ"](#) for more information on proxy requirements.

2.3.23. NSSProxyNickname

The nickname of the client certificate to send if the remote server requests client authentication.

2.4. Environment Variables

Depending on the `NSSOptions` configuration, you can set a number of `CGI` and `SSI` environment variables. This can be expensive in terms of CPU usage, however, and it is recommended that you only set these variables when they are required. For example, do not set these variables on a per-server basis.

Refer to [Section 2.3.17, "NSSOptions"](#) for information on available NSS options.

Refer to [Appendix D, CGI and SSI Environment Variables](#) for more information on the available CGI and SSI variables.

2.5. Database Management

NSS stores its certificates and keys in a set of files referred to as the "certificate database." The NSS database also stores any `CRL` s. For NSS 3.x, the default files are `cert8.db`, `key3.db` and `secmod.db`.

Refer to the NSS documentation at <http://www.mozilla.org/projects/security/pki/nss/> for more information on these files.

Several NSS tools are available for managing certificates, keys, PKCS#11 modules and CRLs. These are provided with the NSS distribution. The following is a brief overview:

The following is a list of useful commands. You can refer to certificates by either their Distinguished Name (DN) or by a nickname that is assigned when the certificate is added to the database. The nickname is the preferred method of referring to certificates.

The default location of the database files is `/opt/fortitude/alias`, but you can use the `-d` option with all of these commands to specify a different database location.

2.6. Importing OpenSSL Certificates

If you have existing `OpenSSL` certificates you can import them into an NSS certificate database.

The following is an example of importing a server certificate (nickname `server-cert`). The backslash (`\`) characters are used to indicate line breaks for printing purposes:

```
openssl pkcs12 -export -in /path/to/certificate -inkey /path/to/keyfile\  
-out server.p12 -name "Server-Cert" -passout pass:foo\  
pk12util -i server.p12 -d /path/to/database-dir/ -W foo
```

To import a CA certificate, use the following command:

```
certutil -A -n "myca" -t "CT,," -d /path/to/database-dir/ -a -i /path/to/cacertificate
```

To import a CRL, use the following command:

```
openssl crl -in /path/to/crlfile -out /tmp/crl.tmp -inform PEM -outform DER
crlutil -I -t 1 -d /path/to/database-dir/ -i /tmp/crl.tmp
```

To verify that your server certificate was imported properly, use NSS to validate it:

```
certutil -V -n Server-Cert -u V -d .
certutil: certificate is valid
```

2.7. Why SSLv2 is Disabled

All major web browsers (Firefox, Internet Explorer, Mozilla, Netscape, Opera, and Safari) support `SSLv3` and `TLS`, so there is no need for a web server to support `SSLv2`. There are some known attacks against `SSLv2` that are handled by `SSLv3/TLS`. `SSLv2` also does not support features such as client authentication.

3. Setting up CRL Revocation

You can configure `mod_revocator` to automatically download and install remote CRLs on a regular basis without restarting the server. This helps ensure that the CRLs are kept up-to-date with minimal effort. The module can also shut down the server if the CRL expires and a new one cannot be obtained.

CRL revocation options are specified in the `/opt/fortitude/conf.d/revocator.conf` configuration file.

The following options are available, and use the format: `optionName optionValue`:

*CRL*Engine on/off

Enables or disables CRL retrieval.

*CRL*UpdateCritical on/off

Shuts down the server if a CRL cannot be retrieved.

*CRL*AgeCheck on/off

Shuts down the server if a CRL expires.

This option shuts down the server if the age of a downloaded CRL exceeds the time specified in its `Next Update` field. This condition indicates that the CRL may not contain the most recent information available. To avoid the possibility of users authenticating with compromised certificates that would have been added to an up-to-date CRL, you can choose to have the server shut down automatically when a CRL is considered too old.

This check is performed when the CRL is downloaded. Therefore, an already downloaded CRL can become older than its `Next Update` time in the interval between updates and still be considered valid.



Note

This feature does not apply to CRLs that do not have a `Next Update` field.

CRLFile

A space-separated list of URLs to download and install. If multiple remote locations are listed, then the value needs to be enclosed in double-quotes. The syntax of these URLs is as follows: `protocol://urldata;update_interval;max_age`

where:

`protocol`

is one of the supported download protocols. `mod_revocator` can download CRLs using HTTP, HTTP over SSL, LDAP, and LDAP over SSL.

`urldata`

is a valid URL as required by the specified protocol.

Valid URL formats are as follows:

- `exec://path/to/program|argument1|...|argumentn`
- `http[s]://username:password@hostname:port/path[?query_string]`

`update_interval`

is the maximum time in minutes to allow between CRL downloads.

When the server starts, `mod_revocator` downloads all CRLs configured for automatic downloading. To determine when the next download should occur, `mod_revocator` uses the value of either the `update_interval` attribute or the time specified in the `Next Update` field of the CRL, whichever occurs first.

Not all CRLs include a `Next Update` field, so you must specify an update interval for each CRL. To determine an appropriate update interval, consider the network connectivity and available bandwidth at your site and how often the CRL is updated.

`max_age`

is the time in minutes that `mod_revocator` should wait past the time indicated in the CRL's `Next Update` field before determining that the CRL is too old to be valid.

To avoid unnecessary shutdowns, it is recommended that you set this value to no less than five minutes, and that you take into account possible system clock differences between the local server and the CA's CRL download server. A good starting value is 60 minutes.



Note

3.1. CRLFile Attribute Examples

If you have not enabled the `CRLUpdateCritical` option, then this option has no effect.

3.1. CRLFile Attribute Examples

The following examples show typical usage scenarios for the `CRLFile` attribute:

Using the exec URL Format.

```
exec:///opt/fortitude/bin/ldapget|ldap://ldap.example.com:3389/o=example.com?userCertificate%3bbinary?su
```

The `/opt/fortitude/bin/ldapget` program is supplied to demonstrate how this protocol works and to provide LDAP/S support. The `ldapget` command uses the following syntax:

```
/opt/fortitude/bin/ldapget /path/to/certdatabase ldap://...
```

Using the HTTPS URL Format.

```
https://ca.example.com:1025/getCRL?op=getCRL&issuepoint=MasterCRL
```

4. Example CRL Revocation File

CRL revocation options are specified in `/opt/fortitude/conf.d/revocator.conf`.

The following is an example CRL revocation configuration file:

```
CRLEngine on
CRLFile http://somehost.example.com/MasterCRL.crl;60;60
CRLAgeCheck off
CRLUpdateCritical off
```

5. Tomcat SSL Configuration

5.1. Introduction

To configure SSL for Tomcat, you need to set up a *Connector*. A Connector represents an endpoint by which requests are received and responses are returned. Each Connector passes requests to the associated *Container* (normally an *Engine*) for processing.

5.2. Setting up a Connector in Tomcat

To set up a Connector, you need to modify the Tomcat5 configuration file, `/opt/fortitude/server.xml`, and specify the required attributes. These include the listening port, either secure or insecure, the SSL implementation, various cipher options, and other details. These attributes use the following format: `attributeName="<attributeValue>"`, and are discussed in more detail below.

In a Fortitude configuration, you need to configure Tomcat to load the correct SSL implementation (`tomcatjss`) instead of using the built-in version. This component talks to JSS which in turn talks to NSS. This configuration enables the FIPS 140-certified cryptographic module.

The following attributes are required:

port

The port that the server monitors for incoming requests

scheme

This must be set to "https"

clientAuth

Specifies whether or not client authentication is required in the Connector (or port). Valid values are true and false.

sslProtocol

This must be set to `SSL`

sslOptions

A comma-separated list of SSL options to pass to the SSL implementation. The `tomcatjss` module supports SSLv2, SSLv3, and TLSv1.

ssl2Ciphers

A comma-separated list of SSLv2 ciphers that `tomcatjss` should accept or reject from the client

ssl3Ciphers

A comma-separated list of SSLv3 ciphers that `tomcatjss` should accept or reject from the client



Note

For the `ssl2Ciphers` and `ssl3Ciphers` attributes, prefix each cipher name with a plus sign (+) to accept the cipher, or a minus sign (-) to reject the cipher.

SSLImplementation

The SSL plugin that Tomcat needs to call to perform SSL transactions. This must be set to `org.apache.tomcat.util.net.jss.JSSImplementation`.

serverCertNickFile

The file that contains the nickname of the server certificate



Note

The server certificate file uses a single line with the server certificate that you want to use. Blank lines and lines beginning with a hash mark (#) are ignored.

certdbDir

The location of the certificate database. This must be an absolute path.

passwordFile

The location of the file that contains the token password. Uses the default `PlainPasswordFile` class.



Note

The password file uses the following syntax: `token=password`

For example: `internal=httpstest`

passwordClass

A Java class that provides a mechanism to get the token password. This reads from text files.

The name of the NSS software token is "internal".

Example Connector Configuration. The following is an example of a Connector configured to listen on port 9443. Note that this example has been edited for layout purposes.

```
<Connector port="9443"
  maxHttpHeaderSize="8192" maxThreads="150"
  minSpareThreads="25" maxSpareThreads="75"
  enableLookups="false" disableUploadTimeout="true"
  acceptCount="100"
  scheme="https"
  secure="true"
  clientAuth="false"
  sslProtocol="SSL"
  sslOptions="ssl2=true,ssl3=true,tls=true"
  ssl2Ciphers="-SSL2_RC4_128_WITH_MD5,-SSL2_RC4_128_EXPORT40_WITH_MD5,-SSL2_RC2_128_CBC_WITH_MD5,\
  -SSL2_RC2_128_CBC_EXPORT40_WITH_MD5,-SSL2_DES_64_CBC_WITH_MD5,\
  -SSL2_DES_192_EDE3_CBC_WITH_MD5"
  ssl3Ciphers="-SSL3_FORTEZZA_DMS_WITH_NULL_SHA,-SSL3_FORTEZZA_DMS_WITH_RC4_128_SHA,\
  +SSL3_RSA_WITH_RC4_128_SHA,-SSL3_RSA_EXPORT_WITH_RC4_40_MD5,+SSL3_RSA_WITH_3DES_EDE_CBC_
  +SSL3_RSA_WITH_DES_CBC_SHA,-SSL3_RSA_EXPORT_WITH_RC2_CBC_40_MD5,\
  -SSL3_FORTEZZA_DMS_WITH_FORTEZZA_CBC_SHA,-SSL_RSA_FIPS_WITH_DES_CBC_SHA,\
  +SSL_RSA_FIPS_WITH_3DES_EDE_CBC_SHA,-SSL3_RSA_WITH_NULL_MD5,\
  -TLS_RSA_EXPORT1024_WITH_RC4_56_SHA,+TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA"
  SSLImplementation="org.apache.tomcat.util.net.js.JSSImplementation"
  serverCertNickFile="/opt/fortitude/conf/serverCertNick.conf"
  passwordFile="/opt/fortitude/conf/password.conf"
  passwordClass="org.apache.tomcat.util.net.js.PlainPasswordFile"
```

```
certdbDir="/opt/fortitude/alias"  
/>
```

6. Troubleshooting and FAQ

6.1. Does `mod_nss` support `mod_proxy`?

In order to use the `mod_nss` proxy support you need to build your own `mod_proxy` by applying a patch found in http://issues.apache.org/bugzilla/show_bug.cgi?id=36468.

This patch is needed because the SSL proxy function calls are hardcoded into the `mod_proxy` Apache module. Installing this patch means that you can compare the host-name contained in the remote certificate with the hostname you intended to visit. This prevents man-in-the-middle attacks.

You also have to change the SSL functions that `mod_proxy` attempts to use. You need to apply this patch:

```
1038,1039c1038,1039  
< APR_DECLARE_OPTIONAL_FN(int, ssl_proxy_enable, (conn_rec *));  
< APR_DECLARE_OPTIONAL_FN(int, ssl_engine_disable, (conn_rec *));  
---  
> APR_DECLARE_OPTIONAL_FN(int, nss_proxy_enable, (conn_rec *));  
> APR_DECLARE_OPTIONAL_FN(int, nss_engine_disable, (conn_rec *));  
1041,1042c1041,1042  
< static APR_OPTIONAL_FN_TYPE(ssl_proxy_enable) *proxy_ssl_enable = NULL;  
< static APR_OPTIONAL_FN_TYPE(ssl_engine_disable) *proxy_ssl_disable = NULL;  
---  
> static APR_OPTIONAL_FN_TYPE(nss_proxy_enable) *proxy_ssl_enable = NULL;  
> static APR_OPTIONAL_FN_TYPE(nss_engine_disable) *proxy_ssl_disable = NULL;  
1069,1070c1069,1070  
< proxy_ssl_enable = APR_RETRIEVE_OPTIONAL_FN(ssl_proxy_enable);  
< proxy_ssl_disable = APR_RETRIEVE_OPTIONAL_FN(ssl_engine_disable);  
---  
> proxy_ssl_enable = APR_RETRIEVE_OPTIONAL_FN(nss_proxy_enable);  
> proxy_ssl_disable = APR_RETRIEVE_OPTIONAL_FN(nss_engine_disable);  
>
```

6.2. I have specified `SSLv2` in `NSSCipherSuite` but I can't use `SSLv2`. Why not?

`SSLv2` ciphers are not supported in `mod_nss` and cannot be enabled.

Chapter 6. Migrating from NES

1. Introduction

The following sections cover the process of migrating from *Netscape Enterprise Server* (NES) to Fortitude. Because of the differences in the way these two systems operate, a complete one-to-one mapping of configuration options is not possible. The migration process attempts to map as many options as possible to the new system, but some issues may arise. Simple configurations should migrate with few problems. The more complex the configuration, the higher the likelihood of problems in the migration process.

The recommended strategy is to attempt the migration and to check for any errors. The error messages should give an indication of the type of hand-tuning that may be required. Some hand-tuning may be desirable even if the migration was successful (for example, for readability).

2. Migrating the NES Configuration

The `/opt/fortitude/bin/migrate` script is provided to assist in migrating an NES configuration to use with the Apache Web server. This is a Perl script and it takes a single argument; the path to the NES installation, as follows:

```
/opt/fortitude/bin/migrate /path/to/nas/
```

The script prompts you to choose the instance that you want to migrate. Unlike NES, Fortitude does not support multiple instances, so you can only perform the migration once. If you need multiple instances, you should consider the Apache virtual server capabilities. Refer to <http://httpd.apache.org/docs/2.0/vhosts/> for more information on this topic.

Depending on the NES configuration options that you have specified, you may see some warning messages during the migration process. Refer to [Section 8, "Migration Warning Messages"](#) for more information on warning messages.

2.1. Unprocessed Directives

A number of directives are not processed either because no Apache equivalent exists, they are handled automatically, or because they otherwise do not apply to Apache. These are briefly described below:

No Equivalent Directive Available. There is no Apache equivalent for the following three directives. These automatically prepend or append HTML data to the requested file.

- Service add-footer
- Service add-header
- Service append-trailer

2.2. A Simple Migration Session

Automatically Handled Directives. The following directives are ignored as they are handled automatically by Apache.

- AuthTrans get-sslid
- PathCheck check-acl
- PathCheck unix-uri-clean
- Service image-map

Directives That do not Apply to Apache. The following directives are not processed because they do not apply to Apache.

- NameTrans NSServletNameTrans
- NameTrans es-search-nametrans
- Service NSServletService
- Service key-toosmall
- Service send-file
- Service send-range
- Service upload-file

2.2. A Simple Migration Session

The following example shows the output of a simple migration:

```
# /opt/fortitude/bin/migrate /usr/netscape/nes60
Migrating from: /usr/netscape/nes60/
               to: /opt/fortitude

Migrate which instance:
  1. https-test.example.com

Instance (0 to quit): 1
Migrating instance https-test.example.com.
Migrating server.xml and obj.conf...

By default NES handles path-info for every request
Apache lets each handler determine whether pathinfo is examined.
If you would like finer control, see the Apache AcceptPathInfo directive.

Migrating access controls...
Note that in NES you had to provide an LDAP bind name and password. In Apache this is not required.
You may provide one if you like using the AuthLDAPBindDN and AuthLDAPBindPassword options.
These are not added in the migration.

Skipping ACL default
Skipping ACL es-internal
0 ACLs migrated to Fortitude.
Instance (0 to quit): 0
```

2.3. A More Complex Migration Session

The following example shows the output of a more complex migration:

```
# cd /opt/fortitude
# bin/migrate /usr/netscape/migrate
Migrating from: /usr/netscape/migrate
               to: /opt/fortitude

Migrate which instance:
  1. https-test

Instance (0 to quit): 1
Migrating instance https-test.
Migrating server.xml and obj.conf...

By default NES handles path-info for every request.
Apache lets each handler determine whether pathinfo is examined.
If you would like finer control, see the Apache AcceptPathInfo directive.

Unknown PathCheck directive: htaccess-find

Unknown Service directive: service-nsfc-dump

There is no append-trailer equivalent in Apache.

There is no append-trailer equivalent in Apache.

There is no append-trailer equivalent in Apache.

Unknown ObjectType directive: image-switch

Unknown ObjectType directive: html-switch

There is no Apache equivalent to type-by-exp. See the Apache AddType directive.

Unknown ObjectType directive: change-type

Unknown PathCheck directive: add-header

Unknown PathCheck directive: add-footer

Unknown PathCheck directive: add-header

Unknown PathCheck directive: add-footer

Unknown Service directive: nsapi_test

Migrating access controls...
Note that in NES you had to provide an LDAP bind name and password. In Apache this is not required.
You may provide one if you like using the AuthLDAPBindDN and AuthLDAPBindPassword options.
These are not added in the migration.

Skipping ACL default
Skipping ACL es-internal
0 ACLs migrated to Fortitude.
```



Caution

The migration does NOT back up the Apache configuration before performing the operation. You should back up `conf/*` and `conf.d/*` before starting the migration.

3. Migrating Web Applications

Tomcat can automatically deploy WAR files. For this to occur, you need to copy the `.war` file into the `/opt/fortitude/webapps/` directory. Tomcat automatically unpacks it into a directory of the same name as the `.war` file (not including the file extension), and prepares it for use.

If you are using Apache as a front-end to Tomcat (the default), then you also need to modify `/opt/fortitude/conf/webapps.conf` and add a `JkMount` entry to register the URI.

For example, if you have a web application named `helloworld.war`, you should use the following procedure to deploy it for use by Tomcat:

Procedure 6.1. Automatically deploying a `.war` file

1. Copy the web application file (`helloworld.war`) into the `/opt/fortitude/webapps/` directory. Tomcat automatically unpacks this into the `webapps/helloworld/` directory.
2. Modify the `/opt/fortitude/conf/webapps.conf` file to include the following line:

```
JkMount /helloworld/* ajp13
```

3. Restart Apache.

4. Migrating Access Controls

The ACL file for an NES instance can be found in `<server-root>/ht-tpacl/<generated-instance.acl>`

Access control in Apache can be configured either in `httpd.conf` or in an `.htaccess` file. Using an `.htaccess` file has the advantage that you do not need to restart the server when the file is changed. The downside is that a slight performance penalty is incurred because the server reads the file with every page request to determine if the file has changed.

An NES ACL uses the following structure:

```
acl "uri=/acl/protected.html";
authenticate (user,group) {
  prompt = "Enterprise Server";
  method = "basic";
  database = "default";
```

```
};
deny (all) user = "anyone";
allow (all) user = "alpha";
```

The database default is defined in NES in `<server-root>/userdb/dbswitch.conf`. For the following NES configuration file:

```
directory ldapregular ldap://ldap.example.com:3389/o=example.com
ldapregular:binddn uid=admin,ou=Administrators,ou=topologyManagement,o=netscapeRoot
ldapregular:encoded bindpw YWRtaW4=
```

The Apache equivalent for this ACL is:

```
<Location "/acl/protected.html">
AuthName "Enterprise Server"
AuthType basic
AuthLDAPURL ldap://ldap.example.com:3389/o=example.com?uid?sub?(objectClass=*) require user alpha
</Location>
```

The `aclmigrate` script is provided to aid in translating an NES ACL file into one usable by Apache.

This script uses the following syntax:

```
aclmigrate --test/path/to/NES <instance-name>/path/to/Fortitude
```

`--test` prints to stdout any changes that would be made.



Note

This script is provided as an aid only. It is strongly recommended that the final result be examined carefully for accuracy. Further, very complex ACLs may not be converted at all, in which case a warning is printed to stdout.

4.1. A Sample ACL Migration

The following example shows the output of a simple ACL migration:

```
# bin/aclmigrate --test /home/rcrit/netscape/migrate/gat https-test /opt/fortitude
Note that in NES you had to provide an LDAP bind name and password. In Apache this is not required.
You may provide one if you like using the AuthLDAPBindDN and AuthLDAPBindPassword options.
These are not added in the migration.
```

4.1. A Sample ACL Migration

```
Skipping ACL default
Skipping ACL es-internal

<Location "/acl/acl02.html">
  Require user alpha
  AuthType basic
  AuthType "Enterprise Server"
  AuthLDAPURL ldap://ldap.example.com:3389/o%3DTestCentral?uid?sub?(objectClass=*)
</Location>
Skipping ACL uri=/acl/acl03.html

<Location "/acl/acl05.html">
  Require group alphaonly
  AuthType basic
  AuthType "Enterprise Server"
  AuthLDAPURL ldap://ldap.example.com:3389/o%3DTestCentral?uid?sub?(objectClass=*)
</Location>

<Location "/acl/acl13.html">
  Satisfy all
  Require user alpha
  allow from 192.
  AuthType basic
  AuthType "Enterprise Server"
  AuthLDAPURL ldap://ldap.example.com:3389/o%3DTestCentral?uid?sub?(objectClass=*)
</Location>

<Location "/acl/acl14.html">
  Satisfy all
  Require user alpha
  allow from .example.com
  AuthType basic
  AuthType "Enterprise Server"
  AuthLDAPURL ldap://ldap.example.com:3389/o%3DTestCentral?uid?sub?(objectClass=*)
</Location>
This ACL uses ssl. Be sure to check the migrated version very carefully.

<Location "/acl/aclS01.html">
  NSSVerifyClient require
  NSSUserName SSL_CLIENT_S_DN_UID
  Require user alpha gamma
</Location>

<Location "/acl/aclS06.html">
  Satisfy all
  Require user alpha
  NSSRequire %{HTTPS} eq "on"
  AuthType basic
  AuthType "Enterprise Server"
  AuthLDAPURL ldap://ldap.example.com:3389/o%3DTestCentral?uid?sub?(objectClass=*)
</Location>

<Location "/acl/aclS07.html">
  Satisfy all
  Require user alpha
  NSSRequire %{HTTPS} eq "off"
  AuthType basic
  AuthType "Enterprise Server"
  AuthLDAPURL ldap://ldap.example.com:3389/o%3DTestCentral?uid?sub?(objectClass=*)
</Location>
Skipping ACL default
Skipping ACL uri=/upload/*
To convert an ACL using a cipher you will need to manually add the cipher entry.
A place holder has been added.

<Location "/acl/aclC01.html">
  NSSRequire ( %{SSL_CIPHER} = "RC4" )
```

4.1. A Sample ACL Migration

```
</Location>
```

To convert an ACL using a cipher you will need to manually add the cipher entry. A place holder has been added.

```
<Location "/acl/aclNC01.html">
    NSSRequire ( %{SSL_CIPHER} != "RC4" )
</Location>
9 ACLs migrated to Fortitude.
```

This is the NES file that was converted:

```
version 3.0;
acl "default";
authenticate (user, group) {
    prompt = "iPlanet Web Server";
};
allow (read, list, execute,info) user = "anyone";
allow (write, delete) user = "all";

acl "es-internal";
allow (read, list, execute,info) user = "anyone";
deny (write, delete) user = "anyone";

version 3.0;

acl "uri=/acl/acl02.html";
authenticate (user,group) {
    prompt = "Enterprise Server";
    method = "basic";
    database = "ldapregular";
};
deny (all) user = "anyone";
allow (all) user = "alpha";

acl "uri=/acl/acl03.html";
authenticate (user,group) {
    prompt = "Enterprise Server";
    method = "basic";
    database = "nullall";
};
deny (all) user = "anyone";
allow (all) user = "alpha";

acl "uri=/acl/acl05.html";
authenticate (user,group) {
    prompt = "Enterprise Server";
    method = "basic";
    database = "ldapregular";
};
deny (all) user = "anyone";
allow (all) group = "alphaonly";

acl "uri=/acl/acl13.html";
authenticate (user,group) {
    prompt = "Enterprise Server";
    method = "basic";
    database = "ldapregular";
};
deny (all) user = "anyone";
allow (all) (user = "alpha") and (ip = "192.*");

acl "uri=/acl/acl14.html";
authenticate (user,group) {
    prompt = "Enterprise Server";
    method = "basic";
    database = "ldapregular";
};
```

```

deny (all) user = "anyone";
allow (all) (user = "alpha") and (dns = "*.example.com");

acl "uri=/acl/aclS01.html";
authenticate (user,group) {
    prompt = "Enterprise Server";
    method = "ssl";
    database = "ldapregular";
};
deny (all) user = "anyone";
allow (all) user = "alpha,gamma";

acl "uri=/acl/aclS06.html";
authenticate (user,group) {
    prompt = "Enterprise Server";
    method = "basic";
    database = "ldapregular";
};
deny (all) user = "anyone";
allow (all) (user = "alpha" and ssl = "yes");

acl "uri=/acl/aclS07.html";
authenticate (user,group) {
    prompt = "Enterprise Server";
    method = "basic";
    database = "ldapregular";
};
deny (all) user = "anyone";
allow (all) (user = "alpha" and ssl = "no");

acl "default";
authenticate (user, group) {
    prompt = "Enterprise Server";
};
allow (read, list, execute,info) user = "anyone";
allow (write, delete) user = "all";

acl "uri=/upload/*";
allow (all) user="anyone";

acl "uri=/acl/aclC01.html";
deny (all) cipher = "anyone";
allow (all) cipher = "RC4";

acl "uri=/acl/aclNC01.html";
deny (all) cipher = "anyone";
allow (all) cipher != "RC4";

```

5. Migrating CGIs

To use CGI scripts with Fortitude, you can either:

- Copy the scripts from the NES CGI directory to the corresponding Fortitude directory, or
- Configure Fortitude to use the same directory that NES uses.



Note

The Apache and NES environment variables may differ slightly. Refer to the "Environment Variables" entry, below, for a comparison.

To configure Fortitude to use the same directory that NES uses, you need to modify the `httpd.conf` file. This is described below:

Procedure 6.2. Configuring Fortitude to use the NES `cgi-bin` directory

1. Open the `/opt/fortitude/conf/httpd.conf` file for editing.
2. Define a `ScriptAlias` for the directory. This maps the `CGI` directory URI to a physical directory containing the `CGIs`.

```
ScriptAlias /cgi-bin/ "/path/to/my/cgi-bin/"
```

3. Create a directory reference that defines what is, and what is not, allowed to occur in the `CGI` directory. For example, you need to prevent normal users from listing the directory.

The following is an example `CGI` directory configuration:

```
<Directory "/path/to/my/cgi-bin">  
AllowOverride None  
Options None  
Order allow,deny  
Allow from all  
</Directory>
```

You do not need to set the `ExecCGI` option. This is set automatically by the `ScriptAlias`.

6. Other Manual Migration Tasks

Listen sockets

To change the port that your web server accepts connections on, refer to the `ListenApache` configuration directive.

DocumentRoot

To change the default location of your web documents, refer to the `DocumentRoot` configuration directive. Note that this directive is controlled by both the `DocumentRoot` directive and the `<Directory>` entry, which configures how objects in that directory are accessed.

Virtual Servers

An NES virtual server (hardware or software) is configured in `server.xml`. An Apache virtual

7. Known Issues

`server` (or `VirtualHost`) is set using the `VirtualHost` directive in `httpd.conf`.

Refer to the Apache documentation at <http://httpd.apache.org/docs/2.0/vhosts/> for comprehensive coverage of this subject.

Environment Variables

Both NES and Apache adhere to the CGI/1.1 specification. You should be aware, however, that they are not implemented in exactly the same way. Some of these differences are highlighted below:

NES always sets the `HTTPS` variable.

NES sets the `SERVER_URL` variable. This is the URL of the web server itself.

Apache does not always set the `REMOTE_HOST` variable (for example, if no DNS lookup was performed). In NES this would be the IP address.

Apache adds the `SERVER_SIGNATURE`, `REQUEST_URI` and `DOCUMENT_ROOT` variables.

7. Known Issues

The following issues are known to exist when migrating from NES to Apache:

- SSL is not supported when performing LDAP authentication using Apache `mod_auth` modules.

8. Migration Warning Messages

If you are trying to migrate a complex configuration, or if you have implemented custom plugins, etc., you may see one or more of the following error messages during the migration process.

`AuthTrans basic-auth`

Indicates that you have a custom authentication function. You need to port this to Apache or consider one of the Apache built-in functions.

NES provides several methods for creating custom authentication. An NSAPI plugin using the `AuthTrans basic-auth` directive is one such method. If you want to continue using this method, you need to convert this into an Apache authentication plugin.

`AuthTrans basic-ncsa with a DBM database`

Indicates that you need to re-create your DBM file in the `/path/to/dbm/` directory.

The Apache DBM file format is different from NES. In NES, users and groups are contained within the same database. Apache requires users and groups to be stored in separate DBM files.

`NameTrans home-page`

What's this about? Something to do with the Netscape `obj.conf` file `NameTrans` entry that the script doesn't handle?

`PathCheck deny-existence`

What's this about? I can see what's happening the Netscape world but not how to treat it here. How is it related to the following (which makes perfect sense)?

8. Migration Warning Messages

PathCheck deny-existence is not migrated

This directive is similar to access control but it returns a "404 Not Found" error rather than a "401 Authorization Required" or "403 Forbidden" error for files that a user may not be permitted to access.

PathCheck find-links

NES offers three ways of handling symbolic links:

- Follow symlinks
- Do not follow symlinks
- Follow symlinks only if the UID of the link owner is the same as the UID of the web server account

The handling of symbolic links in Apache is similar to but different from NES; you should carefully review these options and their use to ensure that your web server is secure.

Apache uses the *FollowSymLinks* and *SymLinksIfOwnerMatch* options to handle symbolic link checking. These options are part of the Apache *Options* directive.

The migration script currently inserts the *SymLinksIfOwnerMatch* option into a *Location* entry, and consequently is not used. To take advantage of the functionality offered by *SymLinksIfOwnerMatch*, you should change this to a *Directory* entry instead.

PathCheck find-pathinfo

A default NES configuration includes the *PathCheck find-pathinfo* directive. This searches for extra path information that may be included as part of a file request. This extra path information is often referred to as `PATH_INFO` and is frequently used by CGI programs.

In the following example, the file is `/index.html` and `PATH_INFO` is `/more/info`.

```
/index.html/more/info
```

A default NES configuration processes `PATH_INFO` for every page request. Apache, however, lets each handler determine whether `PATH_INFO` should be examined.

PathCheck load-config

NES uses a per-directory configuration option, *Dynamic Configuration* (deprecated in NES 6.0). This option creates an `.nsconfig` file for each directory.

`nsconfig` files are not supported in Apache, however you can use the `htconvert` utility supplied with NES to convert these into `.htaccess` files suitable for use by Apache. This utility uses the following syntax:

```
<server-root>/plugins/htaccess/htconvert /path/to/nsconfigfile
```

ObjectType force-type

The `force-type` directive sets the MIME type for the requested object.

In NES you can set the content-encoding type returned by the web server. Type-by-encoding is not supported in Apache.

ObjectType `type-by-exp`

The `type-by-exp` directive uses a regular expression to assign a MIME type.

There is no Apache equivalent to `type-by-exp`. Refer to the Apache `AddType` directive.

Unknown `<x>` Directive

Indicates that a directive in your NES configuration is not handled by the migration program. This could be caused by the following or other issues:

- You may have implemented a custom plugin that defines additional directives. You need to port this plugin to Apache.
- You may be using a 3rd-party plugin that defines additional directives. You need to contact the 3rd party or determine if an existing alternative is available in Apache.
- The migration process may have encountered a directive that it is unaware of.

Refer to <http://httpd.apache.org/docs/2.0/mod/core.html> for more information on the directives and options available for Apache.

Chapter 7. Additional Resources

For more information about Fortitude, the Apache Web server and Tomcat, refer to the following resources:

Apache Configuration

Apache Foundation documentation — <http://httpd.apache.org/docs/2.0/>

Apache-SSL package information — <http://www.apache-ssl.org> [<http://www.apache-ssl.org/>]

mod_ssl package information — <http://www.modssl.org> [<http://www.modssl.org/>]

Apache 2.0 license information — <http://www.apache.org/licenses>
[<http://www.apache.org/licenses/>]

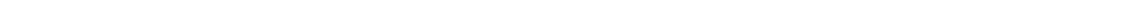
Network Security Services information — <http://www.mozilla.org/projects/security/pki/nss>
[<http://www.mozilla.org/projects/security/pki/nss/>]

Tomcat Configuration

The Apache Tomcat 5.5 Servlet/JSP Container

— <http://tomcat.apache.org/tomcat-5.5-doc/index.html>

Part I. Appendixes



Appendix A. Fortitude 1.0.0 Installation Components

Fortitude 1.0.0 consists of the following RPMs:

RPM Name	Description
fortitude-web	Contains the Apache configuration.
fortitude-web-java	Contains the Tomcat configuration.
fortitude-web-migrate	Contains the NES migration utilities.
fortitude-mod_nss	Provides SSL capabilities to Apache using NSS.
fortitude-mod_revocator	Enables automatic retrieval and installation of Certificate Revocation Lists (CRLs) into a running Apache web server. <code>mod_revocator</code> Requires <code>mod_nss</code> .
fortitude-Apache-Admin-Config	A Perl module that can read and process Apache configuration files. It is used in the NES migration utilities.
tomcatjss	A Java Secure Socket Extension (JSSE)-compliant plugin based on JSS that provides SSL to Tomcat.

Table A.1. Fortitude Installation Components

Appendix B. Unsupported Apache `mod_ssl` Directives

The following directives exist in the `mod_ssl` implementation of Apache but are not supported in Fortitude 1.0.0, which uses `mod_nss`.

- `SSLSessionCache`
- `SSLMutex`
- `SSLCertificateChainFile`
- `SSLCARevocationPath`
- `SSLCARevocationFile`
- `SSLVerifyDepth`
- `SSLCryptoDevice`
- `SSLCertificateFile`
- `SSLCertificateKeyFile`
- `SSLCACertificateFile`
- `SSLCACertificatePath`
- `SSLProxyVerify`
- `SSLProxyVerifyDepth`
- `SSLProxyCACertificateFile`
- `SSLProxyCACertificatePath`
- `SSLProxyCARevocationPath`
- `SSLProxyCARevocationFile`
- `SSLProxyMachineCertificateFile`
- `SSLProxyMachineCertificatePath`

Appendix C. List of Available Ciphers

Cipher Name	NSS Cipher definition	Protocol
rsa_3des_sha	SSL_RSA_WITH_3DES_EDE_CBC_SHA	SSLv3/TLSv1
rsa_des_sha	SSL_RSA_WITH_DES_CBC_SHA	SSLv3/TLSv1
rsa_null_md5	SSL_RSA_WITH_NULL_MD5	SSLv3/TLSv1
rsa_null_sha	SSL_RSA_WITH_NULL_SHA	SSLv3/TLSv1
rsa_rc2_40_md5	SSL_RSA_EXPORT_WITH_RC2_CBC_40_MD5	SSLv3/TLSv1
rsa_rc4_128_md5	SSL_RSA_WITH_RC4_128_MD5	SSLv3/TLSv1
rsa_rc4_128_sha	SSL_RSA_WITH_RC4_128_SHA	SSLv3/TLSv1
rsa_rc4_40_md5	SSL_RSA_EXPORT_WITH_RC4_40_MD5	SSLv3/TLSv1
fortezza	SSL_FORTEZZA_DMS_WITH_FORTEZZA_CBC_SHA	SSLv3/TLSv1
fortezza_rc4_128_sha	SSL_FORTEZZA_DMS_WITH_RC4_128_SHA	SSLv3/TLSv1
fortezza_null	SSL_FORTEZZA_DMS_WITH_NULL_SHA	SSLv3/TLSv1
fips_des_sha	SSL_RSA_FIPS_WITH_DES_CBC_SHA	SSLv3/TLSv1
fips_3des_sha	SSL_RSA_FIPS_WITH_3DES_EDE_CBC_SHA	SSLv3/TLSv1
rsa_des_56_sha	TLS_RSA_EXPORT1024_WITH_DES_CBC_SHA	SSL3/TLSv1
rsa_rc4_56_sha	TLS_RSA_EXPORT1024_WITH_RC4_56_SHA	SSLv3/TLSv1
rsa_aes_128_sha	TLS_RSA_WITH_AES_128_CBC_SHA	SSLv3/TLSv1
rsa_aes_256_sha	TLS_RSA_WITH_AES_256_CBC_SHA	SSLv3/TLSv1

Table C.1. List of Available Ciphers

Appendix D. CGI and SSI Environment Variables

The following is a list of the available CGI and SSI environment variables:

Name	Description
HTTPS	Set to "on" if HTTPS is being used

Table D.1. Always Set

Name	Description
SSL_VERSION_INTERFACE	The version of <code>mod_nss</code> that the server is running.
SSL_VERSION_LIBRARY	The version of NSS that <code>mod_nss</code> was compiled against.
SSL_PROTOCOL	SSLv2, SSLv3 or TLSv1
SSL_CIPHER	The cipher that the connection is using.
SSL_CIPHER_EXPORT	<code>true</code> if the cipher is an export cipher, <code>false</code> otherwise.
SSL_CIPHER_USEKEYSIZE	Number of bits that the cipher is using.
SSL_CIPHER_ALGKEYSIZE	Maximum number of bits possible in the cipher.
SSL_CLIENT_VERIFY	NONE if no client authentication, SUCCESS or FAILED if <code>SSLVerifyCert</code> is set.
SSL_CLIENT_V_START	Client certificate validity start time.
SSL_CLIENT_V_END	Client certificate validity end time.
SSL_CLIENT_M_VERSION	X.509 version of the client certificate.
SSL_CLIENT_M_SERIAL	Serial number of the client certificate.
SSL_CLIENT_A_KEY	Algorithm used for the client key.
SSL_CLIENT_A_SIG	Algorithm used for the signature of the client key.
SSL_CLIENT_S_DN	Distinguished Name (DN) of the client certificate.
SSL_CLIENT_S_DN_[C,ST,L,O,O U,CN,T,I,G,S,D,UID,Email]	Components of the client certificate. Only those that exist in the certificate are created.
SSL_CLIENT_I_DN	Distinguished Name (DN) of the client certificate issuer.
SSL_CLIENT_I_DN_[C,ST,L,O,O U,CN,T,I,G,S,D,UID,Email]	Components of the client issuer certificate. Only those that exist in the certificate are created.
SSL_SERVER_DN	Distinguished Name (DN) of the server certificate.

Name	Description
SSL_SERVER_DN_[C,ST,L,O,OU,CN,T,I,G,S,D,UID,Email]	Components of the server certificate. Only those that exist in the certificate are created.
SSL_SERVER_I_DN_[C,ST,L,O,OU,CN,T,I,G,S,D,UID,Email]	Components of the server issuer certificate. Only those that exist in the certificate are created.
SSL_SERVER_M_VERSION	X.509 version of the server certificate.
SSL_SERVER_M_SERIAL	Serial number of the server certificate.
SSL_SERVER_V_START	Server certificate validity start time.
SSL_SERVER_V_END	Server certificate validity end time.
SSL_SERVER_A_KEY	Algorithm used for the server key.
SSL_SERVER_A_SIG	Algorithm used for the signature of the server key.
SSL_SESSION_ID	SSL Session ID.

Table D.2. +StdEnvVars

Name	Description
SSL_SERVER_CERT	The server certificate in PEM format.
SSL_CLIENT_CERT	The client certificate in PEM format (if available).
SSL_CLIENT_CERT_CHAIN_[0..n]	Each certificate in the client certificate chain in PEM format (including the client certificate itself).

Table D.3. +ExportCertData