

NAREGI-CA Software

# Operation Manual

October 13, 2004

National Institute of Informatics

## CONTENTS

1.	INTRODUCTION .....	1
1.1.	Information Technology .....	1
1.2.	Operating Environment .....	4
2.	CA and RA Setup .....	5
2.1.	New CA (with CA Server Registration) .....	5
2.1.1.	Procedure for using HSM.....	10
2.2.	New CA (without CA Server Registration) .....	11
2.3.	New RA (with RA Server Registration) .....	13
2.4.	PKCS#12 for CA .....	15
2.5.	Updating CA Certificate .....	16
2.6.	aica.cnf Setting .....	17
2.7.	Setting gridmap.cnf .....	28
3.	CA and RA Server Operations.....	30
3.1.	Start and Stop CA Server .....	30
3.2.	Start and Stop CRL Publisher .....	33
3.3.	Web Enrollment Setting .....	35
3.4.	Start and Stop Web Enrollment .....	42
3.5.	Start and Stop RA Server .....	43
3.6.	Start and Stop enrollment check daemon .....	45
3.7.	Setting XKMS Service .....	47
3.8.	Starting and Stopping XKMS Service .....	54
3.9.	Setting Grid Linking Function.....	55
3.10.	Starting and Stopping Grid Linking Function.....	56
3.11.	Remote Access Setting.....	58
3.12.	Automatically Generate SSL Server Certificate .....	60
3.13.	Back Up CA and RA Data .....	60
4.	CA Operations.....	62
4.1.	Create Certificate Request .....	62
4.2.	Issue Certificate for Request .....	64
4.3.	Register Request then Verify and Issue Certificate .....	66
4.4.	Update Certificate .....	67
4.5.	Batch Issue Certificates .....	68
4.6.	Revoke Certificate.....	70
4.7.	Unrevoke Certificate.....	71
4.8.	Issue CRL.....	72

4.9.	Export Certificate and Private Key .....	73
4.10.	Import Private Key.....	74
4.11.	Delete Private Key .....	75
4.12.	Show Profile Settings.....	76
4.13.	Update Profile Settings.....	78
4.14.	Update Extension Information for Profile.....	81
4.15.	Add and Delete Profile .....	86
4.16.	Add and Delete Operator .....	88
5.	Other Operations .....	91
5.1.	Certificate Store.....	91
5.2.	Verify Certificate.....	94
5.3.	View Certificate .....	95
5.4.	Change Certificate .....	96
5.5.	Create Certificate Request .....	98
5.6.	Web Enroll Auxiliary Tool.....	99
5.7.	aica.cnf Tools .....	101
5.8.	Command for aira Offline CA .....	102

# 1. INTRODUCTION

This manual describes the features and the operating environment for NAREGI CA, which is a grid prototype Certificate Authority.

## 1.1. Information Technology



NAREGI CA is a CA server and command shell that operates on UNIX. As described below, the package consists of a variety of utility commands including key generation and certification issuance, verification, and storage.

- CA aisetup.sh  
This shell script builds a CA. After building the CA, you can use the aica commands to operate it. The aisetup.sh command can automatically register the aicad (CA server) and the aicrlpub (CRL publisher) when it updates the aica.cnf file, which is the configuration file for the CA server. When the CA server is running, in addition to local access, it is also possible to access remotely.
- CA ainewca.sh  
This shell script builds a CA. After building the CA, you can use the aica command to operate it. This command does not update the aica.cnf file. With this command, you can operate CA operations based on local file access.
- RA ainewra.sh  
This shell script builds a Registration Authority (RA). With this command you can add RA settings for a specific CA. After configuring your RA, client machines will be able to access the CA server directly.
- CA server aicad  
This is the CA server (daemon) that can be accessed remotely when it is registered in the aica.cnf configuration file. With this daemon you can run the various CA operations, which are defined in the lightweight certificate management protocol (LCMP). These operations include certificate issuance, update, revocation, and export; private key storage and export; certificate

revocation list (CRL) issuance; certificate signing request (CSR) receipt and approval; CA profile settings; and certificate issuance list display.

- CRL publisher `aicrlpub`

This is the CRL publisher server that issues a CRL or an authority revocation list (ARL) at constant time intervals based on file access and remote access. It then outputs the CRL/ARL to a local directory or to a lightweight directory access protocol (LDAP) server.

- CA operation `aica`

This is the CA operation command that you use to configure settings for certificate issuance, update, revocation, and export; private key storage and export; certificate revocation list (CRL) issuance; certificate signing request (CSR) receipt and approval; the CA profile; the certificate issuance list display; and the CA operator. In addition to CA operations through local file access, it is also possible to operate the CA through remote access.

- Enroll modules `airad`, `aienroll`, and CGI modules

Three modules are used to configure the RA server. They are the `aired` daemon, the web enroll CGI modules, and the `aienroll` server that manages session files.

The `airad` daemon receives requests from the `certreq` or the `grid-certreq` for certificate issuance, update, and revocation. If a CGI script is called from an Apache web server, requests for certificate issuance, update, and revocation are made to the CA server through CGI. The two types of enrollment are immediate issuance and validation and issuance by a CA operator. For the later, certificates are validated and issued at regular time intervals through `aienroll`. In addition, before a certificate is issued, CGI provides certificate lifecycle functionality through various types of authentication including anonymous, ID/password, license ID (one-time license), SSL, and license ID/Challenge PIN. In addition, `aienroll` can send an e-mail message notifying of certificate updates.

Especially when the License ID/Challenge PIN authentication mode is used, user management is performed by the RA operator based on the LDAP server. An RA administrator that manages this RA operator is also provided and the privilege for operating RA is separated. A Web interface is provided for

performing these management operations.

- Certificate store `aistore`

This is the certificate store operation command. Certificate verification requires the validation of the path from the trust point and the storage of the root CA certificate and the CRL be stored.

- Certificate verification `certvfy`

The certificate verification command validates certificates using the CA certificate and the CRL located in the aistore.

- Certificate viewer `certview`

Certificates, CRLs and Private keys use many different file formats, which this application can show in as simple plain text. This utility is essential for CA operations.

- Certificate converter `certconv`

The certificate converter transforms the many types of file formats used for certificates and private keys into a mutually readable format. This utility is essential for CA operations.

- CSR creation `certreq` and `grid-certreq`

The `certreq` command creates a key pair and it creates a certificate request based on the PKCS#10 format. The CA reads the PKCS#10 request then issues a certificate. In addition, the `certreq` command works with a CA server or an RA server to manage the certificate lifecycle of remote machines, which includes processing for certificate issuance requests, updates, and revocation. The `grid-certreq` command is a shell script created for use in a grid environment. It calls `certreq` commands internally. Along with providing certificate lifecycle management, the `grid-certreq` command outputs certificates for users in a Globus or a UNICORE environment.

## 1.2. Operating Environment

.....

The NAREGI CA operating environment is shown below.

<b>Compatibility</b>	PC/AT compatible machine (DOS/V) Sun Microsystems Sparc machine
<b>CPU</b>	Pentium III 500 MHz or above recommended
<b>OS compatibility</b>	Turbolinux Server 6.5 Miracle Linux Standard Edition V2.1 Solaris 2.6, 2.8
<b>Memory</b>	128 MB or above recommended
<b>Hard disk capacity</b>	Software installation: 10 MB required Space requirements vary depending on the number of certificates issued. (Example: 1000 certificates requires 10 MB.)
<b>Display</b>	800 x 600 dot resolution, high color or above recommended

## 2. CA and RA Setup

This section describes how to build a CA using NAREGI CA commands. A CA is easy to build, simply execute the `aisetup.sh` or the `ainewca.sh` shell and follow the on-screen instructions.

### 2.1. New CA (with CA Server Registration)

- 1 The `aisetup.sh` shell script is used to build a new CA and to register a CA server. The box below shows the various commands.

#### **aisetup.sh [Option] CA name**

Option:

- add : Builds a CA and registers it with the server (Standard)
- del : Deletes the CA from a server's registration information
- keyalgo **algo** : Specifies the private-key algorithm  
(rsa (Standard), dsa, ecdsa)
- keysize **size** : Specifies the key length to generate(Standard:1024 bits)
- days **day** : Specifies the certificate validity in days
- start **time** : Specifies the start time in "YYYYMMDDHHMMSSZ"
- end **time** : Specifies the termination time in "YYYYMMDDHHMMSSZ"
- p12 **file** : Specifies PKCS#12 and builds a CA
- p11lib **lib** : Specifies the HSM (PKCS#11) library

You can omit the `-add` option. If you specify a CA name then execute the command, a CA information file will be placed below the "NAREGI CA install directory/CA name". In addition, the table below outlines how a CA is built.

Create a new certificate and a private key	The <code>-keysize</code> option specifies the key length when generating a new private key or a new public key and it issues a self-signing certificate. In this example the CA is called the Root CA because it is at the top level.
Import the PKCS#12	The certificate and the private key files (PKCS#12) are placed under the CA Root where they are used to build a CA. In this case the CA becomes a subordinate CA.



- 2 When creating a new certificate and a new private key, execute the command as shown in the box below. When the command is executed, Step 1 for the CA master password begins.

```
bash$ aisetup.sh -keysize 2048 -days 3650 testca
=====
Registration of Certification Authority
=====
Step 1. Initialize CA Master Password.
* This password will be used for CA private key encryption or
* HSM login password. Also, "CAOperator" private key in the
* NAREGI CA certificate store will be encrypted with this password.

Input Master Passwd: (Input password)
Verify - Input Master Passwd: (Input password)
```

- 3 When the CA master password setting is finished, Step 2 for generating the CA private key begins. Wait until the key generation is finished. At this point, the CA information file location is shown and in most cases the data is created under the "NAREGI CA install directory/CA name".

```
Step 2. create a new Certificate Authority
* create a new Certificate Authority for the CA server.
* CA information files are placed in :
* /usr/local/naregi-ca/testca

generate private key (size 2048 bit)
.....0
.....0
```

- 4 When the CA private key generation has completed successfully, the next step is to enter the CA subject.

```
input Distinguished Name (DN).
select directory tag (input number)
(1.C, 2.ST, 3.L, 4.O, 5.OU, 6.CN, 7.Email, 8.Quit)[1]:
Country [JP]: JP
select directory tag (input number)
(1.C, 2.ST, 3.L, 4.O, 5.OU, 6.CN, 7.Email, 8.Quit)[4]: 4
Organization [my organization]:
select directory tag (input number)
(1.C, 2.ST, 3.L, 4.O, 5.OU, 6.CN, 7.Email, 8.Quit)[5]: 5
Organization Unit [business unit]:
select directory tag (input number)
(1.C, 2.ST, 3.L, 4.O, 5.OU, 6.CN, 7.Email, 8.Quit)[8]:
```

The subject has several tag fields, which are described in the table below.

<b>C</b>	This is the country tag. When you enter a country name (for example, Japan is "JP"), make sure to only use one-byte, uppercase alphabet characters.
<b>ST</b>	This is the state tag, which you are not required to use.
<b>L</b>	This is the location tag, which you are not required to use.
<b>O</b>	This is the organization tag. The name can be up to 64 one-byte characters in length and you can also enter Japanese characters.
<b>OU</b>	This is the organizational unit tag. The name can be up to 64 one-byte characters in length and you can also enter Japanese characters.
<b>CN</b>	This is the common nametag. The name can be up to 64 one-byte characters in length and you can also enter Japanese characters.
<b>EMAIL</b>	This is the e-mail tag. You can enter up to 64 characters.

- 5 Check the certificate subject and the period of validity then issue the CA certificate.

```
Certificate DATA:
  serial number: 1
  issuer:
    C=JP, O=my organization, OU=business unit,
  subject:
    C=JP, O=my organization, OU=business unit,
  notBefore: Nov 28 16:00:04 2003
  notAfter: Nov 25 16:00:04 2013

do you sign here ? (y/n)[y]: y
now signing..
.....oo
oo
Update CA information.
```

- 6 Step 3 updates the aica.cnf, which is the configuration file for the CA server. The script automatically executes this update without requiring user input.

Step 3. CA Server registration

- \* new Certificate Authority is created successfully.
- \* now this CA will be registered into the CA Server.
- \* if you want to change default setting, you can do it
- \* manually by editing aica.cnf file with text editor.
- \* aica.cnf file is placed in:
- \* /usr/local/naregi-ca/lib/aica.cnf

import a file to the store successfully.

success to regist a CA: testca

success to regist a CRL Publisher: /usr/local/naregi-ca/testca

success to unregist RAd RegInfo: dummy

success to regist a RAd RegInfo: localhost:testca

-----  
CA server registration has finished successfully.

put "aica" command to start the CA server. Then, you can test the following "aica" command to check if the server works correctly.

- \* aica print -sv localhost:testca -ssl -clid CAOperator001

With this script you can register the CA on a CA server, add CRL publisher information, and add web enrollment information. If the same CA name is already present, you can skip the registration. In addition, currently you can only configure one web enrollment service. Accordingly, when you execute the aisetup.sh script from the second time on, the web enrollment update will not be available. If you want to set web enrollment for a new CA, you must do it manually using the aica.cnf configuration file.

- 7 The CA directory shown in Step 2 contains all the data required for CA operations. Next is a description of each file that a CA contains.

- ca.p12

The ca.p12 is a PKCS#12 file that contains a self-signing certificate and a private key. This file is always read when a CA starts. When building a CA, you will set the Export Password for this ca.p12 file; and when starting a CA, you must always use this password. If the password is entered correctly, you can continue with operations; but if it is entered incorrectly, the program terminates.

- **ca.cer**

This file, which uses a Windows compatible extension name, contains a CA certificate in X.509 PEM format. A user cannot reference a CA certificate in PKCS#12 so only the certificate is prepared separately.

- **ca.cai and profiles**

The ca.cai file contains the profile list maintained by the CA. In addition, cert/\*.cpi contains information for each profile.

- Current serial number
- The issued certificate and its status
- Certificate and CRL period of validity settings
- Extra information added to a certificate and a CRL

You can use the aica command to change this information.

```
bash$ aicad
## Boot the CA Server: input master password for each CA ##
read config file.
...(abridged)...
bash$ aica print -sv localhost:testca -ssl -clid CAOperator001
tring to connect localhost(11411):testca (ssl)
Open Private Key: (Password input)
-----
Certificate Profile: SMIME user
certificate version: 3
current serial number: 1
signature algorithm: md5WithRSAEncryption
...(abridged)...
```

- 8 Once the CA is registered in a CA server, you have successfully completed building it. To check if the settings are going well, start the CA server and try to access it remotely. Bring up the profile display and make sure information like "SMIME user" as shown below is present.

### 2.1.1. Procedure for using HSM

- 1 To use HSM, management of the key depends on the HSM side. Before building CA, set up and initialize HSM in accordance with the HSM product manual.
- 2 After initializing HSM, access HSM from aisetup.sh script to create a private key. At this time, specify the PKCS#11 (Cryptoki) library and label name supplied from the product.

```
bash$ aisetup.sh -p11lib /usr/luna/lib/libcryptoki2.so -p11label myhsm  
-days 3650 testca
```

```
=====
```

Registration of Certification Authority

```
=====
```

Step 1. initialize CA Master Password.

- \* this password will be used for CA private key encryption or
- \* HSM login password. Also, "CAOperator" private key in the
- \* NAREGI CA certificate store will be encrypted with this password.

Input Master Passwd : **(Enter Password)**

Verify - Input Master Passwd : **(Enter Password)**

A user is prompted to input the master password of CA at this time. Depending on the HSM, however, the password may be directly input from the keypad of the HSM, in which case the password is ignored. Even if the password is not used by the CA, it is used for accessing the CAOperator key, so select a safe password.

- 3 The rest of the procedure is the same as that for building a new CA (with CA server registration). Refer to Section 2.1 for details.

## 2.2. New CA (without CA Server Registration)

- 1 Use the `ainewca.sh` shell script to build a CA for local access use The box below shows the various commands.

### **ainewca.sh [Option] CA name**

Option:

- keyalgo **algo** : Specifies the private-key algorithm (rsa (Standard), dsa, ecdsa)
- keysize **size** : Specifies the key length to generate (Standard: 1024 bits).
- days **day** : Specifies the certificate validity in days.
- start **time** : Specifies the start time in "YYYYMMDDHHMMSSZ"
- end **time** : Specifies the termination time in "YYYYMMDDHHMMSSZ"
- p12 **file** : Specifies PKCS#12 and builds a CA
- p11lib **lib** : HSM (PKCS#11) module library
- p11label **name** : PKCS#11 token label for CA private key

The table below outlines how a CA is built.

<b>Create a new certificate and a private key</b>	The -keysize option specifies the key length when generating a new private key or a new public and it issues a self-signing certificate. In this example the CA is called the Root CA because it is at the top level.
<b>Import the PKCS#12</b>	The certificate and the private key files (PKCS#12) are placed under the CA Root where it is used to build a CA. In this case, the CA becomes a subordinate CA.

- 2 When creating a new certificate and a new private key, execute the command as shown in the box below. When you execute the command, key generation will begin.

```
bash$ ainewca.sh -keysize 2048 -days 1095 testca
make new Certification Authority.
generate private key (size 2048 bit)
...O
.....O
input subject directory.
select directory tag (input number)
(1.C, 2.ST, 3.L, 4.O, 5.OU, 6.CN, 7.Email, 8.Quit)[1]:
```

- 3 When key generation has completed, the next step is to enter the certificate subject.

```
select directory tag (input number)
(1.C, 2.ST, 3.L, 4.O, 5.OU, 6.CN, 7.Email, 8.Quit)[1]:
Country [JP]: JP
select directory tag (input number)
(1.C, 2.ST, 3.L, 4.O, 5.OU, 6.CN, 7.Email, 8.Quit)[4]:4
Organization [nitech.ac.jp]: nec corporation
select directory tag (input number)
(1.C, 2.ST, 3.L, 4.O, 5.OU, 6.CN, 7.Email, 8.Quit)[5]:5
Organization Unit []: developer unit
select directory tag (input number)
(1.C, 2.ST, 3.L, 4.O, 5.OU, 6.CN, 7.Email, 8.Quit)[6]:8
```

- 4 Check the certificate subject and the period of validity then issue the certificate. At this time, you will have to enter a password because the CA certificate and the private key are placed in a PKCS#12 file for storage. This password is very important to operate the CA so make sure to remember it.

```
Certificate DATA:
serial number: 0
issuer:
  C=JP, O=nec corporation, OU=developer unit,
subject:
  C=JP, O=nec corporation, OU=developer unit,
notBefore: Jun 06 20:57:58 2002
notAfter: Jun 05 20:57:58 2005
do you sign here ? (y/n)[y]: y
now signing..
Export PKCS#12: (Enter Password)
Verifying - Export PKCS#12: (Enter Password)
.....00
.....00
Update CA information.
import a file to the store successfully.
Succeeded to make new CA !!
```

- 5 Depending on how you execute the command, a "CA name" directory is created. This directory then contains three newly created files: ca.p12, ca.cer, and ca.cai. After checking that these files are present, you have completed building the CA.

## 2.3. New RA (with RA Server Registration)

- 1 Before user enrollment can be enabled you must first build an RA. An RA is built with the `ainewra.sh` shell script using the commands as shown in the box below.

**`ainewra.sh` [Option] CA name**

Option:

**`-op id`** : Specifies the certificate of the CA operator  
**`-pw pwd`** : Specifies the private-key PIN of the CA operator  
**`-sv name`** : Enters the CA server name

To use the `-op` command you must import the operator certificate into the certificate store of the local machine as shown in the procedure below.

Issue an operator certificate from the destination CA server. Use the operator certificate created when the CA was built or use the `aica user -addop` command to issue a new certificate.

Use the `aica export` command to output the certificate to a PKCS#12 formatted file.

Use FTP to copy the certificate to a local machine, and use the `aistore -i` command to import the operator certificate to the local certificate store.

Specify the ID using the `-op` option so you can later check the ID of the imported certificate using the `aistore` command.

For details on this procedure, refer to Section 3.11, Remote Access Setting.

In addition, if you omit the optional `-op` and the `-sv` tags, their space will be left blank in the RAd RegInfo section of the `aica.cnf` configuration file. Accordingly, when you use this command, you must specify all tag options.

- 2 When configuring a new RA server, execute the command as shown in the box below.

```
bash$ ainewra.sh -op CAOOperator001 -sv caserv testca
```

```
-----  
setup a Registration Authority (RA)  
-----
```

```
>> copy RA template files to /usr/local/naregi-ca/testca_ra
```

```
>> update aica.cnf (configuration) file  
success to regist a RAd RegInfo: caserv:testca
```



```
>> input CA Operator access password
    you need to set access password for CAOperator001 in the
    certificate store.
Input Operator Passwd: (Enter Password)
Verify - Input Operator Passwd: (Enter Password)

RA initialization has been finished successfully.
```

If the destination CA name is "testca," a directory called "testca\_ra" will be located below the NAREGI CA installation directory, and this is where the files required to run the web enrollment service and the airad daemon will be located.

In addition, you can build up to 64 RAs on one server.

- 3 To use a web enrollment service, add the lines shown in the box below to the httpd.conf configuration file. You can add these lines manually using a text editor.

```
Alias /testca_ra/img "/usr/local/naregi-ca/testca_ra/img"
<Directory "/usr/local/naregi-ca/testca_ra/img">
    Options Indexes MultiViews
    AllowOverride None
    Order allow,deny
    Allow from all
</Directory>

ScriptAlias /testca_ra "/usr/local/naregi-ca/testca_ra/cgi"
<Directory "/usr/local/naregi-ca/testca_ra/cgi">
    AllowOverride None
    Options None
    SSLOptions +ExportCertData +StdEnvVars
    Order allow,deny
    Allow from all
</Directory>
```

In the example above, a web alias is created for testca\_ca, which is the previously created RA. A CGI script is placed below the /testca\_ra directory and image data is placed below the /testca\_ra/img directory. It is now possible for a user to access the CGI script through [http://raname/testca\\_ra/aienroll](http://raname/testca_ra/aienroll) and use the certificate enrollment service.

For details on this service, refer to Section 3.3, Web Enrollment Setting.

## 2.4. PKCS#12 for CA

.....

- 1 After receiving a certificate issued from the root CA, you then use a private key and a certificate (PKCS#12 formatted file) to build a CA. In this case, you can use the `certreq` command to create a key pair by yourself and request a certificate for the root CA. In addition, you can also have all key pairs and certificates distributed by the root CA.
- 2 Specify the PKCS#12 file and execute the command as shown in the box below. Enter the password one time and check to see if it can be used with the CA certificate.

```
bash$ ainewca.sh -p12 sub-ca.p12 testca
make new Certification Authority.
use pkcs#12 file "ca.p12" as CA certificate.
Enter master CA password: (Enter Password)
CA PKCS#12 file open
.....00
.....00
Update CA information.
import a file to the store successfully.
success to make a new CA !!
clean your "ca.p12" before using new CA. It is more secure for
new CA :-)
```

- 3 Depending on how you execute the command, a "CA name" directory is created. This directory then contains three newly created files: `ca.p12`, `ca.cer`, and `ca.cai`. After checking that these files are present, you have completed building the CA.

## 2.5. Updating CA Certificate

- 1 To change the expansion information or period for which a CA certificate is valid by using the same key pair, the CA certificate can be updated by using “aica renew”. To update a key, build a new CA.

### **aica renew [Option]**

Option:

- self: Updates the Root CA certificate (default).
- p10: Creates PKCS#10 by using CA key (for Sub-CA).
- days **day**: Specifies the number of valid days of the certificate.
- start **time**: Specifies start time in "YYYYMMDDHHMMSSZ".
- end **time**: Specifies end time in "YYYYMMDDHHMMSSZ".

- 2 To change the valid period of a CA certificate, execute as follows.

```
bash$ aica renew -start 20010101 -end 20080401
CA PKCS#12 file open
Input PKCS#12 Password: (Enter Password)
do you modify CA certificate extension ? (y/n)[n]:(Return)

Certificate DATA:
serial number : 1
issuer :
    C=JP, O=my organization, OU=business unit,
subject:
    C=JP, O=my organization, OU=business unit,
notBefore: Jan 01 09:00:00 2001
notAfter : Apr 01 09:00:00 2008

do you sign here ? (y/n)[y]: (Return)
now signing ..
Enter master CA password: (Enter Password)
Verifying - Enter master CA password:(Enter Password)
```

- 3 To create a PKCS#10 file by using a CA key pair, execute as follows.

```
bash$ aica renew -p10
CA PKCS#12 file open
Input PKCS#12 Password: (Enter Password)
save a ca pkcs10 file (ca.p10) ... done.
```

## 2.6. aica.cnf Setting

The aica.cnf file is where you configure the operating settings for the CA server, the CRL publisher, the web enrollment service, the certificate store location, and the default subject. It is a text file that can be changed manually or through scripts using the aiconftool utility. The location of this file is normally “NAREGI CA installation directory/lib/aica.cnf.”

- General Info Setting


The general info setting is for configuring the certificate store directory that is used to save and to verify certificates.

```
[general info]
store_dir=/usr/local/naregi-ca/store
salt_val=BsEWel12dsfFUf14dWBs7EsfaUYs5
[general info end]
```

The location of the certificate store is normally “NAREGI CA installation directory/store/,” and the full path is specified in the store\_dir field.

Each field is described in the table below.

<b>store_dir</b>	Specifies the certificate store directory that NAREGI CA will use. Up to 254 characters can be used.
<b>salt_val</b>	The salt value is used when encrypting a password. For automatic startup, the password can be saved in the aica.cnf configuration file. Accordingly, this setting is for the part of the key that is encrypted. Up to 62 characters can be used.

 In addition, the setting string begins after the “=” sign and continues until a line feed. Do not place a “¥” mark at the end of the directory name; and keep in mind that a “#” mark at the beginning of a string is interpreted as a comment.

- CAd Setting

This section describes the operating settings for a CA server

```
[CAd]
capath.0      =/usr/local/naregi-ca/testca
caname.0      =testca
capwd.0       =$aicry${l2SrquYkB80fNYU5mBUhJg==}
```

sv_id	=caserver0100
sv_id_pwd	=\$aicry\${4gK6liQryPcWoQCG5f16PA==}
sv_port	=11411
sv_listen	=5
f_ssl_use	=true
f_ssl_reqcert	=true
f_ssl_novfycrl	=true
ssl_timeout	=600
errlog	=/usr/local/naregi-ca/logs/cad_error.log
isslog	=/usr/local/naregi-ca/logs/cad_issue.log
acclog	=/usr/local/naregi-ca/logs/cad_access.log

Each field is described in the table below.

<b>capath.X</b>	Sets the CA directory so that a CA server can be operated remotely. Up to 254 characters can be used. The .X is replaced by a numerical value and it is possible to register up to 64 CAs.
<b>caname.X</b>	Sets the CA name for a CA server so that it can be operated remotely. Up to 32 characters can be used. The .X is replaced by a numerical value and it is possible to register up to 64 CAs.
<b>capwd.X</b>	Starting a CA server requires the input of each CA startup password. Setting this field enables a CA server to start automatically.
<b>sv_id</b>	Specifies an SSL server certificate for a CA server. To use an SSL server certificate, you must first save it in an NAREGI CA store and then specify its store ID.
<b>sv_id_pwd</b>	This password is required to activate an SSL server certificate when a CA server starts up. Setting this field enables a CA server to start automatically.
<b>sv_port</b>	Specifies the port number for a CA server, which is normally port 11411.
<b>sv_listen</b>	Specifies the number of simultaneous CA server connections (listen).
<b>f_ssl_use</b>	Specifies whether SSL is required to connect to a CA server. Specify by using <b>true</b> or <b>false</b> .
<b>f_ssl_reqcert</b>	Specifies whether SSL client authentication is required to connect to a CA server. Specify by using <b>true</b> or <b>false</b> .
<b>f_ssl_novfycrl</b>	Specifies whether a CRL is required when an SSL client is authenticated through a CA server connection. Specify by using <b>true</b> or <b>false</b> . Normally, a CRL is not required (setting = true) because user registration for a CA server is handled separately.
<b>ssl_timeout</b>	Sets the timeout for a CA server connection. The default setting is 600 seconds.

<b>errlog</b>	Specifies the name for an error log. Up to 254 characters can be used.
<b>isslog</b>	Specifies the name for an issuance log. Up to 254 characters can be used.
<b>acclog</b>	Specifies the name for an access log. Up to 254 characters can be used.
<b>errlog_!otat</b>	Specifies the rotate size for an error log.
<b>isslog_!otat</b>	Specifies the rotate size for an issuance log.
<b>acclog_!otat</b>	Specifies the rotate size for an access log.

- Default CA Setting

This section describes the operating settings for the aica command.

```
[default CA]
ca_dir      =.
ca_port     =11411
#ca_pwd     =abcde

#cl_id      =
#cl_id_pwd  =

#f_ssl_use  =false
#f_ssl_novfcr =true
[default CA end]
```

Under the default settings, the CA information is read from the current directory. If you want to issue a certificate from CA information in a specific directory, change the setting so that it has a “ca\_dir=/home/myname/myca” format. If you want to connect to a specific remote CA, change the setting so that it has a “ca\_dir=servername:caname” format.

Each field is described in the table below.

<b>ca_dir</b>	Specifies the CA directory that has access to the aica command. Up to 254 characters can be used. Normally “.” specifies the current directory. Use “servername:caname” to specify a remote CA.
<b>ca_port</b>	Specifies the port number for a CA server, which is normally port 11411.
<b>ca_pwd</b>	Sets the CA startup password. Setting this field enables a CA to start automatically.
<b>dcl_id</b>	Specifies an SSL client certificate. To use an SSL client certificate, you must first save it in the NAREGI CA store and then specify its store ID.

<b>cl_id_pwd</b>	Entering a password is required to activate an SSL client. Setting this field enables an automatic connection to a CA server.
<b>f_ssl_use</b>	Specifies whether SSL is required to connect to a CA server. Specify by using <b>true</b> or <b>false</b> .
<b>f_ssl_novfycrl</b>	Specifies whether a CRL is required when checking an SSL server certificate through a CA server connection. Specify by using <b>true</b> or <b>false</b> .

- CRL Publisher Setting

This section describes the operating settings for a CRL publisher.

```
[CRL Publisher 0]
ca_dir    =localhost:testca
ca_port   =11411
ca_uid    =caadmin
#ca_pwd   =$aicry${NjkH5y/+x1ALL51Zxh+r/Q==}

cl_id      =CAOperator001_00
cl_id_pwd  =$aicry${yIx4lCqRSvtRmeVSPonzWQ==}
f_ssl_use  =true
f_ssl_novfycrl =true

# working interval (seconds)
interval   =3600
f_exp_mode =true

tm_start   =20030401000000Z
out_dir    =/usr/local/naregi-ca/testca_ra/cgi

#ldap_host =
ldap_port  =389
ldap_base  =ou=testca,o=test,c=JP
ldap_bind  =cn=ldapAdministrator,c=JP
ldap_pwd   =
ld_crl_attr =certificateRevocationList;binary
ld_crl_prof =CRL-All
ld_arl_attr =authorityRevocationList;binary
ld_arl_prof =ARL

errlog     =/usr/local/naregi-ca/logs/pub_error.log
isslog     =/usr/local/naregi-ca/logs/pub_issue.log
errlog_lotate =2048
isslog_lotate =2048
[CRL Publisher 0 end]
```

One CRL publisher can operate in one section. It is possible to set multiple publishers by separating each one using the [CRL Publisher X] tag and the [CRL Publisher X end] tag. "X" has no particular upper-limit value.

A CRL publisher outputs an ARL and a CRL to a specified directory or to an LDAP server. In particular, if outputting to a local directory, three files are generated: out-ARL.crl, out-CRL.crl, and out-CRL-All.crl.

Each field is described in the table below.

<b>ca_dir</b>	Specifies the CA directory that has access to the CRL publisher. Up to 254 characters can be used. Use "servername:caname" to specify a remote CA.
<b>ca_port</b>	Specifies the port number for a CA server, which is normally port 11411.
<b>ca_pwd</b>	Sets the CA startup password. Setting this field enables a CA to start automatically.
<b>cl_id</b>	Specifies an SSL client certificate. To use an SSL client certificate, you must first save it in the NAREGI CA store and then specify its store ID.
<b>cl_id_pwd</b>	Entering a password is required to activate an SSL client. Setting this field enables an automatic connection to a CA server.
<b>f_ssl_use</b>	Specifies whether SSL is required to connect to a CA server. Specify by using <b>true</b> or <b>false</b> .
<b>f_ssl_novfycrl</b>	Specifies whether a CRL is required when checking an SSL server certificate through a CA server connection. Specify by using <b>true</b> or <b>false</b> .
<b>interval</b>	Sets the operating interval for a CRL publisher. Normally the interval is specified the same as the period of validity.
<b>f_exp_mode</b>	Specifies the CRL issuance mode. If the value is set to false, a CRL will be issued even if it is within the period of validity. Normally, specify as "true."
<b>tm_start</b>	Sets the startup time for a CRL publisher. If the CRL operating interval and the CRL period of validity are the same, it is possible to set a CRL publisher so that it starts directly after the period of validity expires.
<b>out_dir</b>	Specifies the CRL output directory. Up to 254 characters can be used.
<b>ldap_host</b>	Specifies the LDAP server where a CRL is generated. Setting this field will disable the out_dir field. Up to 254 characters can be used.
<b>ldap_port</b>	Specifies the port number for an LDAP server, which is normally port 389.
<b>ldap_base</b>	Indicates the distinguished name (DN) of the entry that outputs a CRL. Specify a CA entry. Up to 254 characters can be used.
<b>ldap_bind</b>	The user name that is used to connect to an LDAP server. Connects using a simple bind. Up to 254 characters can be used.



<b>ldap_pwd</b>	The password used to connect to an LDAP server. Up to 30 characters can be used.
<b>ld_crl_attr</b>	Indicates the attribute name of the entry that outputs a CRL. Up to 64 characters can be used.
<b>ld_crl_prof</b>	Indicates a CRL profile name. Normally, specify as "CRL-ALL."
<b>ld_arl_attr</b>	Indicates the attribute name of the entry that outputs an ARL. Up to 64 characters can be used.
<b>ld_arl_prof</b>	Indicates an ARL profile name. Normally, specify as "ARL."
<b>errlog</b>	Specifies the name for an error log. Up to 254 characters can be used.
<b>isslog</b>	Specifies the name for an issuance log. Up to 254 characters can be used.
<b>errlog_lotate</b>	Specifies the rotate size for an error log.
<b>isslog_lotate</b>	Specifies the rotate size for an issuance log.

- RAd settings

The RAd operating settings are shared among three modules: the airad, the enroll cgi, and the aienroll.

```
[RAd]
sv_id           =caserver0100
sv_id_pwd       =$aicry${4gK6liQryPcWoQCG5f16PA==}
sv_port         =11411
sv_listen       =5

f_ssl_use       =true
f_ssl_optreq    =true
#f_ssl_reqcert  =true

f_ssl_novfycrl  =true

ssl_timeout     =600

acclog          =/usr/local/naregi-ca/logs/enroll_access.log
errlog          =/usr/local/naregi-ca/logs/enroll_error.log
isslog          =/usr/local/naregi-ca/logs/enroll_issue.log

acclog_lotate   =2048
errlog_lotate   =2048
isslog_lotate   =2048
[RAd end]
```

Use the RAd settings to configure each operating item for the airad server and to set the log file output that is shared among the airad, the enroll cgi, and the aienroll modules.

Each field is described in the table below.

<b>sv_id</b>	Specifies the SSL server certificate for an airad daemon. To use an SSL server certificate, you must first save it in the NAREGI CA store and then specify its store ID.
<b>sv_id_pwd</b>	This password is required to activate an SSL server certificate when an airad daemon starts up. Setting this field enables a CA server to start automatically.
<b>sv_port</b>	Specifies the port number for an airad daemon, which is normally port 11412.
<b>sv_listen</b>	Specifies the number of simultaneous airad connections (listen).
<b>f_ssl_use</b>	Specifies whether SSL is required to connect to an airad daemon. Specify by using <b>true</b> or <b>false</b> .
<b>f_ssl_optreq</b>	Specifies client authentication as an option for an airad daemon over an SSL connection. Specify by using <b>true</b> or <b>false</b> .
<b>f_ssl_reqcert</b>	Specifies whether SSL client authentication is required to connect to an airad daemon. Specify by using <b>true</b> or <b>false</b> . This has priority over the f_ssl_optreq field.
<b>f_ssl_novfycrl</b>	Specifies whether a CRL is required when an SSL client is authenticated through an airad daemon connection. Specify by using true or false.
<b>ssl_timeout</b>	Sets the timeout for an airad daemon connection. The default setting is 600 seconds.
<b>errlog</b>	Specifies the name for an error log. Up to 254 characters can be used.
<b>isslog</b>	Specifies the name for an issuance log. Up to 254 characters can be used.
<b>Acclog</b>	Specifies the name for an access log. Up to 254 characters can be used.
<b>errlog_lotate</b>	Specifies the rotate size for an error log.
<b>isslog_lotate</b>	Specifies the rotate size for an issuance log.
<b>acclog_lotate</b>	Specifies the rotate size for an access log.

- RAd RegInfo settings

The RAd RegInfo operating settings are shared among three modules: the airad, the enroll cgi, and the aienroll.

```
[RAd RegInfo 0]
raname  =testca_ra
rapath  =/usr/local/naregi-ca/testca_ra

ca_dir   =localhost:testca
ca_port  =11411
ca_uid   =caadmin
ca_pwd   =

cl_id    =CAOperator001_00
cl_id_pwd  =${aicry}${Rkom+NUqRTn19Kwm1oalzg==}

f_ssl_use      =true
f_ssl_novfycrl =true

interval       =60
post_mode      =false

authmode       =0
wwwpwd         =/usr/local/naregi-ca/testca_ra/en.passwd
wwwlicense     =/usr/local/naregi-ca/testca_ra/en.license
wwwsessions    =/usr/local/naregi-ca/testca_ra/sessions.0

#ldap_host     =
ldap_port      =389
ldap_base      =
ldap_user_attr =cn

smtp_host      =
smtp_port      =25
admin_email    =
web_address    =http://localhost/aienroll

#gridmap       =/usr/local/naregi-ca/testca_ra/grid-mapfile

groupname.0    =SMIME user
groupprof.0    =SMIME user
#groupbase.0   =ou=hoge0 unit,o=hoge0,c=JP
#grouphost.0   =

#groupname.1   =testgrp2
#groupprof.1   =SMIME user2
#groupbase.1   =o=test,c=JP
#grouphost.1   =ldapserver
[RAd RegInfo 0 end]
```

Each field is described in the table below.

<b>raname</b>	Sets the RA name. Up to 30 characters can be used.
<b>rapath</b>	Specifies the directory where the RA settings file is stored using a direct path. Up to 254 characters can be used.
<b>ca_dir</b>	Specifies the remote CA that has access to the enroll module. Specify as "servername:caname" using up to 254 characters.
<b>ca_port</b>	Specifies the port number for a CA server, which is normally port 11411.
<b>ca_uid</b>	The user name that is used to connect to a CA server. Normally this field is not used because SSL client authentication is performed.
<b>ca_pwd</b>	The user password that is used to connect to a CA server. Normally this field is not used because SSL client authentication is performed.
<b>cl_id</b>	Specifies an SSL client certificate. To use an SSL client certificate, you must first save it in the NAREGI CA store and then specify its store ID.
<b>cl_id_pwd</b>	Entering a password is required to activate an SSL client. Setting this field enables an automatic connection to a CA server.
<b>f_ssl_use</b>	Specifies whether SSL is required to connect to a CA server. Specify by using <b>true</b> or <b>false</b> .
<b>f_ssl_novfycrl</b>	Specifies whether a CRL is required when checking an SSL server certificate through a CA server connection. Specify by using <b>true</b> or <b>false</b> .
<b>interval</b>	Specifies the aienroll-operating interval. The aienroll module periodically connects to a CA server and checks the CSR queue status.
<b>post_mode</b>	Specifies whether a CA operator will validate and issue a certificate for a request stored in a CA queue. A "true" value specifies validation and issuance by an operator and a "false" value specifies immediate issuance.
<b>offline_ca_mode</b>	Specifies whether application for a certificate is stored in a RA queue as a file, and whether the certificate is confirmed and issued by the RA operator. In this mode, the certificate must be manually issued from a CA and a command for locating the certificate must be executed.
<b>authmode</b>	Specifies whether to authenticate a user's certificate request. Use one of the following integers to specify the mode. 0: Issues a certificate to an anonymous user. 1: Authenticates a user based on an ID/password. 2: Authenticates a user based on a license ID (one-time license).

	4: Authenticates a user based on LicenseID/ChallengePIN. In this mode, an LDAP server must be specified.
<b>wwwpwd</b>	Specifies a passwd file when a certificate request requires ID/password authentication. Up to 254 characters can be used. In anonymous mode, this value is ignored. When an ldap_host has been specified, this value is ignored.
<b>wwwlicense</b>	Specifies a license file when a certificate request requires license/ID authentication. Up to 254 characters can be used. In anonymous mode, this value is ignored.
<b>wwwsessions</b>	Specifies the file where RA session information is stored. Up to 254 characters can be used. This file is normally located in the rapath directory.
<b>ldap_host</b>	Specifies an LDAP server when a certificate request requires ID/password authentication. Up to 254 characters can be used. In anonymous mode, this value is ignored.
<b>ldap_port</b>	Specifies the port number for an LDAP server, which is normally port 389.
<b>ldap_base</b>	Indicates the distinguished name (DN) entry for the search base of the LDAP server. Up to 254 characters can be used.
<b>ldap_user_attr</b>	Specifies the attribute type used to search for a user name. Up to 32 characters can be used.
<b>smtp_host</b>	Specifies the SMTP server used to post issuance information. Up to 254 characters can be used.
<b>smtp_port</b>	Specifies the port number for an SMTP server, which is normally port 25.
<b>admin_email</b>	Species the mail address of a CA operator. Up to 126 characters can be used. If a CSR is accepted in post mode, then a CA operator will send a notification.
<b>web_address</b>	Specifies the URL used to obtain a certificate after being notified of its issuance. Up to 254 characters can be used.
<b>email_sbfilter</b>	Specifies whether the e-mail message input by the user is to be included in the certificate. The input e-mail address is saved in LDAP or sessions.0 and is used to notify of updating. 0...Includes in certificate subject DN. 1...Includes in certificate expansion information SubjectAltName. 2...Does not include in certificate.
<b>notice_update</b>	Transmits an e-mail message notifying of updating of the certificate. When time of expiration of the valid period of the certificate minus the time specified in this field (units: hours) is reached, updating is reported to the user address included in sessions.0.
<b>gridmap</b>	Outputs a gridmap file compliant with Globus, a grid middleware application. Up to 254 characters can be used.

	If a certificate is issued using an ID/password, the ID-subject DN is mapped; and if issued using a license ID, the license ID-subject DN is mapped.
<b>gridcertpath</b>	Specifies a directory to which a certificate that has been issued for grid middleware UNICORE is to be output. Up to 256 characters are valid. To enable this value, be sure to specify the gridmap file, also.
<b>groupname.X</b>	Specifies the group name, which can use up to 30 characters. If linked with LDAP, a domain name also appears.
<b>groupprof.X</b>	Specifies the profile name to apply to a group. Up to 30 characters can be used.
<b>grouphost.X</b>	If linking with LDAP, specifies an LDAP server. Up to 254 characters can be used.
<b>groupbase.X</b>	If linking with LDAP, specifies the base DN for a user entry. Up to 254 characters can be used.

- Subject DN setting

Set the default value for the subject information that is used when building a new CA and when executing the certreq command. You can specify one value for each attribute type (C, O).

```
[subject DN]
C      =JP
#ST    =tokyo
#L     =location
O      =test org
#OU    =test unit
CN     =test
EMAIL  =test@localhost
[subject DN end]
```

## 2.7. Setting gridmap.cnf

The gridmap.cnf file sets the operation of gridmapgen. This file is usually located in “NAREGI CA install directory/lib/gridmapgen.cnf”.

- Setting of Grid MapGen

Set the operating interval of gridmapgen and the output destination of the log.

```
[Grid MapGen]
interval          =20

usermap           =/usr/local/naregi-ca/gridmap/users.csv
gridmap           =/usr/local/naregi-ca/gridmap/grid-mapfile

uudb_bin          =/usr/local/unicore/UUDB/bin

#proxy_host       =server
#proxy_port       =10080

refmap.0          =http://raserver/grid/grid-mapfile
refcerts.0        =http://raserver/grid/certs

refmap.1          =http://raserver/grid2/grid-mapfile2
refcerts.1        =http://raserver/grid2/certs

acclog            =/usr/local/naregi-ca/logs/map_access.log
errlog            =/usr/local/naregi-ca/logs/map_error.log

acclog_lotate     =2048
errlog_lotate     =2048
[Grid MapGen end]
```

Each field is described in the table below.

<b>interval</b>	Sets the operation interval (in seconds) of gridmapgen.
<b>usermap</b>	Specifies the mapping file of a local user and a global user (license ID). Up to 254 characters are valid.
<b>gridmap</b>	Specifies the output destination of grid-mapfile for Globus. Up to 254 characters are valid. This file is also used when UNICORE UUDB is updated, so be sure to specify an output file.
<b>uudb_bin</b>	Specifies a directory that includes the command of UNICORE UUDB. Up to 254 characters are valid. Be sure to specify an absolute path. The Java command must be installed in order to execute the command of UUDB,.
<b>proxy_host</b>	Specifies a host name if a proxy is necessary for obtaining a global map file from an RA server (Web) by refmap.X. Up to 126 characters are valid.

<b>proxy_port</b>	Specifies a port number if a proxy is necessary for obtaining a global map file from an RA server (Web) by refmap.X.
<b>refmap.X</b>	Specifies an RA server (Web) in the form of a URL, to obtain a global map file. Up to 126 characters are valid. Up to 32 sites can be specified.
<b>refcert.X</b>	Necessary for adding a certificate to UNICORE UADB. A certificate file is located on the RA server (Web). Specify its higher directories in the form of a URL. Up to 126 characters are valid. Up to 32 sites can be specified.
<b>errlog</b>	Specifies the name of a file to which an error log is to be output. Up to 254 characters are valid.
<b>acclog</b>	Specifies the name of a file to which an access log is to be output. Up to 254 characters are valid.
<b>errlog_lotate</b>	Specifies the rotate size of an error log. If 0 is specified, rotation is disabled.
<b>acclog_lotate</b>	Specifies the rotate size of an access log. If 0 is specified, rotation is disabled.



## 3. CA and RA Server Operations

This chapter describes how to start the NAREGI CA server.

### 3.1. Start and Stop CA Server

.....

- Start CA Server

Once you start the CA server, immediately execute the `aicad` command. If the `aica.cnf` file and the CA structure are configured correctly, the CA server start information will appear as shown in the box below.

```
bash$ aicad
## Boot the CA Server: input master password for each CA ##
read config file.
port=11411,listen=5
ssl=1,req=1,vfy=9
read server certificate: O=test.naregi.jp, OU=server-c7f914-97aa76,
CN=caserver0100,
set certificate request option.
start aicad daemon process (5398)
```

After you execute the `aicad` command, follow the procedure in “CA Setting” of Section 2.5 to register the CA for remote operations. If a password string is present in the “`capwd.X=`” field, then automatic start and registration is enabled for the CA. If a password string is not present, then you will have to enter the master password each time the CA starts up.

Under default settings, connecting to the CA server requires SSL client authentication. Accordingly, an SSL server certificate, which is automatically generated during the NAREGI CA installation, is required on the CA server side. This server certificate is located in the NAREGI CA certificate store and encrypted using its previously set password. In the `aica.cnf` file, the `sv_id` field is configured with the SSL server certificate ID and the `sv_id_pwd` is configured with the password, so when the CA server starts, it is ready to automatically obtain an SSL server certificate. For details on the SSL server certificate, refer to Section 3.7, Automatically Generate SSL Server Certificate.

If the CA registers successfully, the daemon will start. As shown in the box above, the daemon process ID is 5398. The daemon listens for connections on the specified port (11411) and starts a subprocess when a connection is accepted. After the subprocess negotiates an SSL handshake the CA can operate. Only one daemon process can run at one time because the CA server occupies the specified port.

- Stop CA Server

To stop a CA server, use the kill command and the process will immediately stop.

```
bash$ kill 5398
bash$
```

Actually, the only process that stops is the parent process. If a subprocess has executed some other CA operation, the subprocess will not receive the kill command and it will continue processing until completion.

- Execution Access Rights

When the CA server starts it inherits user rights from the aicad daemon. If a general user executes the aicad daemon, that user must have access to the /tmp/aica/, the certificate store, and the CA directories.

In addition, you can execute the aicad daemon from the root directory; but from a network security standpoint, that is not wise. A better method is to create users for the CA server and assign CA building rights and operating rights depending on each user.

- Remote Access Control

The CA server has two ways to access files: locally and remotely. Local file access has no special access controls. If the password is entered correctly, a user will have access to all CA operations. For remote access a user must negotiate through user authentication based on the ca.passwd and access controls.

Normally, an operator is issued a certificate, which is used to authenticate as an SSL client and to connect to a CA server. Once connected, access control is provided by the lightweight certificate management protocol (LCMP). LCMP has the operation and access control functions shown below.

BindRequest	: Connection request, which is normally required to be on
SignRequest	: Issuance request (Immediate)
ListRequest	: Issuance list request
ProfRequest	: Profile request (General and extension information)
CertRequest	: User certificate (update, revocation, output, key storage, and key output)
CsrRequest	: CSR queue (CSR submission, CSR approval, and CSR rejection)
CrlRequest	: CRL request
ServOpRequest	: Server operation request
ExtendedReq	: Extended request

Two types of operators manage a CA server. A master operator starts the CA server and specifies CA operator rights and a general operator administers within the limits of the access controls shown above. (A general operator also administers the web enroll CGI module.)

- Log File Output

A CA server outputs a log file as described in “CA Setting” of Section 2.5. Normally, an access log (cad\_access.log), an issuance log (cad\_issue.log), and an error log (cad\_error.log) are generated and output to a user-specified directory. The content of each log is described in the table below.

<b>Access log</b>	Outputs all the operation requests made to a CA server and the request results. Contains the date/time, the session ID, the connection host name, the CA user, the CA name, and the operation content.
<b>Issuance log</b>	Outputs information when a certificate is issued or updated and when a CRL is issued. Contains the date/time, the session ID, the connection host name, the CA user, the CA name, and the operation content.
<b>Error log</b>	Outputs all operation and system errors that occur on a CA server. Contains the date/time, the session ID, the connection host name, the CA user, the CA name, operation content, and the error number of the cryptographic library.

In addition, you can rotate the size of a log file, which will add a date/time tag (for example: cad\_access.log.20031127111008) to the log file name. It is recommended that you back up these log files to a separate medium.

### 3.2. Start and Stop CRL Publisher

- CRL Publisher Commands

The box below shows the various commands.

```
aicrlpub [Option]
Option:
  -start time      : Specifies the start time in
                     "YYYYMMDDHHMMSSZ" (In GMT)
  -end time        : Specifies the termination time in
                     "YYYYMMDDHHMMSSZ" (In GMT)
  -s section       : Sets the aica.cnf configuration number (Default is 0)
```

A CRL publisher sets one section per process as a base for operations. Normally, one section describes one CRL output setting of a CA. Accordingly, when running a CRL publisher you must start the processes for all registered CAs.

- Start CRL Publisher

Once you start the CA publisher, immediately execute the aicrlpub command. If the aica.cnf file and the CA structure are configured correctly, the CA publisher start information will appear as shown in the box below.

```
bash$ aicrlpub -s 1
CA PKCS#12 file open
start aicrlpub daemon process (5496)
```

When you execute the aicrlpub command, start the CRL publisher according to the setting in the "CRL.Publisher.X." If a password string is present in the "ca\_pwd.X=" field, then automatic start and registration is enabled for the CRL publisher. If a password string is not present, then at each startup you will have to enter the master password of the access CA or the password of the CA operator's private key. The CRL publisher runs one process per section so it is possible to start multiple processes.

The CRL publisher accesses the CA at regular intervals to check certificate status and to publish a CRL. In addition, for access to a remote CA, the CRL makes an issuance request. When the CRL publisher is not running, it waits in a sleep status that is not a burden on CPU resources. Accordingly, the CRL publisher does not accept outside input, so it will continue to run until it receives a stop signal or until the specified finish time.

- **Stop CRL Publisher**

To stop a CRL publisher, use the kill command and the process will immediately stop.

```
bash$ kill 5496
bash$
```

A CRL publisher does not generate any subprocess but it is possible to run multiple publishers at one time. If you view processes going on with the ps command, you may see that a number of publishers are running. If you find that a number of publishers are running, you must kill each one manually.

- **Execution Access Rights**

When the CA publisher starts it inherits user rights from the aicrlpub daemon. If a general user executes the aicrlpub daemon, that user must have access to the /tmp/aica/, the certificate store, and the CA directories.

In addition, you can execute the aicrlpub daemon from the root directory; but from a network security standpoint, that is not wise. A better method is to create users for the CA publisher and assign CA building rights and operating rights depending on each user.

- **Log File Output**

A CA publisher outputs a log file as described in “CRL Publisher Setting” of Section 2.5. Normally, an issuance log (pub\_issue.log) and an error log (pub\_error.log) are generated and output to a user-specified directory. The content of each log is described in the table below.

<b>Issuance log</b>	Outputs information when a CRL is issued. Contains the date/time, the session ID, the CA name, and the output destination CRL.
<b>Error log</b>	Outputs all operation and system errors that occur on a CRL publisher. Contains the date/time, the session ID, the CA name, the error content, and the error number of the cryptographic library.

In addition, you can rotate the size of a log file, which will add a date/time tag (for example: pub\_access.log.20031127111008) to the log file name. It is recommended that you back up these log files to a separate medium.

### 3.3. Web Enrollment Setting

- Web Enrollment Overview

NAREGI CA operates through a web server to issue and revoke user certificates and it manages certificate lifecycles through a web enrollment function. To enable web enrollment, NAREGI CA has two special modules. The `aienroll.cgi` module is normally installed in the `rapath/cgi/aienroll` directory. When this CGI is called on a web server like Apache, a user can issue, revoke, and update a certificate. The `aienroll` module (`bin/aienroll`) periodically accesses the CA server and checks the status for user certificate issuance and revocation.

An RA operation CGI module that is used in the ChallengePIN/LicenseID authentication mode (`authmode = 4`) is also available. The following CGI are available.

CGI Name	URL	Role
<code>airaadmin.cgi</code>	<code>https://~/CA-name_ra/airaadmin</code>	Management screen for RA administrator. Displays a list of RA operators, a list of applications, and performs various operations.
<code>airaregist.cgi</code>	<code>https://~/CA-name_ra/airaregist</code>	Supplies an application screen for the RA operator.
<code>airaenroll.cgi</code>	<code>https://~/CA-name_ra/airaenroll</code>	Supplies a certificate issuance screen for the RA operator.
<code>airaop.cgi</code>	<code>https://~/CA-name_ra/airaop</code>	Management screen for the RA operator. Displays a list of users, a list of applications, and performs various operations.
<code>airegist.cgi</code>	<code>https://~/CA-name_ra/airegist</code>	Supplies an application screen for the user.
<code>aienroll.cgi</code>	<code>https://~/CA-name_ra/aienroll</code>	Supplies a certificate issuance screen for the user. Also used to issue certificates in other authentication modes.

Operations in this mode are explained in a separate manual. For details, refer to that manual.

One other module that checks enrollment (`bin/aienroll`) to notify the POST mode or updating is available, which periodically accesses the CA server to check expiration of a valid period and sends various notices to the user. For details of enrollment check, refer to Section 3.6, Starting and Stopping Notification of Updating.

Along with two modules for certificate issuance, web enrollment has five authentication levels. Using these modules and levels in different combinations

enables you to manage certificate lifecycles in a flexible manner.

The certificate issuance modules are described below.

<b>Immediate issuance</b>	<p>A user accesses the web enroll CGI and enters the user information.</p> <p>A CSR based on the user input information is generated and posted to the CGI. With immediate issuance, after the CGI makes an issuance request to the CA, it is possible to install a certificate onto a user machine at once.</p>
<b>Validation and issuance by an operator</b>	<p>A user accesses the web enroll CGI and enters the user information. A CSR based on the user input information is generated and posted to the CGI. With validation and issuance by an operator, the CGI posts a CSR to the CSR queue of the CA and the process finishes when the user receives an AcceptID.</p> <p>Next, the CA operator accepts or rejects the certificate request in the CSR queue. The aienroll module checks the operation result and sends an e-mail to the user.</p> <p>If a certificate has been issued, the user will receive a certificate acquisition e-mail with a URL that is used to access the web enroll CGI and install the certificate on the user's machine.</p>

The authentication levels are described below.

<b>Anonymous authentication (authmode=0)</b>	<p>Issues a certificate without authenticating the user. With this level, a certificate is issued immediately without updating and revocation.</p>
<b>ID/password authentication (authmode=1)</b>	<p>A user must enter an ID and a password to be issued a certificate. With this authentication level, both immediate issuance and validation and issuance by an operator are available. In addition, updating and revocation are handled by ID/password authentication or by SSL client authentication.</p> <p>You can set this authentication level by setting the ID/password in the local testca_ra/en.passwd file or by connecting to an external authentication server like an LDAP server.</p>
<b>License ID authentication (One-time license) (authmode=2)</b>	<p>A user must enter a license ID to be issued a certificate. One license ID can be used to issue only one certificate. With this authentication level, both immediate issuance and validation and issuance by an operator are available. In addition, updating and revocation are handled by SSL client authentication.</p> <p>You can use this authentication level by setting a license ID list in the local testca_ra/en.license file.</p>

<b>SSL client authentication (authmode=1/2)</b>	<p>This authentication level is valid only after a certificate has been issued. With this authentication level, it is possible to update and revoke a client certificate. In addition, if a certificate is withdrawn, the client cannot use it any longer, so it must be re-issued to use again.</p>
<b>Challenge PIN / License ID authentication (UPKI mode) (authmode=4)</b>	<p>For LicenseID authentication, application for a certification from the user must be accepted face-to-face and the license ID must be directly reported to the user.</p> <p>In this mode, the user can apply for a certificate on the Web (by setting ChallengePIN) and save application information, including the attributes of inetOrgPerson, to the LDAP server. After that, the RA operator enables issuance of the certificate and the user obtains the certificate by inputting ChallengePIN on the Web or command line. LicenseID is dynamically generated or deleted, and is treated as so-called session information.</p> <p>ChallengePIN is also used when a certificate is updated or its validity period expires. The certificate is not updated or its validity period is not made to expire by SSL client authentication used in authmode=1/2.</p> <p>In this mode, the role of RA is defined.</p> <ul style="list-style-type: none"> <li>• RA administrator ... Appointed by the CA operator and checks and authenticates new RA operators.</li> <li>• RA operator ... Appointed by the RA administrator and checks and issues the certificate of a user.</li> </ul> <p>Basically, use the immediate issuance mode in this mode. In addition, be sure to set an LDAP server and a mail server.</p>



- Web Enrollment Setting

To use web enrollment, you must set the CGI script. The `lib/httpd.conf` file contains an example setting for Apache. If you use the enroll CGI module on Apache, refer to this example to rewrite the `httpd.conf` configuration file.

```
Alias /testca_ra/img "/usr/local/naregi-ca/testca_ra/cgi/img"
<Directory "/usr/local/naregi-ca/testca_ra/cgi/img">
    Options Indexes MultiViews
    AllowOverride None
    Order allow,deny
    Allow from all
</Directory>

ScriptAlias /testca_ra "/usr/local/naregi-ca/testca_ra/cgi"
<Directory "/usr/local/naregi-ca/testca_ra/cgi">
    AllowOverride None
    Options None
    SSLOptions +ExportCertData +StdEnvVars
    Order allow,deny
    Allow from all
</Directory>

Alias /aicomponents "/usr/local/naregi-ca/ratemplate/components"
<Directory "/usr/local/naregi-ca/ratemplate/components">
    Options Indexes MultiViews
    AllowOverride None
    Order allow,deny
    Allow from all
</Directory>
```

On Apache 1.3, insert this example content into the `<IfModule mod_alias.c>` section. In the example above, NAREGI CA is installed in `/usr/local/naregi-ca`. In addition, Apache 2 has no `<IfModule mod_alias.c>` section, so define another similar `ScriptAlias` nearby and place the CGI script in it.

The `testca_ra` directory contains an HTML template, an `en.passwd`, and other files for session management (`sessions.0`). Out of all the data files in this directory, only the `testca_ra/cgi/img` and the `testca_ra/cgi` can be accessed. The CGI module is named `aienroll.cgi` and is located in the `testca_ra/cgi/aienroll`. The URL to access the web enroll CGI is `http://localhost/testca_ra/aienroll`. When you access web enrollment a certificate request window or an authentication window appears.

- **xenroll.dll Location**

To use the web enroll CGI, a private key must be generated on the client side using Internet Explorer. Internet Explorer uses a xenroll.dll, which is an ActiveX component, to enroll a certificate.

The xenroll.dll requires the latest version of Internet Explorer. Sometimes the NAREGI CA web enroll CGI does not work because the xenroll.dll version for the Internet Explorer version used by the client does not match the xenroll.dll version for the Windows operating system used by the client. To prevent a version mismatch from occurring, make sure the user can automatically download the latest xenroll.dll from the `ratemplate/components/` directory.

In addition, the latest xenroll.dll is available as a downloadable patch from the following URL.

<http://support.microsoft.com/default.aspx?scid=kb;en;323172>

The xenroll.dll is a product owned and licensed by Microsoft Corporation; nevertheless, place it in the `ratemplate/components/` directory so that users can download it as a patch. Alternatively, you can ask users to download and apply the MS02-04 patch from Microsoft, which will update the xenroll.dll and enable web enrollment.

- **mod\_ssl Setting**

If you are using web enrollment with SSL client authentication, you must configure the `mod_ssl` module. To use the `mod_ssl` module, you must issue an SSL server certificate, map a CA certificate, and configure the `httpd.conf` file. The steps are shown below.

- 1 The first step is to issue an SSL client certificate to use for the web server. This step assumes that the CA has already been built. Follow the instructions for certificate issuance in Chapter 4, CA Operations and generate a `newcert.cer` file and a `newkey.key` file, which you can use for the SSL server certificate.

```
bash$ cd myca
bash$ certreq -size 1024
...(abridged)...
bash$ aica sign newreq.p10
...(abridged)...
```

- 2 The newkey.key file, which is a private key, is encrypted. If you do not want to enter a password when starting an SSL server, use the certconv file, which is in an unencrypted state. (In addition, you can generate a key for a one-time execution using the certreq -noenc option.)

```
bash$ certconv -key newkey.key -pem -noenc newkey.key
Input PASS Phrase: (Enter Password)
++Check One PKCS#12Bag, type=11002
please input password for output key.
output a file ..
```

- 3 Next, place the SSL server certificate, the private key of the SSL server, and the CA certificate into a designated directory. The CA certificate must belong to the CA that issued the SSL server certificate. After the CA certificate is placed in a directory, you can use the make command in the ssl directory to register the CA in the mod\_ssl module.

```
bash$ mv newcert.cer /etc/httpd/conf/ssl/ssl.crt/server.crt
bash$ mv newkey.key /etc/httpd/conf/ssl/ssl.key/server.key
bash$ cp ca.cer /etc/httpd/conf/ssl/ssl.crt/myca.crt
bash$ cd /etc/httpd/conf/ssl/ssl.crt
bash$ make
...(abridged)...
```

- 4 Open the httpd.conf file with a text editor and configure the mod\_ssl settings. Enable the items located in the <IfDefine SSL> section.

```
SSLCertificateFile /etc/httpd/conf/ssl/ssl.crt/server.crt
SSLCertificateKeyFile /etc/httpd/conf/ssl/ssl.key/server.key
SSLCACertificatePath /etc/httpd/conf/ssl/ssl.crt

SSLVerifyClient optional
SSLVerifyDepth 10
```

In addition, apply the “optional” setting in SSLVerifyClient directive. SSL client authentication cannot be used before a client is issued a valid certificate. Accordingly, after a client is issued a certificate, SSL client authentication is ready to be used. With license ID authentication, access through http is possible before

certificate issuance and client authentication through https is possible after certificate issuance.

- 5 After completing the settings, start the httpd daemon.

```
bash$ httpd -DSSL
```

### 3.4. Start and Stop Web Enrollment

- Start Web Enrollment

The web enroll CGI is run from a web server, so it does not have to be started up manually. In addition, you can only start an enrollment check when the mode for validation and issuance by operator is enabled. You do not need to start the immediate issuance mode.

- Stop Web Enrollment

The web enroll CGI is not a resident service, so you do not need to stop it manually. It only starts when the mode for verification and issuance by an operator is enabled; so immediate issuance mode has no stop command for it.

- Execution Access Rights

The web enroll CGI has nobody user rights because it is started from the httpd daemon. Accordingly, the nobody user needs to have read/write rights on the /tmp/aica directory and read rights on the certificate store. In addition, it needs read rights for directories below the “testca\_ra” and read/execute rights for the testca\_ra/cgi/aienroll.

- Log File Output

A web enrollment session outputs a log file as described in “RAAd Setting” of Section 2.5. Normally, an access log (enroll\_access.log), an issuance log (enroll\_issue.log), and an error log (enroll\_error.log) are generated and output to a user-specified directory. The content of each log is described in the table below.

<b>Access log</b>	Outputs all the operation requests that go through a web server and their request results. Output is mainly based on web enroll CGI. Contains the date/time, the CA user, the CA name, and the operation content.
<b>Issuance log</b>	Outputs the records for certificate issuance and revocation based on the enrollment check. Contains the date/time, the CA user, the CA name, and the operation content.
<b>Error log</b>	Outputs the operation error content based on the web enroll CGI or the enrollment check. Contains the date/time, the CA user, the CA name, the operation error, and the error number of the cryptographic library.

In addition, you can rotate the size of a log file, which will add a date/time tag (for example: enroll\_access.log.20031127111008) to the log file name. It is recommended that you back up these log files to a separate medium.

### 3.5. Start and Stop RA Server

- Start RA Server

Once you start the RA server, immediately execute the `airad` command. If the `aica.cnf` file and the RA structure are configured correctly, the RA server start information will appear as shown in the box below.

```
bash$ airad
starting RA server ... read config file.
port=11412,listen=5
ssl=1,req=48,vfy=9
read server certificate : O=test.naregi.jp, OU=server-69dc86-4d8bea,
CN=caserver0100,
set certificate request option (mode=48)
start airad daemon process (23293)
```

After you execute the `airad` command, follow the procedure in “RA Setting” of Section 2.5 to register the RA for remote operations. A password is set when the RA is built, so it is ready for automatic startup. However, if the password is entered incorrectly, certificate issuance will fail when running web enrollment and other processes. Use the `aiconftool` utility to set the password correctly.

SSL is used to connect to an RA server when you request a certificate from the remote `certreq` command. Accordingly, an SSL server certificate, which is automatically generated during the NAREGI CA installation, is required on the RA server side. If the CA server and the RA server are running on the same machine, the two servers can share an SSL server certificate. However, if they are on separate machines, each one will have its own server CA and server certificate. This server certificate is located in the NAREGI CA certificate store and encrypted using its previously set password. In the `aica.cnf` file, the `sv_id` field is configured with the SSL server certificate ID and the `sv_id_pwd` is configured with the password, so when the RA server starts, it is ready to automatically obtain an SSL server certificate. For details on the SSL server certificate, refer to Section 3.7, Automatically Generate SSL Server Certificate.

If the RA registers successfully, the daemon will start. As shown in the box above, the daemon process ID is 23293. The daemon listens for connections on the specified port (11412) and starts a subprocess when a connection is accepted. After the subprocess negotiates an SSL handshake the RA can operate. Only one daemon process can run at one time because the RA server occupies the specified port.

- Stop RA Server

To stop the RA server, use the kill command and the daemon process will immediately stop.

```
bash$ kill 23293
bash$
```

Actually, the only process that stops is the parent process. If a subprocess has executed some other RA operation, the subprocess will not receive the kill command and it will continue processing until completion.

- Execution Access Rights

When the RA server starts it inherits user rights from the airad daemon. If a general user executes the airad daemon, that user must have access to the /tmp/aica/, the certificate store, and the RA directories.

In addition, you can execute airad from the root directory; but from a network security standpoint, that is not wise. A better method is to create users for the RA server (or the CA server) and assign RA building rights and operating rights depending on each user.

- RA User Authentication

The RA server and the web enrollment have common authentication and operations. For details refer to Section 3.3, Web Enrollment Setting.

- Log File Output

The RA server and the web enrollment have a common log output. For details refer to Section 3.4, Web Enrollment Setting.

### 3.6. Start and Stop enrollment check daemon

.....

- Overview of enrollment check command

As a result of the enrollment check (bin/aienroll), POST is checked and an e-mail message notifying of updating is transmitted. The format of the command is as follows.

```
aienroll [Option]
Option:
  -s section      : Set number (typically 0) of aica.cnf [RAd RegInfo]
```

- Starting enrollment check

To start enrollment check, directly execute the aienroll command. If aica.cnf is correctly set and CA is correctly built, the start of enrollment check can be confirmed as follows.

```
bash$ aienroll -s 0
start aienroll daemon process (5496)
```

When the aienroll command is executed, enrollment check is started in accordance with the setting of the “RAd RegInfo X” section of aica.cnf. If a password character string is specified for “ca\_pwd.X=” at this time, enrollment check is automatically started. If no character string is specified, a user is prompted to input the master password of the CA to be accessed or the password of the CA operator private key. Enrollment check is one process for one section, so it can start two or more processes.

- Stopping enrollment check

To stop enrollment check, directly stop the process by using the kill command.

```
bash$ kill 5496
bash$
```

Enrollment check does not generate child processes at all, but two or more enrollment checks can be started. When the process is viewed by the ps command, many aienroll commands can seem to operate. In this case, kill each enrollment check.

- Execution access right

Enrollment check is executed, taking over the user privilege that executed aienroll.



If aienroll is executed by a general user, the user who executes aienroll must be able to access /Install\_DIR/aica/lock, certificate storage, and “testca\_ra”.

It is also possible to execute aienroll by root, but this is not preferable when general network security is taken into consideration. It is recommended to create a user for the CA server and that that user build CA and start enrollment check.

- Outputting log file

The log of enrollment check is output in the same manner as Web Enroll. Refer to 3.3, Setting Web Enroll.

### 3.7. Setting XKMS Service

#### ● Overview of XKMS

XKMS (Xml Key Management Specification) is defined by W3C and specification of Ver2.0 has been publicized as of 2005. It performs various operations related to PKI in a Web service based on XML.

XKMS references XML Signature and XML Encryption defined by W3C and IETF, and defines X-KISS (Xml Key Information Service Specification) that provides functions to search keys and certificates, and X-KRSS (Xml Key Registration Service Specification) that provides a function to register certificates.

X-KISS defines operations related to certificate information and supplies the following two services.

- Locate Service: Certificate searching function
- Validate Service: Certificate verification function

X-KRSS defines operations related to certificate registration and supplies the following four services.

- Register Service: Certificate issuance function
- Reissue Service: (Same key) certificate updating function
- Revoke Service: Certificate revocation function
- Recover Service: User key recovery function

The NAREGI XKMS service and XKMS Java API supply a Register service and a Revoke service, and do not supply functions either to update a certificate with the same key or to recover a user key.

The supported authentication levels are as follows.

<b>Anonymous authentication</b>	Issues a certificate without authenticating a user. A certificate is immediately issued at this authentication level. When a revocation operation is performed, a certificate is immediately revoked as soon as it has been specified.
<b>License ID authentication (one-time license system)</b>	<p>To issue a certificate, the user must input a license ID. One license ID can be used for issuing only one certificate. A revocation operation cannot be performed at this authentication level.</p> <p>To set this authentication level, set a list of license IDs to a local file such as testca_ra/en.license.</p>

- **Operating environment**

The XKMS service was developed to operate with Java 1.4 or later. Several PKI-related classes that are used by the service, i.e., classes such as X509Certificate and X500Principal could not be used with Java 1.3 or earlier. Java 1.4 or later is therefore essential.

Apache Tomcat+Axis is used as the engine of a Web service. The XKMS service can be used by correctly locating the Axis and XKMS class libraries.

<b>Supported machines</b>	Machines on which JavaVM operates
<b>CPU</b>	CPU supported by JavaVM
<b>Supported OS</b>	Any
<b>Memory</b>	32 MB or more (recommended)
<b>Java version</b>	1.4 or later
<b>Tomcat version</b>	5.0, 5.5
<b>Axis version</b>	1.2
<b>Other necessary Java packages</b>	Java LCMPI API 1.1, Apache xml-security-1.2.1, JavaMail API 1.3.3

In addition, the following Java packages are necessary for operating the XKMS service.

- Java LCMPI API 1.1  
For Java LCMPI API, use jlcmp.jar included in the package of NAREGI CA.
- Apache XML Security 1.2.1  
<http://xml.apache.org/security/>  
Download the latest version of Apache XML Security from the above site.
- JavaMail API 1.3.3  
<http://java.sun.com/products/javamail/>  
The XKMS service transmits an e-mail to notify issuance of a certificate to a CA operator. The JavaMail API is used to transmit the e-mail,.

- **Setting up Apache Axis 1.2**

Here is an example for setting up Apache Axis 1.2 in a Windows environment.

- 1 Install JDK 1.4. Change the environment variables of the OS.  
e.g., set JAVA\_HOME= d:\j2sdk1.4.1\_02
- 2 Install Apache Tomcat 5.5. After installation, start Apache Tomcat from “Service” on the control panel. After starting, access <http://localhost:8080/> to check the operation.
- 3 Install Apache Axis 1.2. Change the environment variables of the OS.

```
set AXIS_HOME=d:\axis-1_2
set AXIS_LIB=%AXIS_HOME%\lib
set
AXISCLASSPATH=%AXIS_LIB%\axis.jar;%AXIS_LIB%\commons-discovery.jar; %AXIS_LIB%\commons-logging.jar;%AXIS_LIB%\jaxrpc.jar;%AXIS_LIB%\saaj.jar;%AXIS_LIB%\log4j-1.2.8.jar;%AXIS_LIB%\xml-apic.jar;%AXIS_LIB%\xercesImpl.jar
set CLASSPATH=.;%AXISCLASSPATH%;%CLASSPATH%
```

- 4 Copy axis-1\_2\webapps\axis to Tomcat5.5\webapps\axis.
- 5 Use “Service” to restart Apache Tomcat.  
After restarting, access <http://localhost:8080/axis/happyaxis.jsp> and verify the operation to see if the Web service is correctly set up. Components can be checked by <http://localhost:8080/axis/happyaxis.jsp>. In this status, the optional components are not included.

- **Setting up optional components for Axis**

Set up Apache XML Security and JavaMail library as optional components.

- 1 Download Apache XML Security 1.2 from the above site. After downloading, decompress the file, and copy to the directory of Axis the Jar file included in libs. Copy destination: Tomcat5.5\webapps\axis\WEB-INF\lib\\*.jar
- 2 Download JavaMail API 1.3.3 from the above site. After downloading, decompress the file and copy the mail.jar file to the directory of Axis.  
Copy destination: Tomcat5.5\webapps\axis\WEB-INF\lib\mail.jar

- 3 Use "Service" to restart Apache Tomcat.

After restarting, access <http://localhost:8080/axis/> and verify the operation to see if the Web service is correctly set up. Components can be checked by <http://localhost:8080/axis/happyaxis.jsp>. Check if the optional components are also recognized.

- **Setting up XKMS service**

Copy the files necessary for the XKMS service from the NAREGI CA directory and deploy the service.

- 1 Copy the Jar file of Java LCMP API from `naregi-ca\jlcmp\lib\jlcmp.jar` to the directory of Axis.

Copy destination: `Tomcat5.5\webapps\axis\WEB-INF\lib\jlcmp.jar`

- 2 Copy the Jar file of the XKMS service from `naregi-ca\xkms\lib\xkms-naregi.jar` to the directory of Axis.

Copy destination: `Tomcat5.5\webapps\axis\WEB-INF\lib\xkms-naregi.jar`

- 3 Deploy the service. Confirm that the Apache Tomcat service is started, move to the `naregi-ca\xkms` directory, and execute the following command.

```
Naregi> java -cp %AXISCLASSPATH% org.apache.axis.client.AdminClient
deploy.wsdd
log4j:WARN No appenders could be found for logger
(org.apache.axis.i18n.ProjectResourceBundle).
log4j:WARN Please initialize the log4j system properly.
File deploy.wsdd is being processed. / [en]-(Processing file deploy.wsdd)
<Admin> processing has been executed. / [en]-(Done processing)</Admin>
```

- 4 The deploy processing will be executed. Access <http://localhost:8080/axis/> and check the list of the Web service. Use <http://localhost:8080/axis/servlet/AxisServlet> to display the list and confirm that `XKMSService` is registered.

- **Creating environment to use XKMS**

To actually use the XKMS service, it is necessary, after setting up RA, to locate an operator certificate and a CA certificate storage file, locate a license ID file, and correct and locate the xkms-naregi.properties file.

Set each file as follows.

- 1 An operator certificate is necessary for connecting to the CA server. Output the operator certificate as a file and locate it in the directory of RA (caname\_ra in this example).

```
Naregi> aistore -id CAOperator001 -ef pk12 -e caop.p12  
Access Private Key: (Enter Password)  
Input Export Password: (Enter Password)  
Verifying - Input Export Password: (Enter Password)  
export a store data successfully.
```

After that, locate the caop.p12 file caname\_ra\caop.p12.

- 2 Create a trustable CA store for the XKMS Validation service. Move to the RA directory and locate the ca.store file as shown below.

```
Naregi> cd caname_ra  
Naregi> keytool -import -file ca.cer -alias cacert -trustcacerts -keystore ca.store  
Input the password for key store: abcdef  
Owner: OU=testca unit, O=testca, C=JP  
Executioner: OU=testca unit, O=testca, C=JP  
Serial number: 1  
Valid date: Mon Sep 12 05:01:24 GMT 2005 Valid period: Thu Sep 11 05:01:24 GMT 2008  
Fingerprint of certificate:  
MD5: 10:C8:EC:91:EE:CD:07:E2:E4:35:90:13:BA:04:57:FA  
SHA1: DD:18:98:30:1D:95:FA:6F:9A:50:64:35:98:01:04:33:35:51:0B:9C  
Do you trust this certificate? [no]: yes  
The certificate has been added to key store.
```

- 3 Open the naregi-ca\xkms\xkms-naregi.properties file and customize it in accordance with the environment.

```
### CA server information ###  
org.naregi.xkms.caServer=localhost  
org.naregi.xkms.caPort=11411  
org.naregi.xkms.caName=caname  
org.naregi.xkms.certProfile=SMIME user  
  
### CA Operator certificate ###  
org.naregi.xkms.opCertP12=d:/naregi-ca/caname_ra/caop.p12  
org.naregi.xkms.opCertP12Pwd=abcde
```

```

### trust CA store for XKMS validation ###
org.naregi.xkms.trustCAStore=d:/naregi-ca/caname_ra/ca.store
org.naregi.xkms.trustCAStorePwd=abcdef

### XKMS authentication mode ###
# 0 .. anonymous
# others .. license ID
org.naregi.xkms.authmode=0

# license ID file (Choice)
org.naregi.xkms.license=d:/naregi-ca/caname_ra/en.license

# license ID on LDAP server (Choice)
#org.naregi.xkms.ldapServer=
org.naregi.xkms.ldapBind=DIGEST-MD5
org.naregi.xkms.ldapBase=c=JP

org.naregi.xkms.ldapAdmin=cn=ldapAdministrator,c=JP
org.naregi.xkms.ldapAdminPwd=
org.naregi.xkms.ldapLicenseAttr=uid

### smtp setting (Optional) ###
org.naregi.xkms.smtpServer=
org.naregi.xkms.systemEmail=
org.naregi.xkms.adminEmail=
org.naregi.xkms.acceptCsr2=d:/naregi-ca/templates/enroll_accept_csr2j.txt

# Japanese mail text
org.naregi.xkms.acceptCsr2sbj=
=?iso-2022-jp?B?GyRCRUU7Uj5aTEA9cSROSC85VBsoQg==?=
org.naregi.xkms.emailCharset=iso-2022-jp

```

Each field is described in the table below (only part of org.naregi.xkms.\*).

<b>caServer</b>	Specifies the server name of the remote CA that the XKMS service accesses.
<b>caPort</b>	Port number of CA server. Typically, 11411 is used.
<b>caName</b>	Specifies the CA name of the remote CA that the XKMS service accesses.
<b>certProfile</b>	Specifies a profile name to be used to issue a certificate.
<b>opCertP12</b>	Specifies the PKCS#12 file name of the operator certificate. To connect to the CA server, SSL client authentication is necessary.
<b>opCertP12Pwd</b>	Specifies a password to access the PKCS#12 file of the operator certificate.
<b>trustCAStore</b>	Specifies a CA certificate store file trusted by the XKMS Validate service.
<b>trustCAStorePwd</b>	Specifies the access password of a CA certificate store file trusted by the XKMS Validate service.
<b>authmode</b>	Specifies whether the user applying for a certificate is to be authenticated. An integer value is specified and the following

	<p>modes are available.</p> <p>0: Enables issuance of a certificate to an anonymous user.</p> <p>Other: Authenticates the user by a license ID (one-time license).</p> <p>The certificate can be revoked only in the anonymous authentication mode.</p>
<b>license</b>	Specifies a license file if license ID authentication is necessary for applying for a certificate. This value is ignored in the Anonymous mode.
<b>ldapServer</b>	Specifies an LDAP server when license ID authentication is necessary for applying for a certificate. This value is ignored in the Anonymous mode.
<b>ldapBase</b>	DN (Distinguished Name) indicating the search base entry of the LDAP server
<b>ldapAdmin</b>	Specifies a user having a write privilege to delete the license ID upon entry and output a certificate when the certificate is issued.
<b>ldapAdminPwd</b>	Specifies the password of a user having a privilege to write an entry.
<b>ldapLicenseAttr</b>	Specifies an attribute to be used to search for a License ID.
<b>smtpServer</b>	Specifies an SMTP server to notify issuance information.
<b>systemEmail</b>	E-mail address of the XKMS system. It must be an e-mail address by which the SMTP server can transmit an e-mail.
<b>adminEmail</b>	Specifies the e-mail address of the CA operator. When application for issuance of a certificate is accepted, a notice of this fact is sent to this e-mail address.
<b>acceptCsr2</b>	Specifies a file that holds the text of the e-mail notification.
<b>acceptCsr2subj</b>	Specifies the subject of the e-mail notification.
<b>emailCharset</b>	Specifies the character set of the e-mail notification.

- 4 After editing the `naregi-ca%$xkms%$xkms-naregi.properties` file, copy it to the `conf` directory of Tomcat.  
Copy destination: `Tomcat5.5%$conf%$xkms-naregi.properties`
- 5 After completing the setting, compile `XKMSRegisterSample.java` and access the XKMS service. If the certificate can be issued, the setting is completed.



### 3.8. Starting and Stopping XKMS Service

.....

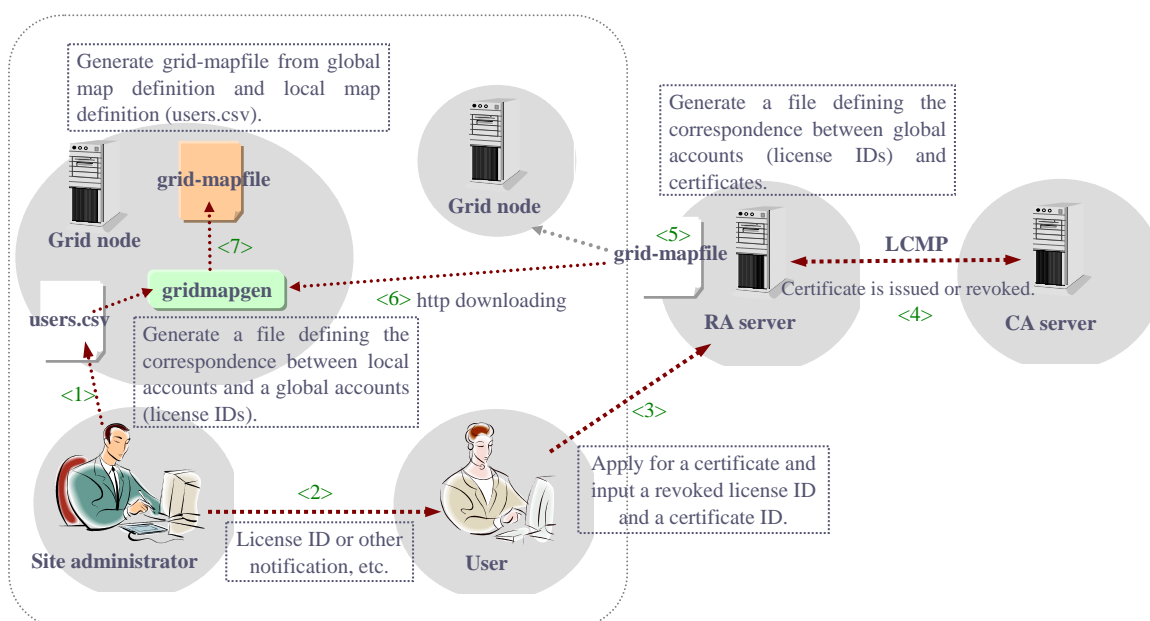
- Starting XKMS service  
To start the XKMS service, start Apache Tomcat. Start Tomcat in accordance with each version and the starting procedure of each OS.
- Stopping XKMS service  
To stop the XKMS service, stop Apache Tomcat. Stop Tomcat in accordance with each version and the starting procedure of each OS.
- Execution access right  
The XKMS server is executed taking over the user privilege of Tomcat. Therefore, the Tomcat user must have the privilege to read “caname\_ra” or below and the privilege to read or write caname\_ra/en.license.  
It is preferable that files holding important information, such as the CA operator certificate, trustable CA store file, and xkms-naregi.properties file hold a privilege to be read by the user who has executed Tomcat.
- User authentication of XKMS service  
For user authentication of the XKMS service, refer to Overview of XKMS above.
- Outputting log file  
The log of XKMS is output to logs¥stdout\_\*.log of Tomcat. If an exception or an operation error occurs, reference this log.  
File name: Tomcat5.5¥logs¥stdout\_\*.log

### 3.9. Setting Grid Linking Function

- Overview of grid linking functions

NAREGI CA has a function to link with Grid middleware Globus or UNICORE, and the user certificate and a local account can be efficiently mapped by using the gridmapgen process.

Specifically, grid-mapfile (RA server) should be output to a specific directory (gridmap of RAd RegInfo should be enabled) when an RA site is built. This grid-mapfile (RA server) is created by the aienroll process. The output file is publicized by httpd and collected by the gridmapgen process operating on the grid node of each site. After obtaining global grid-mapfile (RA server), read the users.csv file located in the local machine, map the subject of the certificate and local account, and, in the end, locate grid-mapfile (local) used by Globus to the local machine.



To use UNICORE, output a certificate file (PEM) to the RA server (enable gridcertpath of RAd RegInfo). If a certificate is immediately issued, the certificate file (PEM) is output by airad or aienroll.cgi. If it is checked and issued by the administrator, the file is output by aienroll. If a local user is added or deleted, the gridmapgen process adds or deletes of a user certificate by using UADB/bin/add or delete command.

ID/password authentication or License ID authentication is always necessary to link a grid map file.

\* Restriction

Because of the specification of the UUDB/bin/delete command, two or more subjects are listed for selection when a subject is to be deleted, if two or more certificates are made to correspond to one local account. A selection operation like this cannot be performed when this command is called from the gridmapgen process,, so one certificate must correspond to one local account.

### 3.10. Starting and Stopping Grid Linking Function

.....

- Starting RA server

The grid linking function can be used on the assumption that all CA, RA, and Web servers have been set and correctly operate. If so, start aicad, and then airad and aienroll. Also start httpd to publicize the grid-mapfile and the certificate file (PEM) that has been issued on a Web site.

- Stopping RA server

Manually stop the RA and Web servers to stop the grid linking function. Stop the process by using the kill command.

- RA server execution access right

The RA server is executed, taking over the user privilege by which airad and aienroll have been executed. If airad and aienroll have been executed by a general user, the user who has executed airad and aienroll must be able to access the certificate store and RA directory.

The /RA-directory/grid/certs/ directory is written by both enroll CGI and airad. The groups of the account that starts httpd and the account that starts airad must be unified, so that they can be accessed by both CGI and airad.

Although it is possible to execute airad by root, this is not preferable in view of general network security. It is recommended to create a user for the RA server (or for the same user of the CA server) and that that user build RA and start the RA server.

- Starting gridmapgen

The gridmapgen command is started by each grid node. To start a process, confirm that the RA and Web server have been started. If grid-mapfile cannot be obtained from the specified URL, an operation error occurs and an error log is output.

```
bash$ gridmapgen  
start gridmapgen daemon process (20229)
```

When the gridmapgen command is executed, a grid map linking module is started in accordance with the setting of the Grid MapGen section of gridmap.cnf. For Globus, grid-mapfile is updated if the usermap that defines a local user, the output destination of grid-mapfile, and the reference destination URL (refmap) are set. For UNICORE, UADB is updated if a UADB command path and a reference destination URL (refcert) are set in addition to the above.

- Stopping gridmapgen

To stop gridmapgen, directly stop the process by using the kill command.

```
bash$ kill 20229  
bash$
```

- gridmapgen execution access right

Gridmapgen updates grid-mapfile or UADB, and must be executed by a user having a privilege to write these files.

Although it is possible to execute gridmapgen by root, this is not preferable when general network security is taken into consideration. It is recommended to create a user for updating grid-mapfile and that that user start a process.

- Outputting gridmapgen log file

gridmapgen outputs a log file set to the Grid MapGen section of gridmap.cnf. Two types of logs, an access log and an error log, are usually output to a specified directory under the names of map\_access.log and map\_error.log, respectively. The output contents of each log are as follows.

<b>Access log</b>	Outputs the operating status of gridmapgen to a log. Date, time, and operation are described in the log.
<b>Error log</b>	Operation error of gridmapgen is output to a log. Date, time, and the nature of the error are described in the log.

The log file can be rotated in file size. When it is rotated, the date of executing rotation is added like map\_access.log.20031127111008. It is recommended to save these files to a medium, including a tape .

### 3.11. Remote Access Setting

SSL client authentication is used to access a CA server. Accordingly, to negotiate an SSL session, the RA server side and the CA operator side require a CA certificate for the SSL server, a CA operator certificate, and a private key. This section describes the steps to obtain these files and where to place them.

- 1 An SSL server certificate is automatically generated during the NAREGI CA installation. For details about automatic generation, refer to Section 3.7, Automatically Generate SSL Server Certificate. A CA certificate for an SSL server certificate is registered in the “root” certificate store.

```
bash$ aistore -st root
[unique-id]          subject          serialNumber
-----
[business unit] C=JP, O=my hoge2, OU=business unit, 1
[server-a9bec5-d95c8d] O=hostname, OU=server-a9bec5-d95c8d, 1
```

For the example shown in the box above, the ID of the CA certificate used for the SSL server is “server-a9bec5-d95c8d.” The O=“hostname” and the OU=“server-\*\*\*\*\*-\*\*\*\*\*” subject matter indicate the CA certificate.

- 2 Specify the certificate ID then extract the CA certificate.

```
bash$ aistore -st root -id server-a9bec5-d95c8d -e sslca.cer
export a store data successfully.
```

- 3 Similarly, the CA operator certificate is registered in the “my” certificate store. Check the certificate ID then extract the certificate as a PKCS#12 formatted file.

```
bash$ aistore -id CAOperator001 -ef pk12 -e caop.p12
Access Private Key: (Enter Password)
Input Export Password: (Enter Password)
Verifying - Input Export Password: (Enter Password)
export a store data successfully.
```

- 4 Transfer the extracted caop.p12 and the sslca.cer files to a separate management machine so they can be used for CA remote access.

- 5 Install each file in the certificate store. Each file is automatically placed in the my store and in the root store. The certificate ID is automatically added and, optionally, you can have the ID explicitly shown.

```
myhost % aistore -i sslca.cer  
import a file to the store successfully.  
myhost % aistore -i caop.p12  
Input PKCS#12 Password: (Enter Password)  
Save Access Password : (Enter Password)  
Verifying - Save Access Password : (Enter Password)  
import a file to the store successfully.
```

- 6 Access the CA server and check to see if the files have been placed correctly in the certificate stores.

```
myhost % aica print -sv caserv:testca -ssl -clid CAOperator001  
tring to connect caserv(11411):testca (ssl)  
Open Private Key: (Enter Password)  
-----  
Certificate Profile : SMIME user  
certificate version : 3  
current serial number: 1  
signature algorithm : md5WithRSAEncryption  
...(abridged)...
```

### 3.12. Automatically Generate SSL Server Certificate

During the NAREGI CA installation, a script for the `initsslcert.sh` shell is executed. This shell script has commands that are used to build an SSL Server CA, which is used by a CA server, and that are used to issue an SSL server certificate.

The SSL server CA is named `ssl-serv` and placed in the NAREGI CA installation directory where it can be operated using `aica` commands. The CA password is “1234567890sslca” and using an SSL server profile creates the server certificate. The CA certificate and the SSL server are both valid for 10 years starting from the time of installation.

A certificate that is automatically generated has a limited period of validity, so you will need to update and reissue the SSL server certificate when it expires. A CA server can use not only a CA that was automatically generated but also an SSL server certificate that was generated by a separate CA. Accordingly, when the time approaches to update a certificate you should build an SSL CA and issue an SSL server certificate.

### 3.13. Back Up CA and RA Data

This section describes the important NAREGI CA data that you should back up. Basically, if you back up the CA directories, the RA directories, and the NAREGI CA installation directory, you can use these backups to restore data to the point at which the back up was taken. For details on each data item, refer to below.

- CA Data

With NAREGI CA you can build a number of CAs on one machine. Each CA that you build has a CA directory that contains a number of data items, which you can control by using `aica` commands. Each CA directory contains its own CA private key, profile management information, user certificates, and user private keys. The directory file structure is shown below.

CA\_Name/

CA\_Name/ca.cer: CA certificate

CA\_Name/ca.p12: CA certificate and private key

CA\_Name/ca.cai: CA management information (e.g., profile list)

CA\_Name/ca.passwd: CA operator information (e.g., authentication, access rights)

CA\_Name/ca.group: CA operator group information

CA\_Name/cert/ProfileName.cpi: Certificate profile information

CA\_Name/cert/ProfileName.cts: User certificate

CA\_Name/cert/ProfileName.kys: User private key

CA\_Name/cert/ProfileName.lpi: CRL profile information

CA\_Name/req/csr.rpi: CSR queue information

The CA backup can save the directories shown above. The restore process expands the backup data and returns the CA information to its former state.

- RA Data

With NAREGI CA you can build a number of RAs on one machine. Each RA that you build has an RA directory that contains structured data, which you can control by using the enrollment CGI script, the airad daemon, and the aienroll server. Each RA directory contains a user session file (sessions.0), a password file (en.passwd), and a license file (en.license). When you use enrollment, you must backup each RA directory.

- Certificate Store

The certificate store, a frequently accessed database, is used to obtain SSL server certificates and to authenticate SSL client certificates. Normally, this data is placed in the NAREGI CA installation directory/store folder.

- aica.cnf

The aica.cnf file is the configuration file for the CA server, the RA server, and the CRL publisher. It is a frequently accessed file that you will need to reference for nearly all of the NAREGI CA commands. Normally, this file is named aica.cnf and placed in the NAREGI CA installation directory/lib/ folder.

- Log Files

These are the log files for the CA server, the RA server, and the CRL publisher. If necessary, you can save a log file, which is normally placed in the NAREGI CA installation directory/logs/ folder.



## 4. CA Operations

This chapter describes the NAREGI CA command chain used for CA operations.

### 4.1. Create Certificate Request

Use the following procedure to issue a certificate.

With the `certreq` command, generate a key pair and create a certificate request (CSR).

With the `aica` command, sign the certificate request and issue the certificate.

This section describes how to use the `certreq` command.

- 1 Use the `certreq` command when you want to generate a new key pair and create a new certificate request.

The box below shows the various commands.

#### **certreq** [Option]

Option:

- req **file** : Stores a certificate request using the name "file"
- key **file** : Stores a private key using the name "file"
- algo **alg** : Indicates the algorithm of the generated key pair  
rsa, dsa, and ecdsa are available (default: rsa)
- size **num** : Specifies the key-pair length (default: 512 bits)
- p **passwd** : Specifies the password used for private key storage
- noenc : Stores a private key without encrypting it
- der : Specifies DER as the storage format for the  
certificate request
- alt : Specifies Subject Alt Name

Each algorithm is described in the table below.


<b>RSA encryption</b>	A public key encryption algorithm that is used as the default method. It can be used for encrypting and digital signing. In general, a secure certificate can be issued by specifying a key length of 1024 bits.
<b>DSA encryption</b>	A public key encryption algorithm that is only used for digital signatures. In general, a secure certificate can be issued by specifying a length of 1024 bits.
<b>ECDSA encryption</b>	A public key encryption algorithm that is only used for digital signatures. ECDSA, which stands for elliptic curve digital algorithm, provides a high level of security. In general, a secure certificate can be issued by specifying a length of 192 bits.

<b>MD5 (Hash function)</b>	Known as a one-way hash function, it can obtain a digest of data (index).
<b>SHA1 (Hash function)</b>	A hash function that is used as the default method. Known as a one-way hash function, it can obtain a digest of data (index).

- 2 To generate a 512-bit key pair and create a certificate request, execute the commands shown in the box below.


```
bash$ certreq
generate private key (size 512 bit)
00000
...00000
input subject directory.
select directory tag (input number)
(1.C, 2.ST, 3.L, 4.O, 5.OU, 6.CN, 7.Email, 8.Quit)[1]:
Country [JP]: JP
: (abridged)
Input PASS Phrase: (Enter Password)
Verifying - Input PASS Phrase: (Enter Password)
```

As shown in the box above, you generate a key and enter the certificate subject and the private key password. This operation creates a newreq.p10 file and a newkey.key file. The file format is X.509 DER and complies with the PKCS#10 standard.

 For details on subject, refer to Section 2.1, New CA.

- 3 If you change the key length and specify an optional output file name, execute the command shown below.

```
bash$ certreq -size 1024 -req a.p10 -key a.key
```

 If you specified DSA or ECDSA for the key pair algorithm, you must generate the parameters required for key generation. In particular, be aware that ECDSA parameter generation takes a lot of time.

## 4.2. Issue Certificate for Request

Use the following procedure to issue a certificate in response to a certificate request.

- 1 Use the aica sign operation to issue a certificate in response to a certificate request. The aica command line has the format aica [operation] [option], which you can use to issue a certificate or a CRL by specifying each operation. In addition, the aica sign operation has the format shown in the box below.

### **aica sign [Option] file**

Option:

- sn **num** : Specifies the serial number
- pf **name** : Specifies the certificate profile (Default: SMIME user)
- o **file** : Stores a certificate file using the name "file"

General option:

- sv **path** : Specifies the "Server name: CA name"
- ssl : Specifies SSL use in a remote CA connection
- clid **name** : Specifies the ID for the SSL client certificate
- u **loginid** : Specifies the CA user name (non-SSL connection)
- p **passwd** : Specifies the CA user password (non-SSL connection)

- 2 A certificate is issued as shown in the box below.

```
bash$aica sign -o a.cer newreq.p10
CA PKCS#12 file open
Input PKCS#12 Password : (Enter Password)
```

Certificate Request signature ok.

Certificate DATA:

serial number : 1

issuer :

C=JP, O=nec corporation, OU=developer unit,

subject:

C=JP, ST=tokyo, O=test, CN=user01,

notBefore: Jun 07 18:24:15 2002

notAfter : Jun 07 18:24:15 2003

do you sign here ? (y/n)[y]: **y**

now signing ..

output certificate .. done.

When you execute the aica command, the first file you will open is the ca.p12 (CA certificate and private key). If you enter an incorrect password here, the operation will suspend and the process will terminate. If you enter the password correctly and the signature for the imported certificate request is correct, the certificate issuance information and a sign here prompt appears.

At this point enter “y” and press the return key to issue the certificate. In a default situation, a file named newcet.cert is output, but you can use the -o option to specify a name for the name. In addition, the input certificate request (PKCS#10) file can be in PEM or DER format.

- 3** You can also specify the serial number for the issued certificate. In such case, use the command shown below.

```
aica sign -sn 4023 -o out.cer newreq.p10
```

This example adds the serial number 4023 to the issued certificate. If this number is already in use, an error will occur after which the program will terminate.

### 4.3. Register Request then Verify and Issue Certificate

A certificate request (CSR) is stored in the CSR queue of a CA from where it can be issued as a certificate. In addition to immediate issuance, it is possible to first have a CA operator verify a certificate request once before issuing it as a certificate.

- 1 The `aica csr` operation is used to register a certificate request and then either issue a certificate or reject the request. The box below shows the various commands.

#### **aica csr [Option]**

Option:

- post **file** : Registers the CSR file in the queue
- sn **num** : Specifies the serial number
- issue **num** : Specifies the CSR AcceptID
- reject **num** : Specifies the CSR AcceptID
- pf **name** : Specifies the certificate profile (Default: SMIME user)
- o **file** : Stores a certificate using the name "file"

- 2 Use the `csr -post` operation to register a certificate request in the CSR queue. If the registration was successful an acceptID will appear.

```
bash$aica csr -post newreq.p10
CA PKCS#12 file open
Input PKCS#12 Password : (Enter Password)
success to post a CSR (acceptID=2).
```

- 3 Use the `csr -issue` operation to issue a certificate for a certificate request registered in the CSR queue.

```
bash$aica csr -issue 2
CA PKCS#12 file open
Input PKCS#12 Password : (Enter Password)
Certificate DATA:
  serial number : 10
  issuer :
    C=JP, O=nec corporation, OU=developer unit,
  subject:
    C=JP, ST=tokyo, O=test, CN=user10,
  notBefore: Jun 07 18:24:15 2002
  notAfter : Jun 07 18:24:15 2003

do you sign here ? (y/n)[y]: y
now signing ..
output certificate .. done.
```

#### 4.4. Update Certificate

It is possible to change the period of validity and the extension information for a certificate while leaving the serial number and the public key as they are.

- 1 Use the aica re-sign operation to update a certificate. The box below shows the various commands.

##### **aica resign [Option] file**

Option:

-o **file** : Stores a certificate using the name "file"

General option:

-sv **path** : Specifies the "Server name: CA name"

-ssl : Specifies SSL use in a remote CA connection

-clid **name** : Specifies the ID for the SSL client certificate

-u **loginid** : Specifies the CA user name (non-SSL connection)

-p **passwd** : Specifies the CA user password (non-SSL connection)

- 2 After running the re-sign operation, specify the certificate to re-sign. If the certificate has been revoked, it cannot be re-signed.

```
bash$ aica resign -o out.cer in.cer  
CA PKCS#12 file open  
Input PKCS#12 Password: (Enter Password)
```

Certificate Request signature ok.

Certificate DATA:

serial number : 1

issuer :

C=JP, O=nec corporation, OU=developer unit,

subject:

C=JP, ST=tokyo, O=test, CN=user01,

notBefore: Jun 07 18:24:15 2002

notAfter : Jun 07 18:24:15 2003

do you sign here ? (y/n)[y]: **y**

now signing ..

output certificate .. done.

## 4.5. Batch Issue Certificates

It is possible to issue certificates by batch processing using a specified CSV file format.

- 1 Use the `aica csv` operation to issue certificates by batch processing. The box below shows the various commands.

### **aica csv file**

General option:

- sv **path** : Specifies the "Server name: CA name"
- ssl : Specifies SSL use in a remote CA connection
- clid **name** : Specifies the ID for the SSL client certificate
- u **loginid** : Specifies the CA user name (non-SSL connection)
- p **passwd** : Specifies the CA user password (non-SSL connection)

- 2 The box below shows how certificates are batch processed.

```
bash$ aica csv sample.csv
CA PKCS#12 file open
Input PKCS#12 Password: (Enter Password)

generating a certificate : no.0
generate private key (size 512 bit)
.....00000
00000
: (abridged)
CSV operation is finished successfully.
```

The example above shows how certificates are batch processed using a specified CSV file. In addition, during the certificate creation process generates several files that are left behind in the `./cert`, the `./req`, the `./key`, and the `./p12` directories.

### **Specifying a CSV format**

The CSV file format has the following specifications.

#### **Format (One line):**

"Profile name",Serial number,"Subject","SubjectAltName",  
Encryption, Key size, P12 generation flag,"p12 and key password"

#### **CSV example:**

```
"SMIME user",10010,"C=JP/OU=sample  
ou/CN=name10","email:name10@localhost",rsa,1024,1,"abcde"
```

**Important item:**

- Do not enter a space after the comma that separates each item.
- Use double quotation marks around the profile name, the subject and the password.
- Make sure to use capital letters for C, O, and OU.
- Separate the subject tag after the leading part using "/" and do not use spaces.
- Do not use a space after the "=" sign in the subject.
- For the encryption enter rsa, dsa, or ecDSA.
- The key size for an RSA algorithm can be from 512 to 2048 bits long.
- The P12 flag is set to 1 to create a PKCS#12 file and set to 0 to not create.
- Make sure to use the same password for the private key file and the p12 file.

In addition, if the serial number or the subject is already in use, after an error appears move on to the next process.



## 4.6. Revoke Certificate

If the private key for a certificate issued to a user is lost or leaked, it must be revoked.

- 1 Use the aica revoke operation to revoke a certificate. The box below shows the various commands.

### **aica revoke num**

General option:

- sv **path** : Specifies the "Server name: CA name"
- ssl : Specifies SSL use in a remote CA connection
- clid **name** : Specifies the ID for the SSL client certificate
- u **loginid** : Specifies the CA user name (non-SSL connection)
- p **passwd** : Specifies the CA user password (non-SSL connection)

- 2 The box below shows how to revoke a certificate.

```
bash$ aica revoke 3
CA PKCS#12 file open
Input PKCS#12 Password: (Enter Password)
-----
Certificate DATA:
  serial number : 3
  subject:
    C=JP, O=testca, OU=myunit, CN=user00, /Email=myname@local
do you revoke this certificate ? (y/n)[y]: y
-----
Set revocation reason >>
  unspecified(0), keyCompromise(1), cACompromise(2),
  affiliationChanged(3), superseded(4), cessationOfOperation(5),
  certificateHold(6), removeFromCRL(8), privilegeWithdrawn(9),
  aaCompromise(10)
select reason code (-1 means 'cancel') [0]: (Enter Relevant Number)
success to revoke a certificate (sn:3)
```

To revoke a certificate, specify its serial number. If the specified serial number is present, append a revocation mark to the certificate status. If the specified serial number is not present, ignore the previous operation.

## 4.7. Unrevoke Certificate

.....

It is possible to cancel the revocation of a certificate and return it to a normal (unrevoked) status.

- 1 Use the aica unrevoke operation to cancel certificate revocation. The box below shows the various commands.

### **aica unrevoke num**

General option:

- sv **path** : Specifies the "Server name: CA name"
- ssl : Specifies SSL use in a remote CA connection
- clid **name** : Specifies the ID for the SSL client certificate
- u **loginid** : Specifies the CA user name (non-SSL connection)
- p **passwd** : Specifies the CA user password (non-SSL connection)

- 2 The box below shows how a to unrevoke a certificate.

```
bash$ aica unrevoke 3
CA PKCS#12 file open
Input PKCS#12 Password: (Enter Password)
success to unrevoke a certificate (sn:3)
```

To unrevoke a certificate, specify its serial number. If the specified serial number is present and the certificate is revoked, remove the revocation mark from the certificate status. If the specified serial number is not present, ignore the previous operation.

## 4.8. Issue CRL

A certificate revocation list (CRL) is issued to check whether a user certificate or a user signature has been cancelled.

- 1 Use the `aica crl` operation to issue a CRL. The box below shows the various commands.

```
aica crl
General option:
-sv path      : Specifies the "Server name: CA name"
-ssl           : Specifies SSL use in a remote CA connection
-clid name    : Specifies the ID for the SSL client certificate
-u loginid    : Specifies the CA user name (non-SSL connection)
-p passwd    : Specifies the CA user password (non-SSL connection)
```

- 2 The box below shows how to issue a CRL.

```
bash$ aica crl
CA PKCS#12 file open
Input PKCS#12 Password: (Enter Password)

output ARL .. done.
output CRL .. done.
output CRL-All .. done.
```

When you issue a CRL, three CRL files are created in the current directory. Each file is saved in PEM format and they differ as described below.

- `out-CRL.crl`  
Contains the revocation information for a user certificate (serial number and revocation date).
- `out-ARL.crl`  
Contains the revocation information for a subordinate CA certificate (serial number and revocation date). This type of CRL is called an authority revocation list (ARL).
- `out-CRL-All.crl`  
Contains the revocation information for both a CRL and an ARL (serial number and revocation date). Normally, use this file when you publish a CRL on the web.

## 4.9. Export Certificate and Private Key

A CA stores certificates and private keys. You can export a user certificate or a user private from the store by specifying a serial number. In addition, you can also export a certificate request that is stored in the CSR queue of a CA.

- 1 Use the `aica export` operation to export a certificate or a private key. The box below shows the various commands.

### **aica export [Option]**

Option:

- sn **num** : Specifies the serial number
- o **file** : Stores a certificate using the "file" name
- cert : Exports a user certificate (Default)
- key : Exports a user private key
- p12 : Exports a PKCS#12 (private key, certificate) formatted file
- csr **num** : Specifies the AcceptID for the CSR that is exported

General option:

- sv **path** : Specifies the "Server name: CA name"
- ssl : Specifies SSL use in a remote CA connection
- clid **name** : Specifies the ID for the SSL client certificate
- u **loginid** : Specifies the CA user name (non-SSL connection)
- p **passwd** : Specifies the CA user password (non-SSL connection)

- 2 The box below shows how a to export a certificate.

```
bash$ aica export -sn 2010 -o user.cer  
CA PKCS#12 file open  
Input PKCS#12 Password: (Enter Password)  
success to export a certificate (sn: 2010)
```

It is also possible to export a private key and a PKCS#12 formatted file in the same way. However, to access a user private key, you must enter the access password. In addition, to output a file, you must enter the output password.

## 4.10. Import Private Key

A CA can store user private keys. If an issued certificate and its private key pair are together in the store, it is possible to import a user private key.

- 1 Use the `aico import` operation to import a user private key into a CA. The box below shows the various commands.

### **aica import [Option]**

Option:

- sn **num** : Specifies the serial number
- key **file** : Specifies "file" as the private key file name

General option:

- sv **path** : Specifies the "Server name: CA name"
- ssl : Specifies SSL use in a remote CA connection
- clid **name** : Specifies the ID for the SSL client certificate
- u **loginid** : Specifies the CA user name (non-SSL connection)
- p **passwd** : Specifies the CA user password (non-SSL connection)

- 2 The box below shows how to import a private key.

```
bash$ aica import -sn 3247 -key user.key
```

```
CA PKCS#12 file open
```

```
Input PKCS#12 Password : (Enter Password)
```

```
get a private key from a file.
```

```
Open Private Key: (Enter Password)
```

```
save a private key to the CA. input new password.
```

```
Save Private Key: (Enter Password)
```

```
Verifying - Save Private Key: (Enter Password)
```

```
success to import a private key (sn:3247)
```

The user private must match with the certificate and the key pair of the specified serial number. In addition, you must set an access password to import a user private key into a CA.

## 4.11. Delete Private Key

A CA can store issued certificates and user private keys. It can also delete a user key from the key store. In addition, you can delete a certificate request that is stored in the CSR queue of a CA.

- 1 Use the `aica delete` operation to erase a user private key from a CA key store. The box below shows the various commands.

### **aica delete [Option]**

Option:

- sn **num** : Specifies the serial number
- key **file** : Specifies "file" as the private key file name
- csr **num** : Specifies the AcceptID for the CSR that is deleted

General option:

- sv **path** : Specifies the "Server name: CA name"
- ssl : Specifies SSL use in a remote CA connection
- clid **name** : Specifies the ID for the SSL client certificate
- u **loginid** : Specifies the CA user name (non-SSL connection)
- p **passwd** : Specifies the CA user password (non-SSL connection)

- 2 The box below shows how a to delete a user private key.

```
bash$ aica delete -sn 3247 -key  
CA PKCS#12 file open  
Input PKCS#12 Password : (Enter Password)  
success to delete a private key (sn: 3247)
```

## 4.12. Show Profile Settings

It is possible to execute a command that will show the CA profile settings, which include the CA issuer, the CA subject, the current serial number of the specified profile, the validity in days, and extension information.

- 1 Use the `aica print` operation to show a CA and its current profile settings. The box below shows the various commands.

### **aica print [Option]**

Option:

- pf **name** : Shows the profile settings for "name"
- all : Shows all profile settings

General option:

- sv **path** : Specifies the "Server name: CA name"
- ssl : Specifies SSL use in a remote CA connection
- clid **name** : Specifies the ID for the SSL client certificate
- u **loginid** : Specifies the CA user name (non-SSL connection)
- p **passwd** : Specifies the CA user password (non-SSL connection)

- 2 Show a CA and its profile settings by following the example shown in the box below.

```
bash$ aica print
CA PKCS#12 file open
Input PKCS#12 Password : (Enter Password)
Issuer :
  C=JP, O=nec corporation, OU=developer unit,
Subject :
  C=JP, O=nec corporation, OU=developer unit,
CA policy : C=option:ST=option:L=option:O=option:OU=option:
CN=option: Email=option
-----
Certificate Profile : SMIME user
certificate version : 3
current serial number: 4141
signature algorithm : SHA1WithRSAEncryption
certificate begin : at signing time
certificate end : Aug 21 15:00:00 2004
X509v3 extensions:
  x509 Basic Constraints:[critical]
    CA:FALSE
    PathLenConstraint:NULL
  x509 Key Usage:
    digitalSignature, nonRepudiation, keyEncipherment,
    dataEncipherment, key Agreement, (0xf8)
  x509 Authority Key Identifier:
    90:66:80:c3:a6:49:5b:65:65:ee:83:84:38:b0:37:...
  x509 Subject Key Identifier:
    90:66:80:c3:a6:49:5b:65:65:ee:83:84:38:b0:37:...
```

If you do not specify a profile name, enter “SMIME user” as the setting to show.

Next, is an explanation of the aica list operation that you can use to show all issued certificates in a list.

- 1 Use the aica list operation to show an issuance list for all certificates with profiles. The box below shows the various commands.

**aica list [Option]**

Option:

-pf **name** : Shows the profile settings for “name”

General option:

-sv **path** : Specifies the “Server name: CA name”

-ssl : Specifies SSL use in a remote CA connection

-clid **name** : Specifies the ID for the SSL client certificate

-u **loginid** : Specifies the CA user name (non-SSL connection)

-p **passwd** : Specifies the CA user password (non-SSL connection)

- 2 Show a issuance list for all certificates with profiles by following the example shown in the box below.

```
bash$ aica list
```

```
CA PKCS#12 file open
```

```
Input PKCS#12 Password : (Enter Password)
```

```
certificate list [profile : SMIME user]
```

state	serial	subject	notBefore	notAfter	revokedDate
--RK,	100,	"C=JP, O=org, CN=name100",	01/03/06	06:56,	02/03/06 06:56, 02/06/03 07:32
---K,	1,	"C=JP, O=org, CN=sample input",	01/06/03	06:52,	02/06/02 06:52,
C---	0,	"C=JP, O=org, CN=myname",	01/06/03	06:44,	02/24/05 06:44,

If you do not specify a profile name, enter “SMIME user” as the issuance list of certificates to show. In addition, for the state, “C” indicates a CA certificate, “R” indicates a revoked certificate, “E” indicates an expired period of validity, and “K” indicates a private key is stored on the CA side. The time display format is “year/month/day hour:minute.”



### 4.13. Update Profile Settings

It is possible to use a command to update a CA profile, which includes settings for the current serial number of a specified profile and the validity in days.

- 1 Use the `aica pfset` operation to set a profile. The box below shows the various commands.

#### **aica pfset [Option]**

Option:

- pf **name** : Sets the profile for "name" (Default: SMIME user)
- ver **num** : Specifies the version of the certificate and the CRL
- sn **num** : Specifies the serial number of the certificate and the CRL
- sec **num** : Specifies the period of validity (seconds) of the certificate and the CRL
- days **num** : Specifies the period of validity (days) of the certificate and the CRL
- start **time** : Specifies the start time of the certificate and the CRL (The format is "YYYYMMDDHHMMSSZ" and "." is clear)
- end **time** : Specifies the end time of the certificate and the CRL (The format is "YYYYMMDDHHMMSSZ")
- hash **alg** : Specifies the hash function used for the certificate signature and the CRL (sha1, md5, and md2 are available)
- sbjtmpl : Sets the subject DN template of the certificate.
- pol : Sets the issuance policy of the certificate.
- dnpol : Sets matching policy of CSR subject DN.

General option:

- sv **path** : Specifies the "Server name: CA name"
- ssl : Specifies SSL use in a remote CA connection
- clid **name** : Specifies the ID for the SSL client certificate
- u **loginid** : Specifies the CA user name (non-SSL connection)
- p **passwd** : Specifies the CA user password (non-SSL connection)

- 2 Set the validity period in days for a profile by following the example shown in the box below.

```
bash$ aica pfset -days 130
CA PKCS#12 file open
Input PKCS#12 Password : (Enter Password)

Update Profile information.
```

If you do not specify a profile name, enter "SMIME user" as the profile to use for setting the validity in days. If you change the validity in days for a CRL, make changes by specifying each CRL profile name.

```
aica pfset -pf CRL -days 3
aica pfset -pf ARL -days 7
aica pfset -pf CRL-All -days 3
```

The example above shows a validity period of three days for CRL and CRL-All and a validity period of seven days for ARL.

### 3 Set the subject template as follows.

```
bash$ aica pfset -sbjtmpl
CA PKCS#12 file open
Input PKCS#12 Password : (Enter Password)
-----
input Distinguished Name (DN) for subject template.
select directory tag (input number)
(1.C, 2.ST, 3.L, 4.O, 5.OU, 6.Quit)[1]:
Country [JP]:
select directory tag (input number)
(1.C, 2.ST, 3.L, 4.O, 5.OU, 6.Quit)[4]:
Organization [nec corporation]:
select directory tag (input number)
(1.C, 2.ST, 3.L, 4.O, 5.OU, 6.Quit)[5]:
Organization Unit [developer unit]:
select directory tag (input number)
(1.C, 2.ST, 3.L, 4.O, 5.OU, 6.Quit)[6]:
Update Profile information.
```

### 4 Set the profile policy as follows.

```
bash$ aica pfset -pol
CA PKCS#12 file open
Input PKCS#12 Password : (Enter Password)
-----
input profile working policy.
reuse same subject DN ? (y/n)[y]:
reuse expired subject DN ? (y/n)[n]:
reuse revoked subject DN ? (y/n)[n]:
reuse subject DN in updating period ? (y/n)[n]:
reuse same public key ? (y/n)[y]:
reuse expired public key ? (y/n)[n]:
reuse public key in updating period ? (y/n)[n]:
replace subject DN with template ? (y/n)[y]:
Update Profile information.
```

5 Set the CSR matching policy as follows.

```
bash$ aica pfset -dnpol
CA PKCS#12 file open
Input PKCS#12 Password : (Enter Password)
input CSR subject matching policy.
C : (0.option, 1.supplied, 2.matched) [0]:
ST : (0.option, 1.supplied, 2.matched) [0]:
L : (0.option, 1.supplied, 2.matched) [0]:
O : (0.option, 1.supplied, 2.matched) [0]:
OU : (0.option, 1.supplied, 2.matched) [0]:
CN : (0.option, 1.supplied, 2.matched) [0]:
UID: (0.option, 1.supplied, 2.matched) [0]:
Em : (0.option, 1.supplied, 2.matched) [0]:
Update Profile information.
```

#### 4.14. Update Extension Information for Profile

It is possible to set extension information for the profile of each certificate. When a certificate is issued, extension information, which is set for the specified profile, is added to the relevant certificate. To set extension information for a profile, you can follow the step-by-step setting information that appears in the dialog window.

- 1 Use the `aica pfext` operation to set a extension information for a certificate profile. The box below shows the various commands.

##### **aica pfext [Option]**

Option:

- pf **name** : Sets the profile settings for "name"
- bscons : Sets extension information for Basic Constraints
- keyusage : Sets extension information for Key Usage
- extkeyusage : Sets extension information for Extended Key Usage
- authkeyid : Sets extension information for Authority Key Identifier
- sbjkeyid : Sets extension information for Subject Key Identifier
- issaltname : Sets extension information for Issuer Alt Name
- sbjaltname : Sets extension information for Subject Alt Name
- certpol : Sets extension information for Certificate Policy
- polmap : Sets extension information for Policy Mapping
- crl dp : Sets extension information for the CRL Distribution Point
- authinfo : Sets extension information for Authority Info Access
- ocspnochk : Sets extension information for OCSP no check
- nscl : Sets extension information for the Netscape CRL URL
- nscomm : Sets extension information for the Netscape comment
- nstype : Sets extension information for the Netscape cert type
- crl num : Sets extension information for CRL Number
- issdp : Sets extension information for the Issuing Distribution Point
- crl reason : Sets extension information for CRL ReasonCode entry

General option:

- sv **path** : Specifies the "Server name: CA name"
- ssl : Specifies SSL use in a remote CA connection
- clid **name** : Specifies the ID for the SSL client certificate
- u **loginid** : Specifies the CA user name (non-SSL connection)
- p **passwd** : Specifies the CA user password (non-SSL connection)

- 2 To change extension information follow the example (BasicConstraints) shown below.

##### **aica pfext -bscons**

CA PKCS#12 file open

Input PKCS#12 Password: **(Enter Password)**

Add or delete X.509 extension for profile.

Select operation (a..add/d..delete/c..cancel)[a]:

After entering the password, you are prompted to add (or update if already present) extension information as specified by the options, to delete information, or to cancel the operation. If you select add, the setting items for the specified option will appear. If you select delete, the operation will end, but no change will occur if the specified option contains no information.

- To set the BasicConstraints, follow the example as shown in the box below.

```
Configure detail information for BasicConstraints extension.  
set critical flag (y/n)[y]: y    (Critical Flag Setting)  
set CA flag (y/n)[n]: n  
set pathLength (-1 means 'NULL') [-1]: -1  
'BasicConstraints' extension is added to 'SMIME user' profile.
```

The example above shows a setting for a user certificate with the CA flag OFF and the pathLength disabled (NULL).

- To set the KeyUsage, follow the example as shown in the box below.

```
Configure detail information for KeyUsage extension.  
set critical flag (y/n)[y]: n    (Critical Flag Setting)  
set digitalSignature flag (y/n)[y]: y  
set nonRepudiation flag (y/n)[y]: y  
set keyEncipherment flag (y/n)[y]: y  
set dataEncipherment flag (y/n)[y]: y  
set keyAgreement flag (y/n)[n]: n  
set keyCertSign flag (y/n)[n]: n  
set cRLSign flag (y/n)[n]: n  
'KeyUsage' extension is added to 'SMIME user' profile.
```

The example above shows settings for a user certificate with the encipherment flag ON for the signature, the repudiation prevention, the key, and the data. The flags for the certificate and the CRL issuance are OFF.

- To set the Extended Key Usage, follow the example as shown in the box below.

```
Configure detail information for ExtendedKeyUsage extension.  
Set Ext KeyUsage OID: 1.2.33.44  
  
Do you continue ? (y/n)[n]: n  
an extension is added to 'SMIME user' profile.
```

The example above shows how an object ID of 1.2.33.44 is set for the Extended Key Usage extension. If the OID is incorrect, the process will cancel.

- To set the Authority Key Identifier, follow the example as shown in the box below.

```
Configure detail information for AuthKeyID extension.
set key identifier (y/n)[y]: y
set issuer DN (y/n)[n]: n
set serial number (y/n)[n]: n
'AuthorityKeyIdentifier' extension is added to 'SMIME user' profile.
```

The Authority Key Identifier normally only has the key identifier enabled. Any other information is added as an extension.

- The moment that the SubjectKeyIdentifier is specified, the key identifier is set.
- To set the Issuer Alt Name, follow the example as shown in the box below.

```
Configure detail information for IssuerAltName extension.
input GeneralNames.
select a GeneralName (input number)
(1.Email, 2.DNS Name, 3.URL, 4.DN, 5.IP Address, 6.Other, 7.Quit)[7]:1
Email Address : caname@localhost
select a GeneralName (input number)
(1.Email, 2.DNS Name, 3.URL, 4.DN, 5.IP Address, 6.Other, 7.Quit)[7]:
'IssuerAltName' extension is added to 'SMIME user' profile.
```

The Issuer Alt Name sets additional CA information that does not appear under the DN of the certificate issuer. The example above adds an e-mail address.

- The moment that the Subject Alt Name is specified, a separate name for the subject is set.
- To set the Certificate Policy, follow the example as shown in the box below.

```
Configure detail information for CertificatePolicy extension.
set critical flag (y/n)[y]: n
Set Policy ID (ex. "1.2.33"): 1.2.33.44
Select field ID ...(1.CPS URI, 2.User Notice) [1]: 1
CPS Uri : http://caserv/ca-policy.txt

Do you continue ? (y/n)[n]: n
an extension is added to 'SMIME user' profile.
```

The above example defines a policy "1.2.33.44" and places its content in the address indicated by the CPS URI.

- To set the Policy Mapping, follow the example as shown in the box below.

Configure detail information for PolicyMappings extension.

Set issuerDomainPolicy : **1.2.33.44**  
 Set subjectDomainPolicy : **2.3.44.55**  
 an extension is added to 'SMIME user' profile.

The example above sets 1.2.33.44 as the domain policy of the certificate issuer and maps 2.3.44.55 as the certificate subject of the domain policy. It is only possible to set one policy mapping.

- To set the CRL Distribution Point, follow the example as shown in the box below.

Configure detail information for CRL Distribution Point extension.

Do you set Distribution Point ? (y/n)[y]: **y**  
 input GeneralNames.  
 select a GeneralName (input number)  
 (1.Email, 2.DNS Name, 3.URL, 4.DN, 5.IP Address, 6.Other, 7.Quit)[7]:**3**  
 URL : **http://www.testca.org/testca/out-CRL-All.crl**  
 select a GeneralName (input number)  
 (1.Email, 2.DNS Name, 3.URL, 4.DN, 5.IP Address, 6.Other, 7.Quit)[7]:  
 Do you set ReasonFlags ? (y/n)[n]: **n**  
 Do you set cRLIssuer ? (y/n)[n]: **n**  
  
 Do you continue ? (y/n)[n]: **n**  
 an extension is added to 'SMIME user' profile.

Specify one URL for the GeneralName to use as a CRL distribution point. If you want to add multiple distribution points, enter “y” after “Do you continue?” and add more points.

- To set the Authority Information Access, follow the example as shown in the box below.

Configure detail information for AuthorityInformationAccess extension.

Set Access Method  
 (1.OCSP, 2.CA Issuers 3.Time Stamping, 4.DVCS, 5.CA Repository) [1]:**1**  
 Set Access Location  
 input GeneralNames.  
 select a GeneralName (input number)  
 (1.Email, 2.DNS Name, 3.URL, 4.DN, 5.IP Address, 6.Other, 7.Quit)[7]:**3**  
 URL: **ocsp://caserver/myca/ocspcgi**  
 select a GeneralName (input number)  
 (1.Email, 2.DNS Name, 3.URL, 4.DN, 5.IP Address, 6.Other, 7.Quit)[7]:  
 an extension is added to 'SMIME user' profile.

The example above sets OCSP issuer information and specifies the URL address for the OCSP CGI.

- The moment the OCSP no check flag is specified, the flag is added.
- The moment the CRL Number is specified, the extension information is added.
- The moment the CRL Reason Code is specified, the entry extension information is added.
- To set the Issuing Distribution Point, follow the example as shown in the box below.

```
Configure detail information for Issuing Distribution Point extension.  
Do you set Distribution Point ? (y/n)[n]: n  
Do you set onlyContainsUserCerts ? (y/n)[n]: y  
Do you set onlyContainsCACerts ? (y/n)[n]: n  
Do you set ReasonFlags ? (y/n)[n]: n  
Do you set indirectCRL ? (y/n)[n]: n  
an extension is added to 'CRL' profile.
```

The example above specifies onlyContainsUserCerts, which will only list user revocation information in the CRL.

Depending on your display requirements, you can enter the other Netscape extension information related to comments and certificate usage.



#### 4.15. Add and Delete Profile

A CA can maintain multiple profiles for each group that issues certificates and it can delete a profile when it is no longer needed.

- 1 Use the `aica prof` operation to add and delete a profile. The box below shows the various commands.

##### **aica prof [Option]**

Option:

-add : Adds a profile  
-del : Deletes a profile  
-rename : Rename a profile

General option:

-sv **path** : Specifies the "Server name: CA name"  
-ssl : Specifies SSL use in a remote CA connection  
-clid **name** : Specifies the ID for the SSL client certificate  
-u **loginid** : Specifies the CA user name (non-SSL connection)  
-p **passwd** : Specifies the CA user password (non-SSL connection)

- 2 The box below shows how to add a profile. Select a profile template, add a profile name, and save the settings.

```
bash$ aica prof -add
CA PKCS#12 file open
Input PKCS#12 Password: (Enter Password)
-----
Add a certificate profile to this CA.

[1] Cross Cert Profile template
[2] Empty Profile template
[3] IPSEC Profile template
[4] Operator Profile template
[5] SMIME user Profile template
[6] SSL client Profile template
[7] SSL server Profile template
[8] Sub-CA Profile template
[0] Exit

Please select a template number [0]: 5

Selected profile template is "SMIME user Profile template"
Input Profile Name: testProf

do you continue this operation ? (y/n)[n]: n
Update CA information.
```

- 3 The box below shows how to delete a profile. When the profiles that a CA is maintaining appear in a list, specify the number of the profile that you want to delete and save the setting.

```
bash$ aica prof -add
CA PKCS#12 file open
Input PKCS#12 Password: (Enter Password)
-----
Delete a certificate profile from this CA.

[1] SMIME user
[2] Operators
[3] SMIME user2
[4] SMIME user3
[5] testProf
[0] Exit

Please select a profile number [0]: 5
do you continue this operation ? (y/n)[n]: n
Update CA information.
```

- 4 Change the name of the profile as follows. The profiles held by CA will be listed. Specify the number of the profile and specify a new name.

```
bash$ aica prof -rename
CA PKCS#12 file open
Input PKCS#12 Password : (Enter Password)
-----
Rename a certificate profile on this CA.

[1] SMIME user
[2] Operators
[3] SMIME user2
[4] SMIME user3
[0] Exit

Please select a profile number [0]: 4

Selected profile is " SMIME user3"

Input New Profile Name : testProf2
do you continue this operation ? (y/n)[n]: n
Update CA information.
```

## 4.16. Add and Delete Operator

When a CA is started and directly controlled from a local machine, you can begin operations after entering the master password. However, when a CA server is accessed remotely, you must first negotiate user (operator) authentication. To enable remote connection to a CA server, you can issue an operator certificate and set the ID/password and access rights.

- 1 Use the `aica user` operation to add and delete an operator. The box below shows the various commands.

### **aica user [Option]**

Option:

- add : Adds an operator
- addop : Adds an operator that will use SSL authentication
- del : Deletes an operator
- mod : Updates operator information
- cpw : Updates an operator password

Option (offline only):

- addop : Adds an operator for SSL authentication.
- addraop : Issues an RA administrator certificate.
- smlib : Specifies PKCS#11 library (for RA administrator).
- smlabel : PKCS#11 label name (for RA administrator)

General option:

- sv **path** : Specifies the "Server name: CA name"
- ssl : Specifies SSL use in a remote CA connection
- clid **name** : Specifies the ID for the SSL client certificate
- u **loginid** : Specifies the CA user name (non-SSL connection)
- p **passwd** : Specifies the CA user password (non-SSL connection)

- 2 The box below shows how a to add an operator.

```
bash$ aica user -addop
CA PKCS#12 file open
Input PKCS#12 Password: (Enter Password)
-----
Add a new Operator to this CA.
.....00
.00
Input PASS Phrase: (Enter Password)
Verifying - Input PASS Phrase: (Enter Password)
success to modify CA user.
```

3 The box below shows how to delete an operator.

```
bash$ aica user -del
CA PKCS#12 file open
Input PKCS#12 Password: (Enter Password)
-----
Delete a CA user from this CA.

Enter delete user name: CAOperator003
success to modify CA user.
```

When an operator is issued a certificate, that operator is given full administrator rights to operate the CA server. To change access rights, change the user information as shown in the box below. If you do not know the operator name, refer to the ca.passwd file.

```
bash$ aica user -mod
CA PKCS#12 file open
Input PKCS#12 Password:
-----
Modify a CA user information.

Enter user name: CAOperator001
Enter user ID (Serial Number) [0]:
Enter group ID (Profile Number) [0]:
Enter user Grant [0x73f7ffff]: BSLAPECeQR
success to modify CA user.
```

Use the variables below to specify access rights.

<b>B</b>	The bind request is required to connect to a CA server.
<b>S</b>	The sign operation request authorizes immediate certificate issuance.
<b>L</b>	The list operation request authorizes obtaining a certificate status list.
<b>A</b>	The profile list request authorizes obtaining the profile list maintained by a CA.
<b>X</b>	The extension request authorizes extended operations. Normally this value is not specified.
<b>P [vadm]</b>	The profile operation request authorizes control of general profile information. Adding v, a, d, or m (view, add, delete, or update) after the P authorizes each individual operation. A P standing alone authorizes all operations.
<b>E [vadm]</b>	The profile extension request authorizes control of extended certificate information. Adding v, a, d, or m (view, add, delete, or update) after the E authorizes each individual operation. An E standing alone authorizes all operations.

<b>C [urne]</b>	The certificate operation request authorizes control of user certificates. Adding u, r, n, or e (update, revoke, unrevoke, or export) after the C authorizes each individual operation. A C standing alone authorizes all operations.
<b>K [ied]</b>	The key operation request authorizes control of user private keys. Adding i, e, or d (store, export, or delete) after the K authorizes each individual operation. A K standing alone authorizes all operations.
<b>Q [pedj]</b>	The CSR operation request authorizes control of the CSR queue. Adding p, e, d, or j (post, export, delete, or judge issuance) after the Q authorizes each individual operation. A Q standing alone authorizes all operations.
<b>R [ie]</b>	The CRL operation request authorizes CRL control. Adding i or e (issue or export) after the R authorizes each individual operation. An R standing alone authorizes both operations.

## 5. Other Operations

This chapter describes other NAREGI CA commands, such as the certificate viewer and the converter.

### 5.1. Certificate Store

Use the `aistore` command to import or export a certificate or a CRL to or from a certificate store. A certificate store requires signature verification processing to take charge of a certificate or a CRL.

A certificate store is divided into “root,” “sub,” “other,” and “my” categories of which “root,” “sub,” and “other” can be used to store a certificate or a CRL. In addition, the “root” stores a root CA certificate, the “sub” stores a subordinate CA certificate, and the “my” stores the main user certificate.

- The box below shows the various commands.

#### **aistore** [Operation] [Option]

Operation:

- i **file** : Imports a file with the name “file”
- e **file** : Exports the specified item using the name “file”
- l : Makes a list of items stored in the specified store
- p : Shows the specified item
- d : Deletes the specified item from a store

Option:

- st **store** : Specifies the store name  
(Specify one: my, other, sub, or root)
- tp **type** : Specifies the store type (Specify one: cert or crl)
- id **id** : Specifies the unique ID for an item
- ef **format** : Specifies the export file format  
(Specify one: der, pem, pk7, pk12)
- pw **pwd** : Specifies the password used for exporting

- **Show Certificate and CRL Lists**

To show a list of certificates in the my store follow the example in the box below.

```
bash$ aistore -l
```

Option:

```
[unique-id]    subject                serialNumber
-----
[mycert00] C=JP, O=test org, CN=t.yamada, 30000
```

By default not specifying anything specifies the my store. The list format shows the unique ID, the subject, and the serial number as a set.

To show a list of certificates in the root store follow the example below.

```
aistore -l -st root -tp crl
```

- **Import Certificate or CRL**

It is possible to automatically identify the type of certificate and import it into the appropriate store.

Use the -id option to specify a unique name for the content within a store. If you do not specify an ID, the certificate subject automatically determines a unique ID. To import, you can use a X.509 DER certificate file format or a PEM certificate (using PKCS#7 or PKCS#12) file format.

```
aistore -i myca.cer -id myca
```

To import an CRL file, first import the CA certificate that issued the CRL then perform the operation.

```
aistore -i myca.crl
```

- **Show Certificate or CRL**

To show an optional certificate from the my store, you must specify the store name and either the unique name for a certificate or a CRL. The unique ID is specified at the time of import or assigned automatically. However, if you do not know an ID, check it by listing all the content of a certificate store. The execute command is shown below.

```
aistore -p -id mycert01
```

To show an optional CRL from the root store, execute the command shown below.

```
aistore -p -id cacrl01 -st root -tp crl
```

- **Export Certificate or CRL**

To export an optional certificate from the my store, you must specify the store name and either the unique name for a certificate or a CRL. The execute command is shown below.

```
aistore -e out.cer -id mycert01
```

To export an optional CRL from the root store, execute the command shown below.

```
aistore -e out.crl -id cacrl01 -st root -tp crl
```

- **Delete Certificate or CRL**

To delete an optional certificate from the my store, you must specify the store name and either the unique name for a certificate or a CRL. The execute command is shown below.

```
aistore -d -id mycert01
```

To delete an optional CRL from the root store, execute the command shown below.

```
aistore -d -id cacrl01 -st root -tp crl
```



## 5.2. Verify Certificate

Use the `certvfy` command to verify a certificate. To verify a CA certificate or a CRL, it must first be imported into a certificate store. After specifying a certificate file formatted in X.509 PEM or DER, you can use the verification operation.

- The box below shows the various commands.

### **certvfy [Option] file**

Option:

- depth **num** : Specifies the path depth during verification
- ifcrl : Performs a revocation only when a CRL is present
- nocrl : Does not perform a revocation check during verification
- novfycrl : Does not perform CRL verification

Follow the example in the box below to use the verification process when a certificate store contains CA certificates but no CRL files.

```
bash$ certvfy 05.cer
[0:ok ] C=JP, O=test org, CN=test,
[1:err] C=JP, O=test org, CN=hoge, /Email=hoge@localhost
05.cer : CERT Verify Failed (CRL not found) : 1
```

Complete verification is performed when you do not specify any option. In the example above, a verification error occurred and the operation terminated because the revocation status of a certificate could not be verified (could not discover a CRL in the certificate store) at a depth of one.

The example below uses the same conditions as above except the `-ifcrl` option is used during the verification operation.

```
bash$ certvfy 05.cer
[0:ok ] C=JP, O=test org, CN=test,
[1:ok ] C=JP, O=test org, CN=hoge, /Email=hoge@localhost
05.cer : CERT Verify OK
```

The `-ifcrl` option only verifies the revocation status if a CRL is present. As shown in the box above, if a CRL is not present, the verification process will run as if all certificates are valid, which makes the operation a success. In addition, even if a CRL is present, you can specify the `-nocrl` option and the verification process will not check the revocation status.

### 5.3. View Certificate

.....

Use the certview command to view as text each type of certificate file, private key file, and PKSC file. First, we describe the commands then we show some usage examples.

- The box below shows the various commands.

```
certview [Option] file1 file2 ...  
Option:  
-p passwd : Specifies the password for the input file
```

- The box below shows the file viewable file types.

```
Certificate: *.cer, *.pem, *.p7b  
Cross certificate: *.ccp  
Private key: *.key, *.pem  
Key parameter: *.pem  
CRL: *.crl, *.pem, *.p7b  
Certificate request: *.req, *.pem, *.p10  
PKCS#12: *.p12, *.pfx  
PKCS#8: *.ctx, *.pem
```

The viewer automatically identifies the file type and shows it in plain text. The view can show the content regardless of the file type (e.g., DER, PEM, BASE64).

## 5.4. Change Certificate

Use the certconv command to inter-convert file types for certificates, private keys, and PKCS. First, we describe the commands then we show some usage examples.

- The box below shows the various commands.

**certconv** [Option] **file1** **file2**...

Option:

-p12 <b>file</b>	: Stores a PKCS#12 file using the name "file"
-p7b <b>file</b>	: Stores a PKCS#7 file using the name "file"
-cert <b>file</b>	: Stores a certificate file using the name "file"
-crl <b>file</b>	: Stores a CRL file using the name "file"
-key <b>file</b>	: Stores a private key using the name "file"
-crtp <b>file</b>	: Stores a cross certificate file using the name "file"
-p10 <b>file</b>	: Stores a certificate request file using the name "file"
-p8 <b>file</b>	: Stores a PKCS#8 (key) file using the name "file"
-pwin <b>pwd</b>	: Specifies a password for the input file
-pwout <b>pwd</b>	: Specifies a password for the output file
-pem	: Converts the output file to PEM format (Default)
-der	: Converts the output file to DER format
-depth <b>dp</b>	: Outputs a certificate using the depth "dp"
-noenc	: Does not encrypt the output file (Private key)

- **Convert between PEM and DER Formats**

```
certconv -der -cert out.cer in.cer
```

```
certconv -pem -p7b out.p7b in.p7b
```

Use the -der option to output a file in DER format.

Use the -pem option to output a file in PEM format (default).

- **Certificate Chain or CRL to PKCS#7 (\*.p7b)**

```
certconv -p7b out.p7b in.cer in.crl ca.cer
```

Input in.cer, in.crt, or ca.cer and the output will be out.p7b.

- **PKCS#7 to Certificate of Optional Depth**

```
certconv -depth 0 -cert out.cer -crl out.crl in.p7b
```

In this example a certificate with a depth of zero (top of the chain) and CRL are extracted as in.p7b files.

- **Certificate Chain and Private Key to PKCS#12 (\*.p12)**

```
certconv -p12 out.p12 in.key in.cer ca.cer
```

```
certconv -p12 out.p12 in.key in.cer ca.cer upca.cer
```

Input in.key, in.cer, or ca.cer and the output will be out.p12.

The PKCS#12 output file is compatible with Netscape.

- **PKCS#12 to Certificate of Optional Depth, Private Key**

```
certconv -key out.key -cert out.cer in.p12
```

```
certconv -depth 1 -cert out.cer in.p12
```

Input in.p12 and the output will be out.cer and out.key. Both Microsoft and Netscape can import files in the PKCS#12 (PFX) format. In addition, in the first example, the depth of the certificate was not specified, so a file with the deepest level, a user certificate, is output.

- **Convert between PKCS#8 and PEM Private Key**

```
certconv -noenc -p8 out.crtx in.key
```

```
certconv -p8 out.crtx in.p12
```

The input contains a private key in the in.key or the in.p12 format and the output is PKCS#8-formatted out.crtx file. In this example, it is possible to output an unencrypted key using the -noenc option.

- **Convert between a Certificate and a Cross-Certificate Pair**

```
certconv -ccp out.ccp a2b.cer b2a.cer
```

```
certconv -depth 1 -cert a2b.cer in.ccp
```

In the upper example, the input is a2b.cer and b2a.cer and the output is out.ccp. In the lower example, the input is in.ccp, the cross-certificate pair depth equals one, and a certificate called issuedByThisCA is output.

## 5.5. Create Certificate Request

The grid-certreq command is a shell script used for certificate acquisition. From this shell script you can call certreq commands to manage the certificate lifecycle. Certificate operations include the -new option, the -rvk option, and the -upd option. With the -new option, you can connect to an RA server where you can issue a certificate. The -rvk and the -upd options are used to revoke and to update a certificate. First, however, the SSL client must be authenticated, then after establishing a connection, authenticate the client certificate.

The box below shows the various commands.

### **certreq [Option]**

#### Option:

- algo **alg** : Indicates the algorithm of the generated key pair  
rsa, dsa, and ecdsa are available (Default: rsa)
- size **num** : Specifies the key-pair length (default: 512 bits)
- req **file** : Stores a certificate request using the "file" name
- key **file** : Stores a private key using the name "file"
- kp **passwd** : Specifies the password used for private key storage
- noenc : Stores a private key without encrypting it
- der : Specifies DER as the storage format for the certificate Request
- alt : Specifies Subject Alt Name
- s : Specifies simple as the subject input mode

#### Remote operation option:

- new **license** : Creates a certificate request and requests a certificate  
Specifies the ID for a one-time license
- rvk : Revokes an SSL client certificate
- upd : Updates an SSL client certificate
- recv **acID** : Obtains an acID certificate (acceptID)
- in **file** : Inputs the CSR file named "file"
- cer **file** : Stores a certificate file using the name "file"
- cacer **file** : Stores a certificate file using the name "file"
- p12 **file** : Stores a PKCS#12 file using the name "file"

#### Connection option:

- sv **path** : Specifies the "Server name: CA name"
- pt **port** : Specifies the port number "port"
- ssl : Specifies SSL use in a remote CA connection
- clid **name** : Specifies the ID for the SSL client certificate
- clcer **name** : Specifies the file name for an SSL client certificate
- clkey **name** : Specifies the file name for an SSL client private key

## 5.6. Web Enroll Auxiliary Tool

Web Enroll uses an ID/Password or a one-time license ID to authenticate a user. At this time, the en.password and en.license files must be located, and aienrtool is available as a data creation support tool. This section explains the command format first, and then shows an example.

- The command format is as follows.

```
aienrtool [Option]
Option:
  -csv file      : Specifies a CSV file for batch generation of
                    passwords or batch mail notification of licenses.
[enroll passwd file mode]
  -u name        : Specifies a user name.
  -p passwd       : Specifies a password.
[enroll license file mode]
  -lic            : Specifies a license generation mode.
  -n num         : Specifies the number of output licenses.
  -s num         : Specifies a start number.
  -hd text       : Specifies a license header character string.
  -iv text       : Specifies an initial vector.
[generate license and send email mode]
  -lsend num     : Executes batch mail notification of licenses.
                    Specify the corresponding RA RegInfo as num.
```

- en.passwd generation mode (one line)

```
bash$ aienrtool -u user001 -p abcde
user001:XLvYdWJfeNdVxVq0fLG4K3WZI4Pn64o=
```

Input a user name and a password as -u and -p, and output en.passwd file format data to the standard output. Redirect to save a file, as follows.

```
aienrtool -u user001 -p abcde > en.passwd
```

- en.passwd generation mode (CSV)

```
bash$ more myuser.csv
user001,abcde
user002,fghij
bash$ aienrtool -csv myuser.csv
user001:UmvvpBSF9vBogm6gTpgKqkHiO/nQ2EE=
user002:GsPapVqiV5+d3NFNUcjluPCL484r0tA=
```

Specify a CSV file having the format of “user,passwd” as -csv and output en.passwd file format data to the standard output.

- en.license generation mode

```
bash$ aienrtool -lic -n 4 -s 0 -hd MYID00 -iv randseed17asf4d17ds
MYID00-YQUJMW-EF8197-C1UWGS
MYID00-YQT4MV-M1GSOX-SILZ7O
MYID00-YQ08MU-2POD7I-46ZSST
MYID00-YQ1ZMT-07L1XK-WUBBX4
```

When the -lic option is specified, the license ID generation mode is selected. Because Web Enroll simply matches the character string included in the en.license file with the character string posted from the user, it operates without problem even with a license generated by some other tool. Use this tool if a license ID cannot be easily generated.

aienrtool generates a unique license, instead of outputting a complete random character string. If the same start number, header character string, and initial vector are used when a license is generated, the same license number is generated.

Follow this procedure to generate a unique license ID:

- <1> Determine a header character string. It is "MYID00" in the above example.
- <2> Determine a start number. It is "0" in the above example.
- <3> Determine an initial vector. It is "randseed17asf4d17ds" in the above example.

To output 1000 license IDs, for example, specify -n 1000. To output another 1000, a unique new license ID can be generated by specifying -s 1000 so that the start number does not overlap.

- License batch mail notification mode (CSV)

```
bash$ more mail.csv
test000@localhost
test001@localhost
bash$ aienrtool -csv mail.csv -lsend 1
proceed to generate a license ID : line 1
proceed to generate a license ID : line 2
```

Specify a CSV file having the format of an e-mail address as -csv, and perform license batch mail notification for specified [RAd RegInfo \*] and updating en.license or LDAP entry data.

## 5.7. aica.cnf Tools

If you want a CA server or a CRL publisher to start up automatically, you must set an encrypted password for the aica.cnf file. Use the aiconftool utility for password encryption settings and for other shell script operations. First, we describe the commands then we show a usage example.

The box below shows the various commands.

```
aiconftool [Option]
[Item option]
  -add           : Adds or updates an item
  -del           : Deletes an item
  -mod           : Updates an item if it is present
  -s section    : Specifies a section name
  -i item       : Specifies an item name
  -v value     : Specifies an item value
  -enc           : Encrypts an item value
[CA option]
  -addca         : Registers the CA on a server
  -delca         : Deletes the CA from a server's registration
                  information
  -addcrl        : Adds CRL publisher settings
  -delcrl        : Deletes CRL publisher settings
  -addenr        : Adds web enrollment settings
  -delenr        : Deletes web enrollment settings
  -ca name      : Specifies the CA name
  -path path    : Specifies the CA path
  -pw passwd   : Specifies the CA password
  -conf path   : Specifies the aica.cnf path
```

Change CA Password used for Automatic Start

```
bash$ aiconftool -add -s CAd -i capwd.0 -v abcde -enc
```

Update the value for the capwd.0 item in the CAd section. In this case the password "abcde" is encrypted and updated.

The aiconftool utility does not output an operation result. Accordingly, open the aica.cnf file directly to check if the value has been updated.



.....

```
bash$ aira csr -show 3
>>>>>>>>>>>> CSR information >>>>>>>>>>>>
Certificate Request:
Data:
  Version: 1 (0x0)
  Subject: C=JP, O=TEST, OU=GRID, CN=mariri,
  Subject Public Key Info:
    Public Key Algorithm: rsaEncryption
    ...
>>>>>>>>>>>> CSR PEM >>>>>>>>>>>>
-----BEGIN CERTIFICATE REQUEST-----
MIIDRDCCAq0CAQAwZzELMAkGA1UEBhMCSIAxDALBgNVBA
oTBFRFU1QxDALBgNV
...
-----END CERTIFICATE REQUEST-----
```

```
bash$ aira csr -reject 3
>>>>>>>>>>>>>>> CSR information >>>>>>>>>>>>>>>
Certificate Request:
Data:
  Version: 1 (0x0)
  Subject: C=JP, O=TEST, OU=GRID, CN=mariri,
  Subject Public Key Info:
    Public Key Algorithm: rsaEncryption
    ...
do you really reject this CSR ? (y/n)[y]:
```

- Locating certificate

```
bash$ aira csr -cert newcert.pem  
locate a certificate (accID=0000003) ... done.
```

By specifying a certificate with the `csr -cert` option, whether a CSR having the same key pair exists can be checked. If there is no problem, the certificate is converted into the PKCS#7 format for external distribution and then distributed. If the `aienroll` process operates, issuance of the certificate is reported to the user.