管理者ガイド

NAREGI Grid Middleware

CA

(Certification Authority)

2008年 **7**月

国立情報学研究所

Documents List

◆ Administrator's Guide Group

- **Administrator's Guide, NAREGI Grid Middleware IS(Distributed Information Service)**
- **Administrator's Guide, NAREGI Grid Middleware IS(Distributed Information Service) - LRPSConfig -**
- **Administrator's Guide, NAREGI Grid Middleware SS(Super Scheduler)**
- **Administrator's Guide, NAREGI Grid Middleware GridVM(Grid Virtual Machine)**
- **Administrator's Guide, NAREGI Grid Middleware Portal**
- **Administrator's Guide, NAREGI Grid Middleware PSE(Problem Solving Environment)**
- **Administrator's Guide, NAREGI Grid Middleware WFT(GUI Workflow Tool)**
- **Administrator's Guide, NAREGI Grid Middleware GVS (Grid Visualization System)**
- **Administrator's Guide, NAREGI Grid Middleware DataGrid**
- **Administrator's Guide, NAREGI Grid Middleware CA(Certification Authority)**
- **Administrator's Guide, NAREGI Grid Middleware UMS(User Management Server)**
- **Administrator's Guide, NAREGI Grid Middleware Authorization Service**
- **Administrator's Guide, NAREGI Grid Middleware Renewal Service**
- **Administrator's Guide, NAREGI Grid Middleware SBC(Synchronous Data Transfer Library)**
- **Administrator's Guide, NAREGI Grid Middleware Mediator**

◆ User's Guide Group

- **User's Guide, NAREGI Grid Middleware IS(Distributed Information Service)**
- **User's Guide, NAREGI Grid Middleware Portal**
- **User's Guide, NAREGI Grid Middleware PSE(Problem Solving Environment)**
- **User's Guide, NAREGI Grid Middleware**
  PSE(Problem Solving Environment) - Command line Interface -
- **User's Guide, NAREGI Grid Middleware WFT(GUI Workflow Tool)**
- **Programming Guide, NAREGI Grid Middleware WFT(GUI Workflow Tool)**
- **User's Guide, NAREGI Grid Middleware GVS(Grid Visualization System)**
- **User's Guide, NAREGI Grid Middleware CA(Certification Authority)**
- **User's Guide, NAREGI Grid Middleware UMS(User Management Server)**
- **User's Guide, NAREGI Grid Middleware Authorization Service**
- **User's Guide, NAREGI Grid Middleware Renewal Service**
- **User's Guide, NAREGI Grid Middleware**
  SBC (Synchronous Data Transfer Library)
- **User's Guide, NAREGI Grid Middleware Mediator**

目次

1. はじめに

Grid プロトタイプ CA（以下 NAREGI CA）の特徴や動作環境を説明します。

1.1. NAREGI CA の特徴

NAREGI CA は、UNIX 上で動作する認証局サーバ･コマンド群です。鍵生成や証明書の発行、検証、保管など、各種ユーティリティコマンドのパッケージです。このパッケージは下記のコマンド群によって構成されています。

- CA 構築　aisetup.sh
  CA 構築を行うシェルスクリプトです。このコマンドにより作成された CA は、aica コマンドによって運用することができます。このコマンドは CA サーバの設定ファイルである aica.cnf の更新を行うため、aicad（CA サーバ）や aicrlpub（CRL Publisher）への登録も自動的に行います。ローカルファイルアクセスによる CA 運用の他、CA サーバが起動している場合はリモート運用も可能です。

- CA 構築　ainewca.sh
  CA 構築を行うシェルスクリプトです。このコマンドにより作成された CA は、aica コマンドによって運用することができます。このコマンドは CA サーバの設定ファイルである aica.cnf の更新を行いません。このため、ローカルファイルアクセスによる CA 運用が可能です。

- RA 構築　ainewra.sh
  RA 構築を行うシェルスクリプトです。このコマンドにより特定の CA に対応した RA の設定を aica.cnf に対して追加します。RA の構築を行うことで、クライアントマシンから CA サーバに直接アクセスすることを防ぎます。

- CA サーバ aicad
  CA サーバ（デーモン）です。設定ファイルの aica.cnf に登録してある CA に対して、リモートアクセスを可能にします。証明書の発行、更新、失効、エクスポート、秘密鍵の保管とエクスポート、CRL の発行、CSR の受付け、承認、証明書プロファイルの設定、証明書発行リストの表示など独自プロトコルである LCMP に定義されている各 CA オペレーションが実行できます。

- CRL Publisher aicrlpub
  CRL Publisher サービスです。一定時間ごとにローカルファイルアクセスまたはリモートアクセスによって CRL/ARL を発行し、ローカルディレクトリや LDAP サーバに CRL/ARL を出力します。

- CA 運用 aica
  CA 運用コマンドです。証明書の発行、更新、失効、エクスポート、秘密鍵の保管とエクスポート、CRL の発行、CSR の受付け、承認、証明書プロファイルの設定、証明書発行リストの表示、CA サーバオペレータ設定が行えます。ローカルファイルアクセスによる CA 運用の他、リモートアクセスによる CA 運用も行えます。

- RA サーバ aired / aienroll / 各種 WEB Enroll CGI
  RA サーバを構成する各モジュールです。RA サーバは1つのサーバではなく、airad デーモンと Web Enroll CGI、セッションファイル管理を行う aienroll の 3 つのモジュールにより構成されます。
  airad は、デーモンとして動作し certreq もしくは grid-certreq コマンドからのリモート証明書発行要求や更新、失効を受け付けます。Apache などの WEB サーバより CGI が呼び出されると、CGI 経由で CA サーバに証明書の発行や更新、失効を要求します。即時発行と CA 運用者確認発行があり、CA 運用者確認発行の場合は aienroll によって、一定時間ごと証明書発行確認を行います。証明書発行や失効前にユーザ認証を行うことができ、匿名、ID/Password、License ID（One Time ライセンス）、SSL 認証、License ID/Challenge PIN の各種認証を行った上でユーザに証明書ライフサイクル管理機能を提供します。また、aienroll により証明書更新通知メールを送付することが可能です。
  特に、License ID / Challenge PIN 認証モードを使用している場合、RA オペレータによって LDAP サーバをベースとしたユーザ管理が行われます。また、この RA オペレータを管理する RA アドミニストレータも用意され、RA 運用の権限分離が行われます。これらの管理を行うために、WEB インタフェースが用意されています。

- 証明書ストア操作 aistore
  証明書ストアの操作コマンドです。証明書の検証を行う場合、トラストポイントからのパス検証が必要であり、ルート CA 証明書や CRL などの保管が必要です。

- 証明書検証 certvfy
  証明書の検証コマンドです。aistore によってストアに保管した CA 証明書や CRL を使用して証明書の検証を行います。

- 証明書ビューア certview
  証明書には数多くのファイル形式が存在し、このファイルをわかりやすくテキスト表示するアプリケーションです。CA の運用には必須のユーティリティです。

- 証明書コンバータ certconv
  証明書コンバータは、証明書や秘密鍵に関連する数多くのファイルフォーマットを相互変換します。CA の運用には必須のユーティリティです。

- CSR 作成 certreq / grid-certreq
  certreq コマンドは、鍵ペアを生成し PKCS#10 形式の証明書要求を作成するコマンドです。作成した PKCS#10 を CA で読み込むことで証明書発行を行います。なお、certreq コマンドは CA サーバもしくは RA サーバに接続し、リモートマシンからの証明書発行要求や更新、失効を行うなど、証明書ライフサイクル管理機能が実装されています。
  grid-certreq コマンドは、グリッド環境用に作成されたシェルスクリプトで、内部的に certreq コマンドを呼び出します。証明書ライフサイクル管理機能を提供するとともに、Globus や UNICORE などのユーザ環境に対応した証明書の出力を行います。

1.2. 動作環境について

NAREGI CA は次のような環境で動作します。

対応機種	PC/AT 互換機（DOS/V） SUN MICROSYSTEMS SPARC マシン
CPU	Pentium III 500MHz 以上推奨
対応 OS	Turbolinux Server 6.5 Miracle Linux Standard Edition V2.1 Solaris 2.6, 2.8
メモリ	128MB 以上推奨
ハードディスク容量	ソフトウェアインストール領域として 10M 必要 証明書発行数に応じて必要 （例．証明書 1000 枚で 10M 程度）
表示	800 × 600 ドット以上 ハイカラー以上推奨

2. CA・RA の構築

NAREGI CA コマンドを使用して CA を構築する方法を説明します。aisetup.sh または ainewca.sh を実行し、表示画面に従って操作を行うことで、簡単に CA を構築できます。

2.1. 新規 CA 構築（CA サーバ登録あり）

1 新しく CA を構築し CA サーバに登録する場合は、”aisetup.sh” シェルスクリプトを使います。コマンドの形式は次のとおりです。

```
aisetup.sh [オプション] CA 名
オプション:
      -add              : CA を構築しサーバに登録します（標準）
      -del              : CA をサーバの登録情報から削除します
      -keyalgo algo     : 秘密鍵アルゴリズムを指定します(rsa(標準),dsa,ecdsa)
      -keysize size     : 生成する鍵長を指定します(標準:1024bit)
      -days day         : 証明書の有効日数を指定します
      -start time       : 開始時間を指定します "YYYYMMDDHHMMSSZ"
      -end time         : 終了時間を指定します "YYYYMMDDHHMMSSZ"
      -p12 file         : PKCS#12 を指定して CA を構築します
      -p11lib lib       : HSM (PKCS#11)のライブラリを指定します
      -p11label name : PKCS#11 トークンラベルを指定します
```

-add オプションは省略できます。CA 名を指定してコマンドを実行すると、NAREGI CA のインストールディレクトリ/CA 名/ 以下に CA 情報ファイルを配置します。なお、CA の構築パターンの概要は次のとおりです。

証明書と秘密鍵を新規に作成する	-keysize オプションで鍵長を指定し、新規に秘密鍵･公開鍵の生成と自己署名証明書の発行を行います。この場合の CA は、最上位の CA（RootCA）となります。なお、-p11lib, -p11label オプションを指定することで、HSM に対応した CA を構築することができます。
PKCS#12 をインポートする	上位の CA から証明書と秘密鍵のファイル（PKCS#12）を配布され、そのファイルを利用して CA を構築します。この場合の CA は、下位 CA となります。

2 新規に証明書と秘密鍵を作成する場合、下記のようにコマンドを実行します。コマンドを実行すると Step.1 CA マスタパスワードの設定が始まります。

```
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 :   （パスワードを入力）
Verify - Input Master Passwd :   （パスワードを入力）
```

3 CA マスタパスワードの設定が終わると、Step.2 CA 秘密鍵の生成が始まります。鍵の生成が終わるまでお待ちください。ここで、CA 情報を保存するディレクトリが表示されますが、通常"NAREGI CA インストールディレクトリ/CA 名/"に CA データが作成されます。

```
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)
.....................................o
.................................o
```

4 CA 秘密鍵の生成が成功すると、証明書サブジェクトの入力を行います。

```
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]:
```

なお、各タグフィールドの概要は次のとおりです。

C	国（Country）の名称です。必ず半角2文字のアルファベット大文字を入力します。日本の場合は「JP」です。
ST	州名（State）の名称です。とくに入力する必要はありません。
L	位置（Location）の名称です。とくに入力の必要はありません。
O	組織（Organization）の名称です。半角 64 文字まで入力可能で、日本語の入力もできます。
OU	下位組織（Organizational Unit）の名称です。半角 64 文字まで入力可能で、日本語の入力もできます。
CN	一般（Common Name）の名称です。半角 64 文字まで入力可能で、日本語の入力もできます。
EMAIL	電子メールアドレス（Email）です。半角 64 文字まで入力可能です。

5 サブジェクトの入力が終了すると、CA 証明書の拡張情報の修正を行うか聞かれます。通常は"n"を指定します。認証局の運営規定にて CA 証明書のプロファイルが規定されている場合は、それに従って修正を行います。

```
do you modify CA certificate extension ? (y/n)[n]: y (←通常は n を指定)

        X509v3 extensions:
           x509 Basic Constraints:[critical]
              CA:TRUE
              PathLenConstraint:NULL
           x509 Key Usage:
              keyCertSign, cRLSign (0x06)
           x509 Authority Key Identifier:
              f5:f3:3a:c6:41:ed:b0:cb:69:06:ec:78:1d:49:a5:06:bb:9d:3e:96:
           x509 Subject Key Identifier:
              f5:f3:3a:c6:41:ed:b0:cb:69:06:ec:78:1d:49:a5:06:bb:9d:3e:96:
--
1. Basic Constraints            2. Key Usage                3. Extended Key Usage
4. Authority Key Identifier     5. Subject Key Identifier   6. Issuer Alt Name
7. Subject Alt Name             8. Certificate Policy       9. Policy Mapping
10. CRL Distribution Point      11. Authority Info Access   12. OCSP no check
13. Netscape CRL Url            14. Netscape Comment        15. Netscape Cert Type
0. Quit
select certificate extension number [0]: (←変更を加える拡張情報の番号を入力)
```

6 証明書のサブジェクトや有効期間の確認を行い、CA 証明書を発行します。

```
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.
```

7 CA サーバの設定ファイルである aica.cnf の更新を行います（Step.3）。この更新はスクリプトによって自動的に行われ、ユーザに対して入力は要求されません。

```
Step 3. CA Server registration
   * new Certificate Authority is created successfully.
   * now this CA will be registerd 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 Publihser : /usr/local/naregi-ca/testca
success to unregist RAd RegInfo : dummy
success to regist a RAd RegInfo: localhost:testca
-----------------------------------------------------------------
CA server registration is finished successfully.

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

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

このスクリプトでは、CA サーバへの CA の登録と、CRL Publisher 情報の追加、WEB Enroll 情報の追加を行います。既に同名の CA が登録されている場合は、登録をスキップします。また、WEB Enroll は現在 1 つしか設定できないため、2 回目以降の aisetup.sh スクリプト実行では、WEB Enroll 更新は行われません。新しい CA に対して WEB Enroll を設定したい場合は、手動で aica.cnf の設定を行ってください。

8 Step.2 で表示される CA ディレクトリには、CA 運用に必要なデータが構成されます。CA を構成する各種ファイルの説明は次のとおりです。

- ca.p12 について

  ca.p12 は CA の自己署名証明書と秘密鍵のペアを持つ PKCS#12 ファイルです。CA を起動するたびに必ずこのファイルを読みにいきます。CA 構築時にこのファイルの Export Password を設定し、その Password を CA 起動時に必ず入力しなくてはなりません。この時、正しいパスワードならば動作を継続しますが、間違ったパスワードならばプログラムが終了します。

- ca.cer について

  Windows に合わせて上記のような拡張子をしていますが、中見は CA の証明書で X.509PEM の形式をしています。 PKCS#12 ではユーザが CA の証明書を参照することができないので、証明書のみ別に用意しています。

- ca.cai と各プロファイルについて

  ca.cai は CA が保持しているプロファイルのリストが含まれます。また、cert/*.cpi には各プロファイルの情報が含まれており、そのファイルは、

  - 現在のシリアルナンバ
  - 発行した証明書とその状態（期限切れや失効情報）
  - 証明書や CRL の有効期間設定
  - 証明書や CRL へ付加する拡張情報

  などを保持しています。この情報は aica コマンドのオプションで変更できます。

9 CA サーバに対して CA の登録が正常に終了すれば、CA 構築が完了します。設定がうまく行われたかどうかをチェックするために、CA サーバを起動してリモートアクセスを実行してみます。プロファイルの表示を行い、下記のように”SMIME user”プロファイル情報が表示されることを確認してください。

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

2.1.1. HSM 利用手順

1 HSM を使用する場合、鍵の管理方法は HSM 側に依存します。CA を構築する前に、HSM 製品のマニュアルに従い、HSM のセットアップと初期化を行なってください。

2 HSM の初期化を行なった後、aisetup.sh から HSM に対しアクセスを行ない、秘密鍵の作成を行ないます。この時、製品より提供される PKCS#11（Cryptoki）ライブラリとラベル名を指定します。

```
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 :   （パスワードを入力）
Verify - Input Master Passwd :   （パスワードを入力）
```

この時、CA のマスタパスワードを入力するように要求されますが、HSM によっては直接 HSM 側にキーパットでパスワードを入力するものもあり、その場合パスワードは無視されます。なお、CA で使用されない場合でも CAOperator 鍵へのアクセス用パスワードとして使用されますので、安全なパスワードを設定してください。

3 ここから先の手順は、新規 CA 構築（CA サーバ登録あり）と同様なため、2.1 項を参照してください。

2.2. 新規 CA 構築（CA サーバ登録なし）

1 ローカルファイルアクセス用の CA を構築する場合は、”ainewca.sh” シェルスクリプトを使います。コマンドの形式は次のとおりです。

```
ainewca.sh [オプション] CA 名
オプション:
    -keyalgo algo :  秘密鍵アルゴリズムを指定します(rsa(標準),dsa,ecdsa)
    -keysize size :  生成する鍵長を指定します(標準:1024bit)
    -days day     :  証明書の有効日数を指定します
    -start time   :  開始時間を指定します  "YYYYMMDDHHMMSSZ"
    -end time     :  終了時間を指定します  "YYYYMMDDHHMMSSZ"
    -p12 file     : PKCS#12 を指定して CA を構築します
```

また、CA の構築パターンの概要は次のとおりです。

証明書と秘密鍵を新規に作成する	-keysize オプションで鍵長を指定し、新規に秘密鍵･公開鍵の生成と自己署名証明書の発行を行います。この場合の CA は、最上位の CA（RootCA）となります。
PKCS#12 をインポートする	上位の CA から証明書と秘密鍵のファイル（PKCS#12）を配布され、そのファイルを利用して CA を構築します。この場合の CA は、下位 CA となります。

2 新規に証明書と秘密鍵を作成する場合、下記のようにコマンドを実行します。コマンドを実行すると、鍵の生成が始まります。

```
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 鍵の生成が終わると証明書のサブジェクトの入力を行います。

```
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 サブジェクトの入力が終了すると、CA 証明書の拡張情報の修正を行うか聞かれます。通常は”n”を指定します。認証局の運営規定にて CA 証明書のプロファイルが規定されている場合は、それに従って修正を行います。

```
do you modify CA certificate extension ? (y/n)[n]: y（←通常は n を指定）

        X509v3 extensions:
           x509 Basic Constraints:[critical]
              CA:TRUE
              PathLenConstraint:NULL
           x509 Key Usage:
              keyCertSign, cRLSign (0x06)
           x509 Authority Key Identifier:
              f5:f3:3a:c6:41:ed:b0:cb:69:06:ec:78:1d:49:a5:06:bb:9d:3e:96:
           x509 Subject Key Identifier:
              f5:f3:3a:c6:41:ed:b0:cb:69:06:ec:78:1d:49:a5:06:bb:9d:3e:96:
--
1. Basic Constraints          2. Key Usage              3. Extended Key Usage
4. Authority Key Identifier   5. Subject Key Identifier 6. Issuer Alt Name
7. Subject Alt Name           8. Certificate Policy     9. Policy Mapping
10. CRL Distribution Point    11. Authority Info Access 12. OCSP no check
13. Netscape CRL Url          14. Netscape Comment      15. Netscape Cert Type
0. Quit
select certificate extension number [0]: （←変更を加える拡張情報の番号を入力）
```

5 証明書のサブジェクトや有効期間の確認を行い、証明書を発行します。このとき、CA の証明書と秘密鍵を PKCS#12 ファイルへ保存するため、パスワードを入力します。このパスワードは CA 操作を行うのに必要なため、必ず覚えてください。

```
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   :  （パスワードを入力）
Verifying - Export PKCS#12   :  （パスワードの確認入力）
.....................................................................oo
.....oo
Update CA information.
import a file to the store successfully.
Succeeded to make new CA !!
```

6 コマンドの実行によりカレントディレクトリに、”CA 名” のディレクトリを作成しその中に ca.p12, ca.cer, ca.cai のファイルを作成します。ファイルの存在を確認できれば、CA の構築が完了します。

2.3. 新規 RA 構築(RA サーバ登録あり)

1 ユーザエンロール機能を有効にするには、RA の構築が必要です。RA を構築する場合は、”ainewra.sh” シェルスクリプトを使います。コマンドの形式は次のとおりです。

```
ainewra.sh [オプション] CA 名
オプション:
      -op id          : CA オペレータの証明書を指定します
      -pw pwd         : CA オペレータの秘密鍵の PIN を指定します
      -sv name        : CA のサーバ名を入力します
      -p7b file       : offline CA モードの場合、CA の PKCS7 ファイルを指定します
```

-op オプションを使う場合は、ローカルマシンの証明書ストアにオペレータ証明書をインポートしておく必要があります。手順は以下の通りです。

① 接続先の CA からオペレータ証明書を発行します。CA 構築時に作成したオペレータ証明書もしくは、aica user –addop にて新規に発行した証明書を使用します。

② aica export コマンドで PKCS#12 形式のファイルに出力します。

③ FTP などでローカルマシンにコピーして、aistore –i コマンドでオペレータ証明書をローカル証明書ストアにインポートします。

④ インポートした証明書の ID は aistore コマンドで確認できますので、その ID を-op で指定します。

この手順の詳細については、「3.11 リモートアクセス設定」を参照してください。

なお、-op, -sv はオプション扱いになっていますが、これらのオプションを省くと aica.cnf の RAd RegInfo セクションに空欄として出力されるため、コマンド使用時にはこれらのオプションをすべて指定する必要があります。

2 新規に RA サーバ(online CA)を構成する場合、下記のようにコマンドを実行します。

```
bash$ ainewra.sh -op CAOperator001 -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 :   (パスワードを入力)
Verify - Input Operator Passwd :   (パスワード確認入力)

RA initialization has been finished successfully.
```

接続先の CA 名称が、”testca”の場合は、ローカルの NAREGI CA インストールディレクトリ以下に”testca_ra”ディレクトリが作成され、ここに Web Enroll や airad の動作に必要なファイルが構成されます。

```
bash$ ainewra.sh –p7b ca.p7b testca
```

オフライン CA モードを構成する場合、下記のようにコマンドを実行します。
この場合も同様に、”testca_ra”ディレクトリが作成され、Web Enroll や airad の動作に必要なファイルが構成されます。その後、aica.cnf の該当する RAd RegInfo セクションの offline_ca_mode=true をセットしてください。
なお、1 つのサーバ上で最大 64 個までの RA を構築することが可能です。

3 Web エンロールを使用する場合は、httpd.conf に以下の行を加えます。この作業はテキストエディタを使用して、手動で行ってください。

```
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>
```

上記の例では、先に作成した RA である testca_ra に対して Web のエイリアスを作成しています。/testca_ra 以下に CGI スクリプトを、/testca_ra/img 以下に画像データを配置することで、ユーザは http://raname/testca_ra/aienroll（CGI）にアクセスして証明書エンロールサービスを利用することが可能になります。
Web エンロールの詳細については、「3.3 Web Enroll の設定」も参照してください。

2.4. PKCS#12 指定 CA 構築

1 上位の CA より証明書を発行してもらい、秘密鍵と証明書（PKCS#12）を使用して CA を構築します。この場合、certreq コマンドにより自身で鍵ペアを作成し、上位 CA に対して証明書を要求することができます。また、上位の CA より鍵ペアと証明書全てを配布してもらうこともできます。

2 PKCS# 12 ファイルを指定して、下記のようにコマンドを実行します。一度、パスワードの確認を行います。このとき、入力された証明書が CA 証明書として利用可能かどうかチェックします。

```
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:  （パスワードを入力）
CA PKCS#12 file open
......oo
.............oo
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 コマンドの実行によりカレントディレクトリに、<CA 名> のディレクトリを作成しその中に ca.p12, ca.cer, ca.cai のファイルを作成します。ファイルの存在を確認できれば、CA の構築が完了します。

2.5. CA 証明書の更新

1 同一の鍵ペアを使用して CA 証明書の拡張情報を変更する場合や、有効期限を変更する場合には、”aica renew”にて CA 証明書の更新操作が行えます。なお、鍵更新を行う場合は、新規に CA を構築してください。

```
aica renew [オプション]
オプション:
      -self          : ルート CA 証明書を更新します。(Default)
      -p10           : CA 鍵を使用して PKCS#10 を作成します。(Sub-CA 向け)
      -days day      : 証明書の有効日数を指定します
      -start time    : 開始時間を指定します "YYYYMMDDHHMMSSZ"
      -end time      : 終了時間を指定します "YYYYMMDDHHMMSSZ"
```

2 CA 証明書の有効期限を変更する場合は、以下のように実行します。

```
bash$ aica renew -start 20010101 -end 20080401
CA PKCS#12 file open
Input PKCS#12 Password:  （パスワードを入力）
do you modify CA certificate extension ? (y/n)[n]:（リターン）

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]:  （リターン）
now signing ..
Enter master CA password:  （パスワードを入力）
Verifying - Enter master CA password:（パスワードを入力）
```

3 CA 鍵ペアを使用して PKCS#10 ファイルを作成する場合は、以下のように実行します。

```
bash$ aica renew –p10
CA PKCS#12 file open
Input PKCS#12 Password:  （パスワードを入力）
save a ca pkcs10 file (ca.p10) ... done.
```

2.6. aica.cnf の設定

aica.cnf ファイルには、CA サーバや CRL Publisher、WEB Enroll の動作設定や、証明書ストア位置、デフォルトサブジェクトなどの設定が保存されています。テキストファイルなので、手動で変更することも可能であり、aiconftool などを使用してスクリプトから修正することも可能です。このファイルは通常、"NAREGI CA インストールディレクトリ/lib/aica.cnf"に配置されています。

- general info の設定

証明書の保存や検証に使用する証明書ストアのディレクトリ等を設定します。

```
[general info]
store_dir =/usr/local/naregi-ca/store
salt_val  =BsEWeI12dsfFUf14dWBs7EsfaUYs5
[general info end]
```

証明書ストアは通常、"NAREGI CA インストールディレクトリ/store/" ディレクトリに構成され、そのフルパスがこの store_dir に指定されます。

各フィールドの概要は次のとおりです。

store_dir	NAREGI CA が使用する証明書ストアディレクトリを指定します。254 文字まで有効です。
salt_val	パスワードなどを暗号化するときに使用する salt_val です。aica.cnf では、サーバの自動起動を行うためにパスワードの保存ができますが、その暗号化処理の鍵の一部として設定します。62 文字まで有効です。

なお設定文字列は、'='の後すぐから改行までが有効になります。また、ディレクトリの最後には'¥'を記入しないで下さい。また、文字列の先頭に '#' があるとその行はコメントとして解釈されます。

- CAd の設定

CA サーバの動作設定セクションです。

```
[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

errlog_rotate       =2048
isslog_rotate       =2048
acclog_rotate       =2048
[CAd end]
```

各フィールドの概要は次のとおりです。

capath.X	CA サーバがリモート操作可能な CA（ディレクトリ）を設定します。254 文字まで有効です。.X には数値が入り、最大 64 個までの CA が登録可能です。
caname.X	CA サーバがリモート操作可能な CA 名を設定します。32 文字まで有効です。.X には数値が入り、最大 64 個までの CA が登録可能です。
capwd.X	CA サーバの起動時には、それぞれの CA の起動パスワードを入力する必要があります。このフィールドを設定すると、CA サーバの自動起動が可能になります。
sv_id	CA サーバの SSL サーバ証明書を指定します。SSL サーバ証明書は、NAREGI CA ストアに保存されている必要があり、ストア ID を指定することで、SSL サーバ証明書の利用が可能になります。
sv_id_pwd	CA サーバの起動時には、SSL サーバ証明書をアクティブにするためにパスワード入力が必要です。このフィールドを設定すると、CA サーバの自動起動が可能になります。
sv_port	CA サーバのポート番号です。標準では 11411 を使用します。
sv_listen	CA サーバの同時接続数（Listen）です。
f_ssl_use	CA サーバ接続に SSL が必要か指定します。**true / false** にて指定します。
f_ssl_reqcert	CA サーバ接続に SSL クライアント認証が必要か指定します。**true / false** にて指定します。
f_ssl_novfycrl	CA サーバ接続で SSL クライアント認証を行ったときに、CRL が必要か指定します。**true / false** にて指定します。通常、CA サーバの利用者登録は別に行いますので、CRL を必要としない（true）を設定します。
ssl_timeout	CA サーバ接続のタイムアウトの設定を行います。標準では 600 秒が指定されます。
errlog	エラーログの出力ファイル名を指定します。254 文字まで有効です。

isslog	発行ログの出力ファイル名を指定します。254 文字まで有効です。
acclog	アクセスログの出力ファイル名を指定します。254 文字まで有効です。
errlog_rotate	エラーログのローテートサイズを指定します。0 を指定するとローテートを無効にします。
isslog_rotate	発行ログのローテートサイズを指定します。0 を指定するとローテートを無効にします。
acclog_rotate	アクセスログのローテートサイズを指定します。0 を指定するとローテートを無効にします。

- default CA の設定

aica コマンドの動作設定セクションです。

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

#cl_id      =
#cl_id_pwd          =

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

初期の設定では、カレントディレクトリの CA 情報を読みにいきます。もし、特定のディレクトリの CA 情報をもとに証明書を発行したい場合は、

ca_dir =/home/myname/myca

のように設定して下さい。また、特定のリモート CA へ接続するように設定したい場合は、

ca_dir =servername:caname

のように設定して下さい。

各フィールドの概要は次のとおりです。

ca_dir	aica コマンドがアクセスする CA ディレクトリを指定します。254 文字まで有効です。通常は、カレントディレクトリを示す”.”が指定されます。リモート CA を指定する場合は、”サーバ名:CA 名”を指定します。
ca_port	CA サーバのポート番号です。標準では 11411 を使用します。
ca_pwd	CA の起動パスワードを設定します。このフィールドを設定すると、CA の自動起動が可能になります。
cl_id	SSL クライアント証明書を指定します。SSL クライアント証明書は、NAREGI CA ストアに保存されている必要があり、ストア ID を指定することで、SSL クライアント証明書の利用が可能になります。

cl_id_pwd	SSL クライアント証明書をアクティブにするためにパスワード入力が必要です。このフィールドを設定すると、CA サーバへの自動接続が可能になります。
f_ssl_use	CA サーバ接続に SSL が必要か指定します。**true / false** にて指定します。
f_ssl_novfycrl	CA サーバ接続で SSL サーバ証明書のチェックを行うときに、CRL が必要か指定します。**true / false** にて指定します。

- CRL Publisher の設定

CRL Publisher の動作設定セクションです。

```
[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
upd_chk_interval    =60
f_exp_mode          =false

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_rotate       =2048
isslog_rotate       =2048
[CRL Publisher 0 end]
```

CRL Publisher の設定は、1 つのセクションで1つの aicrlpub が動作します。複数の Publisherを設定する場合は、[CRL Publisher X][CRL Publisher X end]の区切りをもつ複数のセクションを持つことで可能です。”X”には、特に上限値はありません。

CRL Publisher は指定したディレクトリまたは、LDAP サーバに対して ARL と CRL を出力します。特に、ローカルディレクトリに出力する場合は、out-ARL.crl, out-CRL.crl, out-CRL-All.crl の 3 つのファイルを出力します。

各フィールドの概要は次のとおりです。

ca_dir	CRL Publisher がアクセスする CA ディレクトリを指定します。254 文字まで有効です。リモート CA を指定する場合は、"サーバ名:CA 名" を指定します。
ca_port	CA サーバのポート番号です。標準では 11411 を使用します。
ca_pwd	CA の起動パスワードを設定します。このフィールドを設定すると、CA の自動起動が可能になります。
cl_id	SSL クライアント証明書を指定します。SSL クライアント証明書は、NAREGI CA ストアに保存されている必要があり、ストア ID を指定することで、SSL クライアント証明書の利用が可能になります。
cl_id_pwd	SSL クライアント証明書をアクティブにするためにパスワード入力が必要です。このフィールドを設定すると、CA サーバへの自動接続が可能になります。
f_ssl_use	CA サーバ接続に SSL が必要か指定します。**true / false** にて指定します。
f_ssl_novfycrl	CA サーバ接続で SSL サーバ証明書のチェックを行うときに、CRL が必要か指定します。**true / false** にて指定します。
interval	CRL Publisher のベース動作間隔(秒)を設定します。通常は、CRL の有効期限と同じ間隔を指定します。f_exp_mode で強制発行もしくはエクスポートモードで動作します。
upd_chk_interval	更新チェック発行を行う場合のチェック間隔(秒)を設定します。最新の CRL と CA のユーザリストを比較し、失効状態に変化があれば CRL を更新します。0 を指定した場合、更新チェックを行いません。
f_exp_mode	CRL の発行モードを指定します。この値を false にすると、CRL が有効期限内でも CRL の強制発行を行います。通常は、true を指定します。
tm_start	CRL Publisher の起動時間を設定します。CRL Publisher の動作間隔と CRL の有効期限が同一の場合、CRL の有効期限が切れた直後に CRL Publisher を起動するように設定できます。
out_dir	CRL の出力ディレクトリを指定します。254 文字まで有効です。
ldap_host	CRL の出力 LDAP サーバを指定します。このフィールドが設定されている場合、out_dir は無効になります。254 文字まで有効です。
ldap_port	LDAP サーバのポート番号です。標準では 389 を使用します。
ldap_base	CRL を出力するエントリを示す DN(Distinguished Name)です。CA のエントリ等を指定します。254 文字まで有効です。
ldap_bind	LDAP サーバへの接続ユーザ名です。シンプルバインドにて接続します。254 文字まで有効です。

ldap_pwd	LDAP サーバへの接続を行うときに使用するパスワードです。30 文字まで有効です。
ld_crl_attr	CRL を出力するエントリの属性名です。64 文字まで有効です。
ld_crl_prof	CRL を示すプロファイル名です。通常は"CRL-All"を指定します。
ld_arl_attr	ARL を出力するエントリの属性名です。64 文字まで有効です。
ld_arl_prof	ARL を示すプロファイル名です。通常は"ARL"を指定します。
errlog	エラーログの出力ファイル名を指定します。254 文字まで有効です。
isslog	発行ログの出力ファイル名を指定します。254 文字まで有効です。
errlog_rotate	エラーログのローテートサイズを指定します。0 を指定するとローテートを無効にします。
isslog_rotate	発行ログのローテートサイズを指定します。0 を指定するとローテートを無効にします。

- RAd の設定

RAd の動作設定セクションです。RAd の動作設定は、airad, Enroll CGI, aienroll の各モジュールによって共有されます。

```
[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_rotate       =2048
errlog_rotate       =2048
isslog_rotate       =2048
[RAd end]
```

RAd には、airad サーバの動作を設定する各項目と、airad, Enroll CGI, aienroll で共有するログファイル出力の設定を行います。

各フィールドの概要は次のとおりです。

sv_id	airad の SSL サーバ証明書を指定します。SSL サーバ証明書は、NAREGI CA ストアに保存されている必要があり、ストア ID を指定することで、SSL サーバ証明書の利用が可能になります。
sv_id_pwd	airad の起動時には、SSL サーバ証明書をアクティブにするためにパスワード入力が必要です。このフィールドを設定すると、CA サーバの自動起動が可能になります。
sv_port	airad のポート番号です。標準では 11412 を使用します。
sv_listen	airad の同時接続数（Listen）です。
f_ssl_use	airad 接続に SSL が必要か指定します。**true / false** にて指定します。
f_ssl_optreq	airad の SSL 接続で、クライアント認証をオプションとして指定します。**true / false** にて指定します。
f_ssl_reqcert	airad 接続に SSL クライアント認証が必須か否かを指定します。**true / false** にて指定します。**f_ssl_optreq** よりも優先します。
f_ssl_novfycrl	airad 接続で SSL クライアント認証を行ったときに、CRL が必要か指定します。**true / false** にて指定します。
ssl_timeout	airad 接続のタイムアウトの設定を行います。標準では 600 秒が指定されます。
errlog	エラーログの出力ファイル名を指定します。254 文字まで有効です。
isslog	発行ログの出力ファイル名を指定します。254 文字まで有効です。
acclog	アクセスログの出力ファイル名を指定します。254 文字まで有効です。
errlog_rotate	エラーログのローテートサイズを指定します。0 を指定するとローテートを無効にします。
isslog_rotate	発行ログのローテートサイズを指定します。0 を指定するとローテートを無効にします。
acclog_rotate	アクセスログのローテートサイズを指定します。0 を指定するとローテートを無効にします。

- RAd RegInfo の設定

RAd RegInfo の動作設定セクションです。RAd RegInfo の動作設定は、airad, Enroll CGI, aienroll の各モジュールによって共有されます。

```
[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
offline_ca_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_bind           =0
ldap_base           =c=JP

ldap_user_attr      =cn
ldap_license_attr   =uid
ldap_pin_attr       =userPassword

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

email_sbjfilter     =0

notice_update       =168

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

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

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

各フィールドの概要は次のとおりです。

raname	RA の名称を設定します。30 文字まで有効です。

rapath	RA の設定ファイルが保存されるディレクトリを絶対パスで指定します。254 文字まで有効です。
ca_dir	Enroll モジュールがアクセスするリモート CA を指定します。”サーバ名:CA 名”を指定し、254 文字まで有効です。
ca_port	CA サーバのポート番号です。標準では 11411 を使用します。
ca_uid	CA サーバに接続するユーザ名です。通常は、SSL クライアント認証を行うため、このフィールドは使用しません。
ca_pwd	CA サーバに接続するユーザパスワードです。通常は、SSL クライアント認証を行うため、このフィールドは使用しません。
cl_id	SSL クライアント証明書を指定します。SSL クライアント証明書は、NAREGI CA ストアに保存されている必要があり、ストア ID を指定することで、SSL クライアント証明書の利用が可能になります。
cl_id_pwd	SSL クライアント証明書をアクティブにするためにパスワード入力が必要です。このフィールドを設定すると、CA サーバへの自動接続が可能になります。
f_ssl_use	CA サーバ接続に SSL が必要か指定します。**true / false** にて指定します。
f_ssl_novfycrl	CA サーバ接続で SSL サーバ証明書のチェックを行うときに、CRL が必要か指定します。**true / false** にて指定します。
interval	aienroll の動作間隔を設定します。aienroll は定期的に CA サーバに接続し、キューに溜まっている CSR の処理具合をチェックします。
post_mode	証明書の申請を CA のキューに溜めて CA 運用者による確認発行を行うか指定します。true の場合は運用者確認発行、false の場合は即時発行となります。
offline_ca_mode	証明書の申請を RA のキューにファイルとして溜めて、RA 運用者による確認発行を行うか指定します。このモードの場合、手動で CA から証明書を発行し、証明書配置用コマンドを実行する必要があります。
authmode	証明書申請ユーザの認証を行うか指定します。整数値にて指定され、以下のモードが存在します。 0…匿名ユーザに証明書の発行を許可します。 1…ID/Password によるユーザ認証を行います。 2…License ID（One Time ライセンス）によるユーザ認証を行います。 4…LicenseID/ChallengePIN によるユーザ認証を行ないます。このモードでは LDAP サーバの指定が必須です。
wwwpwd	証明書の申請に ID/Password 認証が必要な場合に、passwd ファイルを指定します。254 文字まで有効です。Anonymous mode ではこの値は無視されます。ldap_host が指定されていた場合、この値は無視されます。
wwwlicense	証明書の申請に License ID 認証が必要な場合に、license ファイルを指定します。254 文字まで有効です。Anonymous mode ではこの値は無視されます。

wwwsessions	RA のセッション情報を格納するファイルを指定します。254 文字まで有効です。このファイルは通常、rapath のディレクトリに格納されます。
ldap_host	証明書の申請に ID/Password 認証が必要な場合に、LDAP サーバを指定します。254 文字まで有効です。Anonymous mode ではこの値は無視されます。
ldap_port	LDAP サーバのポート番号です。標準では 389 を使用します。
ldap_bind	LDAP BIND 認証方式の指定。 0…Simple Bind 1…SASL Cram MD5 2…SASL Digest MD5
ldap_base	LDAP サーバの検索ベースエントリを示す DN（Distinguished Name）です。254 文字まで有効です。
ldap_user_attr	ユーザ名の検索で使用する属性型を指定します。32 文字まで有効です。
ldap_license_attr	LicenseID の検索で使用する属性型を指定します。32 文字まで有効です。
ldap_pin_attr	Challenge PIN の検索で使用する属性型を指定します。32 文字まで有効です。
smtp_host	発行情報を通知するための SMTP サーバを指定します。254 文字まで有効です。
smtp_port	SMTP サーバのポート番号です。標準では 25 を使用します。
admin_email	CA 運用者のメールアドレスを指定します。126 文字まで有効です。Post mode で CSR を受け付けると、CA 運用者に通知が送付されます。
web_address	発行情報で通知される証明書取得用の URL を指定します。254 文字まで有効です。
email_sbjfilter	ユーザが入力した EMAIL を証明書に含めるかどうかを指定します。なお、入力したメールアドレスは LDAP もしくは sessions.0 に保存され更新通知に使用されます。 0…証明書サブジェクト DN に含める 1…証明書拡張情報 SubjectAltName に含める 2…証明書には含めない
notice_update	証明書の更新通知メールを送信します。証明書期限切れになる時間から、この項目で指定された時間（単位：hour）を引いた時刻になると、sessions.0 に含まれているユーザメールアドレスに対して更新通知を行ないます。
gridmap	グリッドミドルウェアである Globus に対応したグリッドマップファイルを出力します。254 文字まで有効です。ID/Password で証明書を発行した場合は、ID-Subject DN のマッピング、LicenseID の場合は licenseID-Subject DN のマッピングを出力します。
gridcertpath	グリッドミドルウェアである UNICORE 向けに発行した証明書をファイルとして出力するディレクトリを指定します。254 文字まで有効

	です。この値を有効にする場合は、必ず gridmap のファイルも指定してください。
groupname.X	グループ名を指定します。30 文字まで有効です。LDAP との連携などでは、ドメイン名を示すこともあります。
groupprof.X	グループに適用するプロファイル名を指定します。30 文字まで有効です。
grouphost.X	LDAP との連携を行う場合に、LDAP サーバを指定します。254 文字まで有効です。
groupbase.X	LDAP との連携を行う場合に、ユーザエントリのベース DN を指定します。254 文字まで有効です。
groupemail.X	グループの代表となるメールアドレスを指定します。126 文字まで有効です。authmode=4 のときに、RA オペレータ対して通知を行ないます。

- subject DN の設定

  新規 CA 構築や certreq コマンドで使用する、サブジェクト情報のデフォルト値を設定します。1つの属性型(C, O 等)に対して、1つの値が指定できます。

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

2.7. gridmap.cnf の設定

gridmap.cnf ファイルでは、gridmapgen の動作設定を行います。このファイルは通常、”NAREGI CA インストールディレクトリ/lib/gridmapgen.cnf”に配置されています。

- Grid MapGen の設定

gridmapgen の動作間隔やログの出力先などの設定を行います。

```
[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]
```

各項目の内容は以下の通りです。

interval	gridmapgen の動作間隔（秒）を設定します。
usermap	ローカルユーザとグローバルユーザ（ライセンス ID）のマッピングファイルを指定します。254 文字まで有効です。
gridmap	Globus 用の grid-mapfile の出力先を指定します。254 文字まで有効です。なお、このファイルは UNICORE UUDB の更新時にも使用するため、必ず出力ファイルの指定を行います。
uudb_bin	UNICORE UUDB のコマンドを含むディレクトリを指定します。254 文字まで有効です。必ず絶対パスで指定してください。なお、UUDB のコマンドを実行する場合、Java コマンドのインストールが必要です。
proxy_host	refmap.X にて、RA サーバ（WEB）からグローバルなマップファイルを取得する時に、プロクシが必要であればホスト名を指定します。126 文字まで有効です。

proxy_port	refmap.X にて、RA サーバ（WEB）からグローバルなマップファイルを取得する時に、プロクシが必要であればポート番号指定します。
refmap.X	グローバルなマップファイルを取得するための RA サーバ（WEB）を URL の形式で指定します。126 文字まで有効です。最大で 32 個までのサイトを指定できます。
refcert.X	UNICORE UUDB に対して、証明書を追加する場合に必要です。RA サーバ（WEB）上に証明書ファイルが配置されており、その上位のディレクトリまでを URL の形式で指定します。126 文字まで有効です。最大で 32 個までのサイトを指定できます。
errlog	エラーログの出力ファイル名を指定します。254 文字まで有効です。
acclog	アクセスログの出力ファイル名を指定します。254 文字まで有効です。
errlog_rotate	エラーログのローテートサイズを指定します。0 を指定するとローテートを無効にします。
acclog_rotate	アクセスログのローテートサイズを指定します。0 を指定するとローテートを無効にします。

3. CA・RA サーバの運用

NAREGI CA のサーバ起動について説明します。

3.1. CA サーバの起動と停止

- CA サーバ起動

  CA サーバを起動する場合、aicad コマンドを直接実行してください。aica.cnf の設定や CA 構築が正しく行われている場合は、下記のように CA サーバの起動を確認できます。

```
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=kunasiri.dbg.bs1.fc.nec.co.jp,
OU=server-c7f914-97aa76, CN=caserver0100,
set certificate request option.
start aicad daemon process (5398)
```

  aicad コマンドを実行すると、aica.cnf の”CAd”セクションの設定に従って、リモート操作可能な CA の登録を行います。この時、”capwd.X=”にパスワード文字列が指定してある場合は、自動的に CA の起動と登録が行われます。この文字列が空欄の場合は、起動時にその CA のマスタパスワードの入力が要求されます。

  また、標準の設定では CA サーバへの接続には SSL のクライアント認証を要求します。このため、CA サーバ側には SSL サーバ証明書が必要であり、NAREGI CA のインストール時に SSL サーバ証明書を自動的に生成しています。このサーバ証明書は、NAREGI CA の証明書ストアにインストールされ、既定のパスワードで暗号化されています。aica.cnf の sv_id には SSL サーバ証明書の ID が、sv_id_pwd にはパスワードが設定されており、CA 起動時には自動的に SSL サーバ証明書を取り込むようになっています。SSL サーバ証明書については、後述の「SSL サーバ証明書自動生成」の項目を参照してください。

  CA の登録が無事完了すると、デーモンプロセスが起動されます。デーモンプロセスのプロセス ID は上記の例では 5398 と表示されています。このプロセスにて、設定したポート（標準は 11411）への接続を待ちうけ、接続を accept すると子プロセスを起動し、子プロセスにて SSL Handshake とそれに続く CA 操作が実行されます。CA サーバは設

定したポートを占拠するため、2 つ以上のデーモンプロセスを立ち上げることはできません。

- CA サーバ停止

  CA サーバを停止する場合、kill コマンドにて直接プロセスを停止してください。

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

  ここでプロセスが停止するのは親プロセスのみとなります。もし、何らかの CA オペレーションが行われており、その操作を実行する子プロセスが存在する場合、子プロセスにはシグナルは送られず、操作が終了するまで子プロセスは生きています。

- 実行アクセス権

  CA サーバは、aicad を実行したユーザ権限を引き継いで実行されます。もし、aicad を一般ユーザにて実行した場合、aicad の実行ユーザが /Install_DIR/aica/lock/ や証明書ストア、CA ディレクトリにアクセス可能でなければなりません。
  また、aicad を root で実行することも可能ですが、一般的なネットワークセキュリティを考えた場合、好ましい方法とは言えません。CA サーバ用のユーザを作成し、そのユーザによる CA の構築や、CA サーバの起動が望まれます。

- リモート操作に対するアクセス制御

  CA に対しては、ローカルファイルアクセスとリモートアクセスの 2 通りが可能です。ローカルファイルアクセスを行う場合は、アクセス制御は特になく、パスワード入力が正しければ全ての CA 操作が行えます。リモートアクセスの場合は、ca.passwd に従ったユーザ認証とアクセス制御が行われます。

  通常、CA オペレータ証明書が発行され、オペレータはこれを使用して SSL クライアント認証で CA サーバに接続します。その後、LCMP プロトコルの操作系統によってアクセス制御が施されます。LCMP により行える操作とアクセス制御は以下の通りです。

  bindRequest ... 接続要求。常にアクセス ON の必要あり。
  signRequest ... (即時)発行要求
  listRequest ... 発行一覧取得要求
  profRequest ... プロファイル操作(一般情報、拡張情報)要求
  certRequest ... ユーザ証明書操作(更新、失効、出力、鍵保管、鍵出力)要求
  csrRequest ... CSR キュー操作(CSR 提出、CSR 承認・拒否)要求

crlRequest ... CRL 操作要求
servOpRequest ... サーバオペレーション要求
extendedReq ... 拡張操作要求

このため、CA の管理を行う場合は、CA サーバの起動や一般 CA オペレータ権限を指定できる CA のマスタオペレータと、上記のアクセス制御内で限られた操作が可能な、一般 CA オペレータに分けられます。(Web Enroll CGI といったモジュールもこの一般オペレータに含まれます)

- ログファイル出力

CA サーバは、aica.cnf の"CAd"セクションに設定されたログファイルを出力します。通常は、アクセスログ、発行ログ、エラーログの 3 種類を出力し、それぞれ cad_access.log, cad_issue.log, cad_error.log の名称で指定のディレクトリに出力されます。それぞれのログの出力内容は以下の通りです。

アクセスログ	CA サーバへの全ての操作要求とその結果をログに出力します。ログには、日時、セッション ID、接続元ホスト名、CA ユーザ、CA 名、操作内容が記述されます。
発行ログ	証明書の発行や更新、CRL の発行が行われた場合にログを出力します。ログには、日時、セッション ID、接続元ホスト名、CA ユーザ、CA 名、操作内容が記述されます。
エラーログ	CA サーバにて発生した、全ての操作エラーやシステムエラーをログに出力します。ログには、日時、セッション ID、接続元ホスト名、CA ユーザ、CA 名、操作内容の他、暗号ライブラリのエラー番号も表示されます。

また、ログファイルはファイルサイズでのローテートが可能であり、ローテートが行われると、cad_access.log.20031127111008 のようにローテート実行時の日付が付加されます。これらのファイルをテープ等に保管することをお勧めします。

3.2. CRL Publisher の起動と停止

- CRL Publisher コマンド概要

  コマンドの形式は次のとおりです。

  aicrlpub [オプション]
  オプション:
  -start **time** : 起動時間を"YYYYMMDDHHMMSSZ"に指定します
  (設定時間は GMT です)
  -end **time** : 終了時間を"YYYYMMDDHHMMSSZ"に指定します
  (設定時間は GMT です)
  -s **section** : aica.cnf の設定番号(標準 0).

  CRLPublisher は、1 プロセスで 1 セクションの設定をベースとして動作します。通常、1 セクションに 1 つの CA の CRL 出力設定が記述されるため、CRL Publisher を実行する場合は、登録されている全ての CA に対してプロセスの起動が必要になります。

- CRL Publisher 起動

  CRL Publisher を起動する場合、aicrlpub コマンドを直接実行してください。aica.cnf の設定や CA 構築が正しく行われている場合は、下記のように CRL Publisher の起動を確認できます。

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

  aicrlpub コマンドを実行すると、aica.cnf の”CRL Publisher X”セクションの設定に従って、CRL Publisher を起動します。この時、”ca_pwd.X=”にパスワード文字列が指定してある場合は、自動的に CRL Publisher の起動が行われます。この文字列が空欄の場合は、起動時にアクセス先 CA のマスタパスワードまたは、CA オペレータ秘密鍵のパスワード入力が要求されます。CRL Publisher は、1 つのセクションに対して1プロセスとなるため、複数のプロセスを起動することが可能です。

  CRL Publisher は定期的に CA にアクセスし、証明書状態をチェックして CRL の発行を行います。または、リモート CA にアクセスし、CRL の発行要求を行います。停止中は sleep にて待ち状態になり、CPU 負荷をかけることはありません。このため、CRL Publisher は外部からの入力を受け付けることはなく、停止用のシグナルを受け取るまで、または指定した終了時間まで動作しつづけます。

- CRL Publisher 停止

  CRL Publisher を停止する場合、kill コマンドにて直接プロセスを停止してください。

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

  CRL Publisher は子プロセスの生成などは一切行いませんが、複数の CRL Publisher を起動することは可能であるため、ps 等でプロセスを見ると aicrlpub が多数動作している状態もありえます。この場合は、それぞれの CRL Publisher を手動で kill してください。

- 実行アクセス権

  CRL Publisher は、aicrlpub を実行したユーザ権限を引き継いで実行されます。もし、aicrlpub を一般ユーザにて実行した場合、aicrlpub の実行ユーザが /Install_DIR /aica/lock/ や証明書ストア、CA ディレクトリにアクセス可能でなければなりません。
  また、aicrlpub を root で実行することも可能ですが、一般的なネットワークセキュリティを考えた場合、好ましい方法とは言えません。CA サーバ用のユーザを作成し、そのユーザによる CA の構築や、CRL Publisher の起動が望まれます。

- ログファイル出力

  CRL Publisher は、aica.cnf の"CRL Publisher X"セクションに設定されたログファイルを出力します。通常は、発行ログ、エラーログの 2 種類を出力し、それぞれ pub_issue.log, pub_error.log の名称で指定のディレクトリに出力されます。それぞれのログの出力内容は以下の通りです。

発行ログ	CRL の発行が行われた場合にログを出力します。ログには、日時、セッション ID、CA 名、出力先 CRL が記述されます。
エラーログ	CA Publisher で発生した、全ての操作エラーやシステムエラーをログに出力します。ログには、日時、セッション ID、CA 名、エラー内容の他、暗号ライブラリのエラー番号も表示されます。

  また、ログファイルはファイルサイズでのローテートが可能であり、ローテートが行われると、pub_access.log.20031127111008 のようにローテート実行時の日付が付加されます。これらのファイルをテープ等に保管することをお勧めします。

3.3. Web Enroll の設定

- Web Enroll 概要

  NAREGI CA には、Web サーバ経由でユーザ証明書の発行や失効を行う、証明書ライフサイクル管理用の Web Enroll 機能があります。Web Enroll を実現するために、NAREGI CA は複数のモジュールが用意されています。1 つは aienroll.cgi で、通常、このモジュールは rapath/cgi/aienroll としてインストールされます。Apache などの Web サーバからこの CGI を呼び出すことで、ユーザは証明書の発行や失効、更新といった操作が行えます。

  また、ChallengePIN/LicenseID 認証モード（authmode=4）で使用される RA オペレーション CGI モジュール提供されています。提供される CGI は以下の通りです。

CGI 名	URL	役割
airaadmin.cgi	https://～/CA 名_ra/airaadmin	RA アドミニストレータ向けの管理画面。RA オペレータのリスト表示や申請一覧、各種操作を行う。
airaregist.cgi	https://～/CA 名_ra/airaregist	RA オペレータ向けの申請画面を提供。
airaenroll.cgi	https://～/CA 名_ra/airaenroll	RA オペレータ向けの証明書発行画面を提供。
airaop.cgi	https://～/CA 名_ra/airaop	RA オペレータ向けの管理画面。ユーザのリスト表示や申請一覧、各種操作を行う。
airegist.cgi	https://～/CA 名_ra/airegist	ユーザ向けの申請画面を提供。
aienroll.cgi	https://～/CA 名_ra/aienroll	ユーザ向けの証明書発行画面を提供。他の認証モードでも証明書発行用に使用。

  このモードによる運用方法については、別途マニュアルが用意されているので詳細はそちらをご覧ください。

  もう 1 つのモジュールとして、POST モードやオフライン CA モード、更新通知を行うためのエンロールチェック（bin/aienroll）が用意されており、これは CA サーバに定期的にアクセスし、失効状態のチェックやユーザへの各種通知を行います。エンロールチェックの詳細については「更新通知の起動と停止」をご参照ください。

  Web Enroll には、3 つの証明書発行モードと 5 つの認証レベルが用意されています。これらの発行モードと認証レベルを組み合わせることで、柔軟な証明書ライフサイクル管理が行えます。

  証明書発行モードは以下の通りです。

即時発行	ユーザは Web Enroll CGI にアクセスし、ユーザ情報を入力します。入力された情報を元に証明書要求（CSR）を作成し、CGI に POST します。即時発行モードでは、CGI は CA に発行申請を行い、証明書を即時に発行しユーザマシンにインストールできます。

運用者確認発行 post_mode=true	ユーザは Web Enroll CGI にアクセスし、ユーザ情報を入力します。入力された情報を元に証明書要求（CSR）を作成し、CGI に POST します。運用者確認発行では、CGI は CSR を CA の CSR キューに POST し、AcceptID を受け取りいったん処理を終了します。 CA 運用者は、CSR キューにある証明書要求に対し、証明書発行許可または拒否の操作を行います。操作結果は、aienroll によりチェックされ、ユーザに email が送信されます。 証明書が発行された場合は、証明書取得 URL が email に記載されていますので、ユーザは Web Enroll CGI にアクセスし証明書をユーザマシンにインストールします。
オフライン CA 発行 offline_ca_mode =true	ユーザは Web Enroll CGI にアクセスし、ユーザ情報を入力します。入力された情報を元に証明書要求（CSR）を作成し、CGI に POST します。オフライン CA 発行では、CSR を RA サーバ上に保管し、運用者が CA に手動で CSR をコピーして証明書を発行します。 証明書発行後、運用者が RA サーバ上に証明書を配置することで、aienroll によりチェックされ、ユーザに email が送信されます。 証明書が発行された場合は、証明書取得 URL が email に記載されていますので、ユーザは Web Enroll CGI にアクセスし証明書をユーザマシンにインストールします。

認証レベルは以下の通りです。

匿名認証 (authmode=0)	ユーザに対して認証を行わずに証明書を発行します。この認証レベルでは、証明書は即時に発行されます。また、更新や失効操作は行えません。
ID/Password 認証 (authmode=1)	証明書を発行する場合、ユーザは ID とパスワードの入力が必要です。この認証レベルでは、即時発行、運用者確認発行の 2 つのモードが利用できます。また、更新や失効操作は ID/Password 認証または SSL クライアント認証によって行えます。 この認証レベルを設定する場合、testca_ra/en.passwd といったローカルファイルに ID/Password を設定する方法と、LDAP サーバなど外部の認証サービスと連携する方法があります。
License ID 認証 (One Time ライセンス方式) (authmode=2)	証明書を発行する場合、ユーザは License ID の入力が必要です。1 つの License ID は 1 枚の証明書発行にのみ使用できます。この認証レベルでは、即時発行、運用者確認発行の 2 つのモードが利用できます。また、更新や失効操作は SSL クライアント認証によって行えます。 この認証レベルを設定する場合、testca_ra/en.license といったローカルファイルに License ID のリストを設定します。
SSL クライアント認証 (authmode=1/2)	この認証レベルは証明書が発行された後にのみ有効です。この認証が行われた場合、使用したクライアント証明書の更新や失効操作が可能になります。なお、失効操作を行った場合はクライアント証明書が利用できなくなるため、再度証明書発行からやり直す必要があります。
Challenge PIN / License ID 認証	LicenseID 認証では、ユーザからの証明書申請を対面で受け取り、ライセンス ID を直接ユーザに通知する必要がありました。

(UPKI モード) **(authmode=4)**	このモードでは、ユーザは WEB で証明書を申請し（ここで ChallengePIN を設定）、inetOrgPerson の各属性値を含む申請情報を LDAP サーバに保管できます。その後、RA オペレータによって証明書の発行が許可され、ユーザは WEB もしくはコマンドラインで ChallengePIN を入力して証明書を取得します。LicenseID は動的に生成・削除され、いわゆるセッション情報として扱われます。 証明書の更新や失効にもChallengePINが使用され、authmode=1/2 で使用される SSL クライアント認証による更新・失効は行いません。 このモードでは、RA のロールが定義されています。 ・ RA アドミニストレータ … CA オペレータにより任命され、新規の RA オペレータの審査・証明書を行う。 ・ RA オペレータ … RA アドミニストレータにより任命され、ユーザの審査・証明書発行を行う。 このモードでは、基本的に即時発行モードを利用してください。また、必ず LDAP サーバとメールサーバの設定を行ってください。

- Web Enroll 設定

  Web Enroll を使用するためには、CGI の設定が必要です。Apache 向けのサンプル設定ファイルが lib/httpd.conf に用意されています。Apache で Enroll CGI を利用する場合は、この内容を参考に httpd.conf を書き換えてください。

```
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>
```

  Apache 1.3 系では、上記の内容を<IfModule mod_alias.c>セクション内に記述します。

（上記の例では、NAREGI CA は/usr/local/naregi-ca にインストールされています）また、Apache2 系では<IfModule mod_alias.c>がありませんので、他の ScriptAlias が設定されている付近に記述します。

testca_ra ディレクトリには、テンプレート HTML ファイルや、en.passwd、セッション管理ファイル（sessions.0）などの設定ファイルが配置されます。これらのデータに対してアクセスされないように、testca_ra/cgi/img と testca_ra/cgi のみをアクセス可能にします。CGI モジュールである aienroll.cgi は testca_ra/cgi/aienroll として配置されます。Web Enroll CGI にアクセスする場合は、http://localhost/testca_ra/aienroll の URL に対してアクセスします。Web Enroll の設定により、証明書申請画面または認証画面が表示されます。

- xenroll.dll 配置

  Web Enroll CGI を使用するためには、クライアント側の Internet Explorer で秘密鍵の生成が行えなければなりません。この時、証明書エンロール用の ActiveX コンポーネントである xenroll.dll を使用します。

  xenroll.dll を使用する場合は、最新のバージョンでなければなりません。クライアント側の Internet Explorer のバージョンや Windows のバージョンにより含まれている xenroll.dll のバージョンが異なるため、NAREGI CA の Web Enroll CGI で鍵の生成が行えないことがあります。これを防ぐために、xenroll.dll のバージョンが違う場合にユーザ側で動的にダウンロードできるよう ratemplate/components/ディレクトリに最新の xenroll.dll を配置してください。

  なお、最新の xenroll.dll は下記のサイトよりパッチとして提供されています。

  http://support.microsoft.com/default.aspx?scid=kb;ja;323172

  xenroll.dll はマイクロソフト社の製品に含まれるため、Windows ライセンスを保持し、なおかつ、ユーザへのパッチ配布として ratemplate/components/に DLL を配置してください。もしくは、ユーザ側にて MS02-04 のパッチを適用してもらい、クライアントマシンの xenroll.dll の更新を行った上で、Web Enroll を利用することも可能です。

- mod_ssl 設定

  Web Enroll にて SSL クライアント認証を行う場合は、mod_ssl の設定が必要です。mod_ssl の設定を行う場合、SSL サーバ証明書の発行や、CA 証明書の配置、httpd.conf の設定が必要です。以下の手順で行ってください。

1 Web サーバ用の SSL クライアント証明書を発行します。CA の構築は既に行われているものとします。証明書の発行は「CA の運用」章に従って行います。newcert.cer と newkey.key が作成されるのでこのファイルを SSL サーバ証明書として使用します。

```
bash$ cd myca
bash$ certreq –size 1024
   …(省略)…
bash$ aica sign newreq.p10
   …(省略)…
```

2 秘密鍵の newkey.key は暗号化されています。SSL サーバ起動時にパスワード入力を行いたくない場合は、certconv を使用して暗号化しない状態にできます。（または、1 の実行時に certreq –noenc オプションで鍵を生成します）

```
bash$ certconv -key newkey.key -pem -noenc newkey.key
Input PASS Phrase:（パスワードを入力）
++Check One PKCS#12Bag, type=11002
please input password for output key.
output a file ..
```

3 SSL サーバ証明書、SSL サーバ秘密鍵、CA 証明書を所定のディレクトリに配置します。CA 証明書は、SSL サーバ証明書を発行した CA のものでなければなりません。また、CA 証明書を配置したあとは、ssl のディレクトリで make コマンドを実行し、mod_ssl に対して CA を登録します。

```
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
…(省略)…
```

4 テキストエディタで httpd.conf を開き、mod_ssl の設定を行います。<IfDefine SSL>セクションの中の下記の項目を有効にします。

```
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
```

なお、SSLVerifyClient には”optional”の設定が適しています。証明書発行前には有効

な証明書を持っていないため、SSL クライアント認証は行えませんが、証明書発行後では SSL クライアント認証が利用できるためです。また、License ID 認証であれば、証明書発行前は http でアクセスし、証明書発行後に https のクライアント認証とすることも可能です。

5 設定が完了したら、httpd デーモンを起動してください。

```
bash$ httpd -DSSL
```

3.4. Web Enroll の起動と停止

- Web Enroll 起動

  Web Enroll CGI は Web サーバより実行されるため、とくに手動で起動する必要はありません。また、エンロールチェックは「運用者確認発行」や「オフライン CA」モードを有効にした場合や、更新通知を利用したい場合にのみ起動します。

- Web Enroll 停止

  Web Enroll CGI は常駐サービスではないため、とくに手動で停止する必要はありません。また、エンロールチェックは「運用者確認発行」や「オフライン CA」モードを有効にした場合や、更新通知を利用したい場合にのみ起動するため、これ以外では停止操作は必要はありません。

- 実行アクセス権

  Web Enroll CGI は、httpd から起動されるため nobody ユーザ権限を持っています。このため nobody ユーザに対して、/Install_DIR/aica/lock は rw 権限や証明書ストアには r 権限が必要です。また、"testca_ra" 以下に対する r 権限と testca_ra/cgi/aienroll に対する rx 権限を持たなければなりません。

- ログファイル出力

  Web Enroll は aica.cnf の"RAd"セクションに設定されたログファイルを出力します。通常は、アクセスログ、発行ログ、エラーログの 3 種類を出力し、それぞれ enroll_access.log, enroll_issue.log, enroll_error.log の名称で指定のディレクトリに出力されます。それぞれのログの出力内容は以下の通りです。

アクセスログ	Web サーバ経由の全ての操作要求とその結果をログに出力します。主に Web Enroll CGI により出力されます。ログには、日時、CA ユーザ、CA 名、操作内容が記述されます。
発行ログ	エンロールチェックにより確認した証明書発行記録や失効記録をログに出力します。ログには、日時、CA ユーザ、CA 名、操作内容が記述されます。
エラーログ	Web Enroll CGI またはエンロールチェックにより発生した操作エラーをログに出力します。ログには、日時、CA ユーザ、CA 名、操作エラーの他、暗号ライブラリのエラー番号も表示されます。

  また、ログファイルはファイルサイズでのローテートが可能であり、ローテートが行われると、enroll_access.log.20031127111008 のようにローテート実行時の日付が付加されます。これらのファイルをテープ等に保管することをお勧めします。

3.5. RA サーバの起動と停止

- RA サーバ起動

  RA サーバを起動する場合、airad コマンドを直接実行してください。aica.cnf の設定や RA 構築が正しく行われている場合は、下記のように RA サーバの起動を確認できます。

```
bash$ airad
starting RA server ... read config file.
port=11412,listen=5
ssl=1,req=48,vfy=9
read server certificate : O=yume.dbg.bs1.fc.nec.co.jp,
OU=server-69dc86-4d8bea, CN=caserver0100,
set certificate request option (mode=48)
start airad daemon process (23293)
```

  airad コマンドを実行すると、aica.cnf の”RAd”セクションの設定に従って、リモート操作可能な RA の登録を行います。RA は構築時にパスワード入力を行い、常に自動起動するようになっていますが、入力したパスワードに間違いがあると、Web Enroll などで証明書が発行できなくなります。この場合は、aiconftool を使って、パスワードを設定してください。

  certreq といったリモートコマンドから証明書を申請する場合、RA サーバへの接続に SSL が使用されます。このため、RA サーバ側には SSL サーバ証明書が必要であり、NAREGI CA のインストール時に SSL サーバ証明書を自動的に生成しています。CA と RA を同一のサーバで運用する場合は、SSL サーバ証明書を共有しますが、別サーバの場合は、それぞれ別のサーバ CA、サーバ証明書を持ちます。このサーバ証明書は、NAREGI CA の証明書ストアにインストールされ、既定のパスワードで暗号化されています。aica.cnf の”RAd”セクションの sv_id には SSL サーバ証明書の ID が、sv_id_pwd にはパスワードが設定されており、RA 起動時には自動的に SSL サーバ証明書を取り込むようになっています。SSL サーバ証明書については、後述の「SSL サーバ証明書自動生成」の項目を参照してください。

  RA の登録が無事完了すると、デーモンプロセスが起動されます。デーモンプロセスのプロセス ID は上記の例では 23293 と表示されています。このプロセスにて、設定したポート（標準は 11412）への接続を待ちうけ、接続を accept すると子プロセスを起動し、子プロセスにて SSL Handshake とそれに続く RA 操作が実行されます。RA サーバは設定したポートを占拠するため、2 つ以上のデーモンプロセスを立ち上げることはできません。

- RA サーバ停止

  RA サーバを停止する場合、kill コマンドにて直接プロセスを停止してください。

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

  ここでプロセスが停止するのは親プロセスのみとなります。もし、何らかの RA オペレーションが行われており、その操作を実行する子プロセスが存在する場合、子プロセスにはシグナルは送られず、操作が終了するまで子プロセスは生きています。

- 実行アクセス権

  RA サーバは、airad を実行したユーザ権限を引き継いで実行されます。もし、airad を一般ユーザにて実行した場合、airad の実行ユーザが /Install_DIR/aica/lock/ や証明書ストア、RA ディレクトリにアクセス可能でなければなりません。
  また、airad を root で実行することも可能ですが、一般的なネットワークセキュリティを考えた場合、好ましい方法とは言えません。RA サーバ用（もしくは CA サーバ用と同一の）ユーザを作成し、そのユーザによる RA の構築や、RA サーバの起動が望まれます。

- RA のユーザ認証

  RA サーバの認証や動作モードは、Web Enroll と共通です。「3.3 Web Enroll の設定」を参照してください。

- ログファイル出力

  RA サーバのログ出力は、Web Enroll と共通です。「3.3 Web Enroll の設定」を参照してください。

3.6. 更新通知の起動と停止

- エンロールチェックコマンド概要

  エンロールチェック（bin/aienroll）によりPOSTチェックや更新通知メールの送信が行われます。また、オフライン CA モード時には必ず起動します。コマンドの形式は次のとおりです。

```
aienroll [オプション]
オプション:
      -s section      : aica.cnf [RAd RegInfo]の設定番号(標準  0).
```

- エンロールチェックの起動

  エンロールチェックの起動を行う場合、aienroll コマンドを直接実行してください。aica.cnf の設定や CA 構築が正しく行われている場合は、下記のようにエンロールチェックの起動を確認できます。

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

  aienroll コマンドを実行すると、aica.cnf の”RAd RegInfo X”セクションの設定に従って、エンロールチェックを起動します。この時、”ca_pwd.X=”にパスワード文字列が指定してある場合は、自動的にエンロールチェックの起動が行われます。この文字列が空欄の場合は、起動時にアクセス先 CA のマスタパスワードまたは、CA オペレータ秘密鍵のパスワード入力が要求されます。エンロールチェックは、1 つのセクションに対して1プロセスとなるため、複数のプロセスを起動することが可能です。

- エンロールチェックの停止

  エンロールチェックを停止する場合、kill コマンドにて直接プロセスを停止してください。

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

  エンロールチェックは子プロセスの生成などは一切行いませんが、複数のエンロールチェックを起動することは可能であるため、ps 等でプロセスを見ると aienroll が多数動作している状態もありえます。この場合は、それぞれのエンロールチェックを手動で kill してください。

- 実行アクセス権

  エンロールチェックは、aienroll を実行したユーザ権限を引き継いで実行されます。もし、aienroll を一般ユーザにて実行した場合、aienroll の実行ユーザが /Install_DIR/aica/

lock/ や証明書ストア、”testca_ra” 以下にアクセス可能でなければなりません。
また、aienroll を root で実行することも可能ですが、一般的なネットワークセキュリティを考えた場合、好ましい方法とは言えません。CA サーバ用のユーザを作成し、そのユーザによる CA 構築や、エンロールチェック起動が望まれます。

- ログファイル出力
  エンロールチェックのログ出力は、Web Enroll と共通です。「3.3 Web Enroll の設定」を参照してください。

3.7. XKMS サービスの設定

- **XKMS 概要**

  XKMS（Xml Key Management Specification）は W3C により定義され、2005 年現在で Ver2.0 の仕様が公開されています。XML をベースとする Web サービスにおいて、PKI に関連する各種の操作を行います。

  XKMS では、W3C と IETF により定義された XML Signature と XML Encryption を参照しており、鍵や証明書の検索機能を提供する X-KISS（Xml Key Information Service Specification）と証明書のレジストレーション機能を提供する X-KRSS（Xml Key Registration Service Specification）が定義されています。

  X-KISS は証明書情報に関する操作を定義しており、以下の 2 つのサービスを提供します。

  - Locate Service ： 証明書の検索機能
  - Validate Service ： 証明書の検証機能

  X-KRSS は証明書レジストレーションに関する操作を定義しており、以下の 4 つのサービスを提供します。

  - Register Service ： 証明書発行機能
  - Reissue Service ：（同一鍵）証明書更新機能
  - Revoke Service ： 証明書失効機能
  - Recover Service ： ユーザ鍵のリカバリ機能

  NAREGI XKMS サービスと XKMS Java API では、Register サービスと Revoke サービスを提供しており、同一鍵による証明書更新やユーザ鍵のリカバリ機能は提供しておりません。

  また、サポートしている認証レベルは以下の通りです。

匿名認証	ユーザに対して認証を行わずに証明書を発行します。この認証レベルでは、証明書は即時に発行されます。失効操作を行う場合、証明書を指定することで、即時に失効されます。
License ID 認証 (One Time ライセンス方式)	証明書を発行する場合、ユーザは License ID の入力が必要です。1 つの License ID は 1 枚の証明書発行にのみ使用できます。この認証レベルの時は失効操作は行えません。 この認証レベルを設定する場合、testca_ra/en.license といったローカルファイルに License ID のリストを設定します。

- **動作環境**

  XKMS サービスは Java 1.4 以降で動作するように開発されています。サービスが使用するいくつかの PKI 関連のクラス、すなわち X509Certificate や X500Principal などのクラスは Java1.3 以前では使用することができないため、Java 1.4 以降が必須になります。

  Web サービスのエンジンとして、Apache Tomcat＋Axis を使用します。Axis と XKMS クラスライブラリを適切に配置することで、XKMS サービスが使用できます。

対応機種	JavaVM が動作する機種
CPU	JavaVM が対応している CPU
対応 OS	任意
メモリ	32MB 以上推奨
Java バージョン	1.4 以降
Tomcat バージョン	5.0, 5.5
Axis バージョン	1.2
その他、必要な Java パッケージ	Java LCMP API 1.1, Apache xml-security-1.2.1, JavaMail API 1.3.3

この他、XKMS サービスを動かすために、必要な Java パッケージが存在します。

- Java LCMP API 1.1
  Java LCMP API は NAREGI CA のパッケージに含まれている jlcmp.jar を使用します。
- Apache XML Security 1.2.1
  http://xml.apache.org/security/
  Apache XML Security は最新のバージョンを上記サイトよりダウンロードして使用します。
- JavaMail API 1.3.3
  http://java.sun.com/products/javamail/
  XKMS サービスは CA 運用者に証明書発行通知のメールを送信します。メール送信には JavaMail API を使用します。

- **Apache Axis 1.2 のセットアップ**
  Windows 環境に Apache Axis 1.2 をセットアップする場合の例を挙げます。

  1 JDK 1.4 をインストールします。OS の環境変数を変更します。
  ex. set JAVA_HOME= d:¥j2sdk1.4.1_02

  2 Apache Tomcat 5.5 をインストールします。インストール後、コントロールパネルの「サービス」から Apache Tomcat を起動します。起動後に http://localhost:8080/ へアクセスし、動作確認を行います。

  3 Apache Axis 1.2 をインストールします。OS の環境変数を変更します。

```
set AXIS_HOME=d:¥axis-1_2
set AXIS_LIB=%AXIS_HOME%¥lib
set
AXISCLASSPATH=%AXIS_LIB%¥axis.jar;%AXIS_LIB%¥commons-discov
ery.jar;  %AXIS_LIB%¥commons-logging.jar;%AXIS_LIB%¥jaxrpc.jar;%AXI
S_LIB%¥saaj.jar;%AXIS_LIB%¥log4j-1.2.8.jar;%AXIS_LIB%¥xml-apis.jar;
%AXIS_LIB%¥xercesImpl.jar
set CLASSPATH=.;%AXISCLASSPATH%;%CLASSPATH%
```

  4 axis-1_2¥webapps¥axis を Tomcat5.5¥webapps¥axis にコピーします。

  5 「サービス」にて Apache Tomcat を再起動します。
  再起動後に http://localhost:8080/axis/へアクセスし、Web サービスが正しくセットアップされているか動作検証します。http://localhost:8080/axis/happyaxis.jsp にてコンポーネントをチェックできます。なお、この状態ではオプショナル･コンポーネントは含まれていません。

- **Axis 用オプショナル･コンポーネントのセットアップ**
  オプショナル･コンポーネントとして、Apache XML Security と JavaMail ライブラリをセットアップします。

  1 Apache XML Security 1.2 を前述のサイトよりダウンロードします。ダウンロード後、ファイルを解凍し、libs に含まれる Jar ファイルを Axis のディレクトリにコピーします。コピー先：Tomcat5.5¥webapps¥axis¥WEB-INF¥lib¥*.jar

2 JavaMail API 1.3.3 を前述のサイトよりダウンロードします。ダウンロード後、ファイルを解凍し、mail.jar ファイルを Axis のディレクトリにコピーします。
コピー先：Tomcat5.5¥webapps¥axis¥WEB-INF¥lib¥mail.jar

3 「サービス」にて Apache Tomcat を再起動します。
再起動後に http://localhost:8080/axis/へアクセスし、Web サービスが正しくセットアップされているか動作検証します。http://localhost:8080/axis/happyaxis.jsp にてコンポーネントをチェックできます。オプショナル･コンポーネントも認識されていることを確認します。

- **XKMS サービスのセットアップ**
  XKMS サービスに必要なファイルを NAREGI CA ディレクトリからコピーし、サービスをデプロイします。

1 Java LCMP API の Jar ファイルを naregi-ca¥jlcmp¥lib¥jlcmp.jar から Axis のディレクトリにコピーします。
コピー先：Tomcat5.5¥webapps¥axis¥WEB-INF¥lib¥jlcmp.jar

2 XKMS サービスの Jar ファイルを naregi-ca¥xkms¥lib¥xkms-naregi.jar から Axis のディレクトリにコピーします。
コピー先：Tomcat5.5¥webapps¥axis¥WEB-INF¥lib¥xkms-naregi.jar

3 サービスをデプロイします。Apache Tomcat サービスが起動中であることを確認し、naregi-ca¥xkms ディレクトリに移動し、以下のコマンドを実行します。

```
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.
ファイル deploy.wsdd の処理中 / [en]-(Processing file deploy.wsdd)
<Admin>処理を実行しました / [en]-(Done processing)</Admin>
```

4 デプロイの処理が行われます。http://localhost:8080/axis/へアクセスし、Web サービスのリストを確認します。http://localhost:8080/axis/servlet/AxisServlet にてリストを表示し、XKMSService が登録されていることを確認します。

- **XKMS 利用環境の構築**

  XKMS サービスを実際に利用する場合、RA をセットアップ後にオペレータ証明書と CA 証明書ストアファイルの配置、ライセンス ID ファイルの配置、xkms-naregi.properties ファイルの修正と配置が必要です。

  以下に、それぞれのファイルの設定方法を記載します。

1. CA サーバに接続するためにオペレータ証明書が必要です。オペレータ証明書をファイルとして出力し、RA のディレクトリ（caname_ra とする）に配置します。

```
Naregi> aistore -id CAOperator001 -ef pk12 -e caop.p12
Access Private Key:　（パスワードを入力）
Input Export Password:　（パスワードを入力）
Verifying - Input Export Password:　（パスワードを入力）
export a store data successfully.
```

   この後、caop.p12 ファイルを caname_ra¥caop.p12 に配置します。

2. XKMS Validation サービス向けに、信頼する CA ストアを作成します。RA ディレクトリに移動し、以下のように ca.store ファイルを配置します。

```
Naregi> cd caname_ra
Naregi> keytool -import -file ca.cer -alias cacert -trustcacerts -keystore ca.store
キーストアのパスワードを入力してください:　abcdef
所有者: OU=testca unit, O=testca, C=JP
実行者: OU=testca unit, O=testca, C=JP
シリアル番号: 1
有効日: Mon Sep 12 05:01:24 GMT 2005　有効期限: Thu Sep 11 05:01:24 GMT 2008
証明書のフィンガープリント:
        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
この証明書を信頼しますか? [no]:　yes
証明書がキーストアに追加されました。
```

3. naregi-ca¥xkms¥xkms-naregi.properties ファイルを開き、環境に合わせてカスタマイズを行います。

```
### 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
```

各項目の内容は以下の通りです。(org.naregi.xkms.* の部分のみ)

caServer	XKMS サービスがアクセスするリモート CA のサーバ名を指定します。
caPort	CA サーバのポート番号です。標準では 11411 を使用します。
caName	XKMS サービスがアクセスするリモート CA の CA 名を指定します。
certProfile	証明書発行にて使用するプロファイル名を指定します。
opCertP12	オペレータ証明書の PKCS#12 ファイル名を指定します。CA サーバへの接続には SSL クライアント認証が必要になります。
opCertP12Pwd	オペレータ証明書の PKCS#12 ファイルにアクセスするためのパスワードを指定します。
trustCAStore	XKMS Validate サービスにて信頼する CA 証明書ストアファイルを指定します。
trustCAStorePwd	XKMS Validate サービスにて信頼する CA 証明書ストアファイルのアクセスパスワードを指定します。
authmode	証明書申請ユーザの認証を行うか指定します。整数値にて指定され、以下のモードが存在します。

	0…匿名ユーザに証明書の発行を許可します。 その他…License ID（One Time ライセンス）によるユーザ認証を行います。 なお、証明書失効は匿名認証モードでのみ行えます。
license	証明書の申請にLicense ID認証が必要な場合に、licenseファイルを指定します。Anonymous modeではこの値は無視されます。
ldapServer	証明書の申請にLicense ID認証が必要な場合に、LDAPサーバを指定します。Anonymous modeではこの値は無視されます。
ldapBase	LDAP サーバの検索ベースエントリを示す DN（Distinguished Name）です。
ldapAdmin	証明書発行時に、エントリ上の LicenseID の削除や証明書の出力を行うため、書き込み権限のあるユーザを指定します。
ldapAdminPwd	エントリに書き込み権限のあるユーザのパスワードを指定します。
ldapLicenseAttr	LicenseID の検索で使用する属性型を指定します。
smtpServer	発行情報を通知するための SMTP サーバを指定します。
systemEmail	XKMS システムのメールアドレスです。SMTP サーバにてメール発信可能な EMAIL アドレスである必要があります。
adminEmail	CA 運用者のメールアドレスを指定します。証明書発行申請を受付けるとこのメールアドレスに通知されます。
acceptCsr2	メール通知の本文を保持するファイルを指定します。
acceptCsr2sbj	メール通知のサブジェクトを指定します。
emailCharset	メール通知のキャラセットを指定します。

4 naregi-ca¥xkms¥xkms-naregi.properties ファイルを編集し終えたら、Tomcat の conf ディレクトリにコピーします。
コピー先：Tomcat5.5¥conf¥xkms-naregi.properties

5 設定が完了したら、XKMSRegisterSample.java をコンパイルし、XKMS サービスにアクセスします。証明書が発行できれば設定は完了です。

3.8. XKMS サービスの起動と停止

- XKMS サービス起動
  XKMS サービスを起動する場合、Apache Tomcat を起動します。それぞれのバージョン、OS 別の起動手順に従って、Tomcat を起動してください。

- XKMS サービス停止
  XKMS サービスを停止する場合、Apache Tomcat を停止します。それぞれのバージョン、OS 別の起動手順に従って、Tomcat を停止してください。

- 実行アクセス権
  XKMS サーバは、Tomcat のユーザ権限を引き継いで実行されます。このため Tomcat ユーザに対して、”caname_ra” 以下に対する読み取り権限と caname_ra/en.license に対する rw 権限を持たなければなりません。
  また、CA オペレータ証明書や信頼する CA ストアファイル、xkms-naregi.properties ファイルなど、重要な情報を保持しているファイルは、Tomcat の実行ユーザのみ読み込み権限を保持する形が望まれます。

- XKMS サービスのユーザ認証
  XKMS サービスのユーザ認証は、前項の「XKMS 概要」を参照してください。

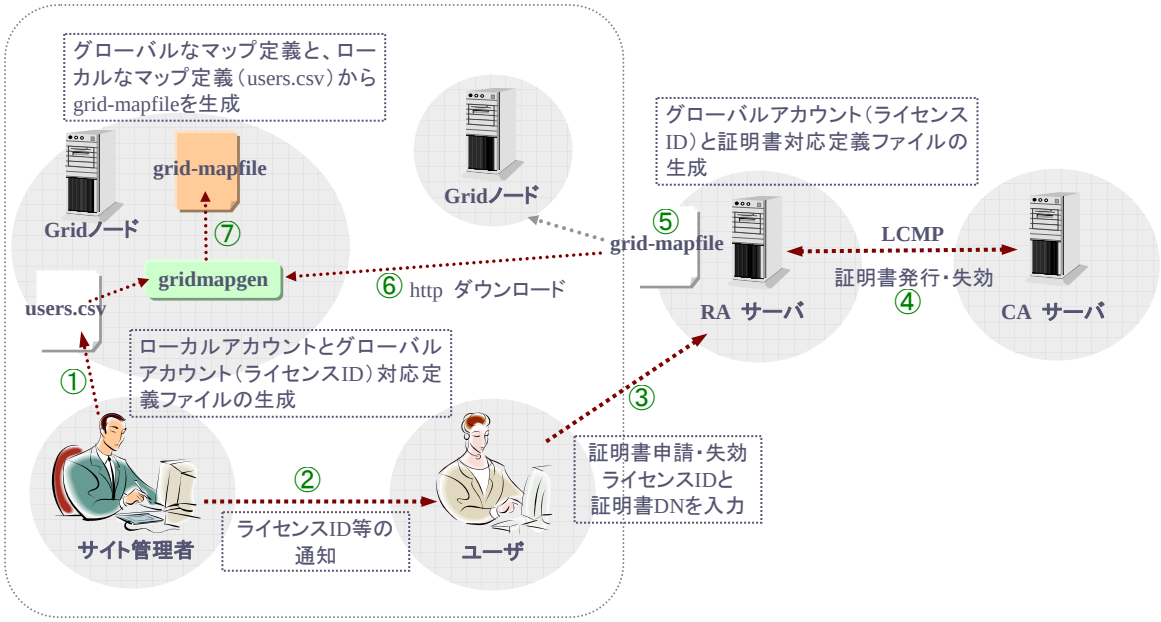
- ログファイル出力
  XKMS のログ出力は、Tomcat の logs¥stdout_*.log にて出力されます。例外の発生や動作エラーが起きた場合は、このログを参照してください。
  ファイル名 : Tomcat5.5¥logs¥stdout_*.log

3.9. グリッド連係機能の設定

- グリッド連係機能概要

  NAREGI CA には、Grid ミドルウェアである Globus もしくは UNICORE と連携する機能があり、gridmapgen プロセスを使用することで、ユーザ証明書とローカルアカウントのマッピングを効率よく行えるようになります。

  具体的には、RA サイト構築の際に、grid-mapfile（RA サーバ）を特定のディレクトリに出力するようにします（RAd RegInfo の gridmap を有効にする）。この grid-mapfile（RA サーバ）の作成は aienroll プロセスにより行われます。出力されたファイルは、httpd により外部に公開され、各サイトのグリッドノード上で動作している gridmapgen プロセスにより収集されます。グローバルな grid-mapfile（RA サーバ）を取得後、ローカルマシンに配置されている users.csv ファイルを読み込み、証明書のサブジェクトとローカルなアカウントのマッピングを行い、最終的に Globus で使用される grid-mapfile（ローカル）をローカルマシンに配置します。



  UNICORE を使用する場合は、RA サーバ上に証明書ファイル（PEM）を出力するようにします（RAd RegInfo の gridcertpath を有効にする）。即時発行の場合は airad もしくは aienroll.cgi により、管理者確認発行では aienroll により証明書ファイル（PEM）が出力されます。gridmapgen プロセスはローカルユーザの追加もしくは削除があった場合、UUDB/bin/add もしくは delete コマンドを使用してユーザ証明書の追加や削除を実行します。

  なお、グリッドマップファイル連携を行う場合は、ID/Password 認証もしくは LicenseID 認証が必ず必要になります。

※制限事項
UUDB/bin/delete コマンドの仕様では、1 つのローカルアカウントに対して複数の証明書を対応させると、削除時に複数のサブジェクトがリストされ、削除対象を選択するようになっています。gridmapgen プロセスからこのコマンドを呼び出す場合は、こうした選択操作を行えないため、1つのローカルアカウントに対して 1 つの証明書を対応させる必要があります。

3.10. グリッド連係機能の起動と停止

- RA サーバ起動
  グリッド連携機能を使用する場合は、CA / RA / Web サーバが全て設定済みであり、正常に動作することが前提となります。この上で、aicad を起動し、airad、aienroll を起動します。また、grid-mapfile と発行した証明書ファイル（PEM）を WEB サイトにより公開するため httpd の起動も行います。

- RA サーバ停止
  グリッド連携機能を停止する場合は、RA / Web サーバを手動で停止します。Kill コマンドを使用してプロセスを停止してください。

- RA サーバ実行アクセス権
  RA サーバは、airad / aienroll を実行したユーザ権限を引き継いで実行されます。もし、airad / aienroll を一般ユーザにて実行した場合、airad/ aienroll の実行ユーザが証明書ストア、RA ディレクトリにアクセス可能でなければなりません。
  /RA ディレクトリ/grid/certs/ ディレクトリについては、エンロール CGI と airad 双方により書き込みが行われます。httpd の起動アカウントと airad 等の起動アカウントのグループを統一し、CGI と airad 双方からアクセスできるような設定が必要です。
  また、airad を root で実行することも可能ですが、一般的なネットワークセキュリティを考えた場合、好ましい方法とは言えません。RA サーバ用（もしくは CA サーバ用と同一の）ユーザを作成し、そのユーザによる RA の構築や、RA サーバの起動が望まれます。

- gridmapgen 起動
  gridmapgen は、各グリッドノードにて起動します。プロセスを起動する場合は、RA / Web サーバが起動していることを確認してください。指定の URL より grid-mapfile の取得が行えない場合は、動作エラーとなりエラーログを出力します。

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

gridmapgen コマンドを実行すると、gridmap.cnf の” Grid MapGen”セクションの設定に従って、グリッドマップ連携モジュールを起動します。Globus 向けであれば、ローカルユーザを定義する usermap と、grid-mapfile の出力先、参照先 URL（refmap）の設定があれば grid-mapfile の更新を行います。UNICORE 向けであれば、これに加えて UUDB コマンドパスと、参照先 URL（refcert）の設定があれば UUDB の更新を行います。

- gridmapgen 停止

  gridmapgen を停止する場合、kill コマンドにて直接プロセスを停止してください。

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

- gridmapgen 実行アクセス権

  gridmapgen は、grid-mapfile もしくは UUDB の更新を行います。これらのファイルへの書き込み権限のあるユーザにて実行してください。

  また、gridmapgen を root で実行することも可能ですが、一般的なネットワークセキュリティを考えた場合、好ましい方法とは言えません。grid-mapfile 更新用のユーザを作成し、そのユーザによるプロセスの起動が望まれます。

- gridmapgen ログファイル出力

  gridmapgen は gridmap.cnf の” Grid MapGen”セクションに設定されたログファイルを出力します。通常は、アクセスログ、エラーログの 2 種類を出力し、それぞれ map_access.log, map_error.log の名称で指定のディレクトリに出力されます。それぞれのログの出力内容は以下の通りです。

アクセスログ	gridmapgen の動作状況をログに出力します。ログには、日時、操作内容が記述されます。
エラーログ	gridmapgen の動作エラーをログに出力します。ログには、日時、エラー内容が記述されます。

また、ログファイルはファイルサイズでのローテートが可能であり、ローテートが行われると、map_access.log.20031127111008 のようにローテート実行時の日付が付加されます。これらのファイルをテープ等に保管することをお勧めします。

3.11. リモートアクセスの設定

CA サーバにアクセスする場合、SSL クライアント認証が行われるため、RA サーバや CA オペレータマシン側に SSL サーバの CA 証明書と、CA オペレータ証明書＋秘密鍵が必要です。ここでは、これらのファイルの取得と配置手順を解説します。

1 SSL サーバ証明書は NAREGI CA のインストール時に自動的に生成されます。自動生成については、次節の「SSL サーバ証明書自動生成」を参照してください。SSL サーバ証明書の CA 証明書は証明書ストアの”root”に登録されています。

```
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
```

上記の例では、証明書 ID の”server-a9bec5-d95c8d”が、SSL サーバ用の CA 証明書です。O=”hostname”, OU=”server-******-******”のサブジェクトを持つ証明書が CA 証明書を表しています。

2 証明書 ID を指定して CA 証明書を取り出します。

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

3 同様に CA オペレータ証明書も証明書ストアの”my”に登録されています。証明書 ID をチェックし、PKCS#12 の形式で取り出します。

```
bash$ aistore -id CAOperator001 -ef pk12 -e caop.p12
Access Private Key:  （パスワードを入力）
Input Export Password:  （パスワードを入力）
Verifying - Input Export Password:  （パスワードを入力）
export a store data successfully.
```

4 取り出した caop.p12 と sslca.cer ファイルを、CA リモートアクセスを行うために、別の管理用マシンに移動します。

5 それぞれのファイルを証明書ストアにインストールします。それぞれ、自動的にmyストアや root ストアに配置されます。証明書 ID は自動的に付加されますが、オプションで明示的に指定することも可能です。

```
myhost % aistore -i sslca.cer
import a file to the store successfully.
myhost % aistore -i caop.p12
Input PKCS#12 Password:  （パスワードを入力）
Save Access Password :  （パスワードを入力）
Verifying - Save Access Password :  （パスワードを入力）
import a file to the store successfully.
```

6 CA サーバへのアクセスを行い、証明書ストアへの配置が正しく行われたか確認します。

```
myhost % aica print -sv caserv:testca -ssl -clid CAOperator001
tring to connect caserv(11411):testca (ssl)
Open Private Key:  （パスワードを入力）
-----------------------------------
Certificate Profile : SMIME user
   certificate version   : 3
   current serial number: 1
   signature algorithm   : md5WithRSAEncryption
   …(省略)…
```

3.12. SSL サーバ証明書自動生成

NAREGI CA では、インストール時に initsslcert.sh シェルスクリプトを実行しています。このシェルスクリプトは、CA サーバが使用する SSL サーバ用の CA の構築と、SSL サーバ証明書の発行を行っています。

SSL サーバ用の CA は、”NAREGI CA インストールディレクトリ/serv-ssl/”に配置されており、aica コマンドにより操作が可能です。この CA のパスワードは、”1234567890sslca”となっており、サーバ証明書は”SSL server”プロファイルに作成されます。CA 証明書と SSL サーバ証明書は共にインストール時点から 10 年の有効期限を持っています。

自動生成される証明書の有効期限は無限ではないため、証明書の更新または新規に SSL サーバ証明書の発行が必要になります。CA サーバは、自動生成された CA だけではなく、別の CA から発行された SSL サーバ証明書も利用可能であるため、証明書の更新時期が近づいた場合は新たに SSL 用の CA を構築し、SSL サーバ証明書の発行を行ってください。

3.13. CA・RA データバックアップ

NAREGI CA でバックアップ対象となるいくつかの重要なデータを説明します。基本的には、各 CA ディレクトリと各 RA ディレクトリ、NAREGI CA インストールディレクトリがバックアップされていれば、これをそのままリストアすることで、バックアップ時点のデータに復元できます。個々のデータ項目については以下を参照してください。

- CA データ
  NAREGI CA では 1 つのマシン上にいくつもの CA を構成することができます。それぞれの CA は、CA ディレクトリ以下にデータが構成されており、aica コマンドによって操作できます。個々の CA ディレクトリには、CA の秘密鍵やプロファイル管理情報、ユーザ証明書、ユーザ秘密鍵などが含まれており、下記のようなファイルで構成されます。
  CA_Name/
  CA_Name/ca.cer ... CA の証明書
  CA_Name/ca.p12 ... CA の証明書＋秘密鍵
  CA_Name/ca.cai ... CA 管理情報（プロファイルリスト等）
  CA_Name/ca.passwd ... CA のオペレータ情報（認証・アクセス権）
  CA_Name/ca.group ... CA のオペレータグループ情報

CA_Name/cert/ProfileName.cpi ... 証明書プロファイル情報
CA_Name/cert/ProfileName.cts ... ユーザ証明書
CA_Name/cert/ProfileName.kys ... ユーザ秘密鍵
CA_Name/cert/CRL.lpi ... CRL プロファイル情報
CA_Name/req/csr.rpi ... CSR キュー情報

CA のバックアップは、上記のディレクトリを保存することで行えます。リストアはデータをそのまま展開することで CA 情報を復元することができます。

- RA データ
  NAREGI CA では 1 つのマシン上にいくつもの RA を構成することができます。それぞれの RA は、RA ディレクトリ以下にデータが構成されており、Enroll CGI や airad, aienroll などのプロセスにより利用されます。個々の RA ディレクトリには、ユーザセッションファイル（sessions.0）の他、パスワードファイル（en.passwd）、ライセンスファイル（en.license）が配置されます。エンロール機能を使用する場合は、必ずそれぞれの RA ディレクトリのバックアップが必要です。

- 証明書ストア
  証明書ストアは SSL サーバ証明書の取得や、SSL クライアント認証時の検証に使用するなど、使用頻度の高いデータの 1 つです。通常、このデータは”NAREGI CA インストールディレクトリ/store/”に配置されます。

- aica.cnf
  CA サーバや RA サーバ、CRL Publisher などの動作設定ファイルです。NAREGI CA のほぼ全てのコマンドで参照する必要があり非常に使用頻度の高いデータの 1 つです。通常、このデータは”NAREGI CA インストールディレクトリ/lib/aica.cnf”に配置されます。

- ログファイル
  CA サーバや RA サーバ、CRL Publisher のログファイルです。必要であればログを保存します。通常、このデータは”NAREGI CA インストールディレクトリ/logs/”に配置されます。

4. CA の運用

NAREGI CA コマンド群による CA の運用方法に関して説明します。

4.1. 証明書要求作成

証明書を発行する場合、次の手順で操作を行います。

① certreq コマンドで、鍵ペアの生成と証明書要求（CSR）の作成を行います。

② aica コマンドで証明書要求に署名を行い、証明書を発行します。

この節では、certreq コマンドの使用法について解説を行います。

1 新しく鍵ペアの生成と証明書要求の作成を行う場合、certreq を使用します。
コマンドの形式は次のとおりです。

```
certreq [オプション]
オプション:
    -req file      : “file”名で証明書要求ファイルを保存します
    -key file      : “file”名で秘密鍵ファイルを保存します
    -algo alg      : 生成する鍵ペアのアルゴリズムです。
                     rsa,dsa,ecdsa を指定できます（標準:rsa）
    -size num      : 鍵ペアの鍵長を指定します（標準:512bit）
    -p passwd      : 秘密鍵の保存用パスワードを指定します
    -noenc         : 秘密鍵を暗号化せずに保存します
    -der           : 証明書要求の保存形式を DER にします
    -alt           : Subject Alt Name を指定します
```

なお、各アルゴリズムの概要は次のとおりです。

RSA 暗号	標準的に使用される公開鍵暗号です。暗号化と電子署名の双方に使用できます。一般的に安全な証明書を発行する場合は、鍵長に 1024bit を指定します。
DSA 暗号	電子署名のみ使用できる公開鍵暗号です。一般的に安全な証明書を発行する場合は 1024bit を指定します。
ECDSA 暗号	電子署名のみ使用できる公開鍵暗号です。楕円曲線暗号と呼ばれ、強度の安全性を提供します。一般的に安全な証明書を発行する場合 192bit を指定します。
MD5（ハッシュ関数）	一方向関数と呼ばれデータのダイジェスト（指紋）を取得できます。
SHA1（ハッシュ関数）	標準的に使用されるハッシュ関数です。一方向関数と呼ばれデータのダイジェスト（指紋）を取得できます。

2 512bit の鍵ペアを生成し、証明書要求を作成する場合は、以下のようにコマンドを実行してください。

```
bash$ certreq
generate private key (size 512 bit)
ooooo
...ooooo
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
   : (省略)
Input PASS Phrase:   (パスワードを入力)
Verifying - Input PASS Phrase:   (パスワードを入力)
```

上記のような流れにそって、秘密鍵の生成、証明書のサブジェクト入力、秘密鍵のパスフレーズの入力を行います。この操作で、newreq.p10 newkey.key が作成されます。ファイル形式は X.509 DER 形式、すなわち PKCS#10 と同様なものを作成します。

サブジェクトの概要については「2.1 新規 CA 構築」を参照

3 鍵長を変更したり、任意の出力ファイル名を指定する場合は、以下のようにコマンドを実行してください。

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

鍵ペアのアルゴリズムに DSA や ECDSA を指定した場合は、鍵生成に必要なパラメータの生成から行います。特に ECDSA のパラメータ生成は時間がかかりますので、ご注意ください。

4.2. 証明書要求への発行

証明書要求に対して証明書を発行する場合、次の手順で操作を行います。

1 aica sign オペレーションにより証明書要求に対して証明書を発行することができます。aica コマンドは、aica [operation] [options] の形式になっており、各オペレーションを指定することで、証明書や CRL の発行を行うことができます。なお、aica sign オペレーションの形式は次のとおりです。

```
aica sign [オプション] file
オプション:
      -sn num        : シリアルナンバを指定します
      -pf name       : 証明書プロファイルを指定します
                       （標準:SMIME user）
      -o file        : “file”名で証明書ファイルを保存します
一般オプション:
      -sv path       : “サーバ名:CA 名”を指定します
      -ssl           : リモート CA 接続に SSL を使用します
      -clid name     : SSL クライアント証明書の ID を指定します
      -u loginid     : CA ユーザ名を指定します（非 SSL 接続）
      -p passwd      : CA ユーザパスワードを指定します（非 SSL 接続）
```

2 証明書の発行は下記のとおりです。

```
bash$aica sign -o a.cer newreq.p10
CA PKCS#12 file open
Input PKCS#12 Password :  （パスワードを入力）

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.
```

aica コマンドを実行すると、最初に ca.p12（CA の証明書と秘密鍵ファイル）を開きます。ここで、パスワードの入力を間違えると、操作は中断され、処理が終了します。正し

いパスワードを入力し、読み込んだ証明書要求の署名が正しかった場合は、発行する証明書の情報を表示し、署名するか否かを問い合わせます。
ここで、'y'かリターンキーを入力すれば、証明書の発行を行います。標準では、newcert.cer を出力しますが、-o オプションにてファイル名を指定可能です。また入力ファイルする証明書要求（PKCS#10）は、PEM または DER 形式のどちらの形式でも構いません。

3 発行する証明書のシリアルナンバを指定することも可能です。この場合、下記のようにコマンドを入力します。

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

上記の例では、発行する証明書に 4023 番のシリアルナンバを付加します。すでにその番号を使っていた場合は、エラーの後プログラムが終了します。

4.3. 証明書要求登録と確認発行

証明書要求(CSR)を CA の CSR キューに溜め込み、キューの中にある CSR から証明書を発行することができます。リモートマシンからの証明書要求に対して即時に発行せず、一度 CA 運用者が確認した上で証明書の発行操作が可能です。

1 aica csr オペレーションにより証明書要求の POST、発行、または発行拒否を行います。コマンドの形式は次のとおりです。

```
aica csr [オプション]
オプション:
      -post file      : CSR ファイルをキューに登録します
      -sn num         : シリアルナンバを指定します
      -issue num      : CSR AcceptID を指定します
      -reject num     : CSR AcceptID を指定します
      -pf name        : 証明書プロファイルを指定します
                        (標準:SMIME user)
      -o file         : "file"名で証明書ファイルを保存します
```

2 csr -post オペレーションにより、証明書要求を CSR キューに登録できます。登録に成功すると、AcceptID が返されます。

```
bash$aica csr –post newreq.p10
CA PKCS#12 file open
Input PKCS#12 Password :  (パスワードを入力)
success to post a CSR (acceptID=2).
```

3 csr -issue オペレーションにより、CSR キューに登録されている証明書要求から証明書を発行します。

```
bash$aica csr –issue 2
CA PKCS#12 file open
Input PKCS#12 Password :  (パスワードを入力)
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. 証明書の更新

証明書のシリアルナンバや公開鍵をそのままにして証明書の有効期限や拡張情報のみ変更することができます。

1 aica resign オペレーションにより証明書の更新を行います。コマンドの形式は次のとおりです。

```
aica resign [オプション] file
オプション:
      -o file          : “file”名で証明書ファイルを保存します
一般オプション:
      -sv path         : “サーバ名:CA 名”を指定します
      -ssl             : リモート CA 接続に SSL を使用します
      -clid name       : SSL クライアント証明書の ID を指定します
      -u loginid       : CA ユーザ名を指定します（非 SSL 接続）
      -p passwd        : CA ユーザパスワードを指定します（非 SSL 接続）
```

2 resign オペレーションの後に証明書ファイルを指定すれば再署名を行うことが可能です。ただし、その証明書が失効されていた場合は、再署名できません。

```
bash$ aica resign -o out.cer in.cer
CA PKCS#12 file open
Input PKCS#12 Password :  （パスワードを入力）

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. 証明書の一括発行

指定したフォーマットに従ったCSVファイルを使用することで、証明書の一括発行を行うことができます。

1 aica csv オペレーションにより証明書の一括発行を行います。コマンドの形式は次のとおりです。

```
aica csv file
一般オプション:
    -sv path      : “サーバ名:CA 名”を指定します
    -ssl          : リモート CA 接続に SSL を使用します
    -clid name    : SSL クライアント証明書の ID を指定します
    -u loginid    : CA ユーザ名を指定します（非 SSL 接続）
    -p passwd     : CA ユーザパスワードを指定します（非 SSL 接続）
```

2 証明書の一括発行は下記のとおりに行います。

```
bash$ aica csv sample.csv
CA PKCS#12 file open
Input PKCS#12 Password :  （パスワードを入力）

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

上記の例のように、指定したCSVファイルに従って証明書を一括発行します。なお、証明書の作成過程において生成される各種ファイルは、./cert ./req ./key ./p12 以下にファイルとして残されます。

CSV フォーマットについて

CSV ファイルのフォーマットは次のとおりです。

形式（一行分）：

"プロファイル名",シリアルナンバ,"サブジェクト","SubjectAltName",
暗号,鍵サイズ,P12 作成フラグ,"p12 and key パスワード"

CSV 例 ：

"SMIME user",10010,"C=JP/OU=sample ou/CN=name10",

"email:name10@localhost",rsa,1024,1,"abcde"

注意事項 :

- 各々の項目は ',' で区切りスペースを入れないで下さい。
- プロファイル名とサブジェクト、パスワードはダブルクォーテーション "" で挟んで下さい。
- サブジェクトのタグ C,O,OU,… 等は必ず大文字にして下さい。
- サブジェクトのタグの先頭以外は必ず'/'で区切り、スペースを入れないで下さい。
- サブジェクトの項目は必ず'='の後にスペースなしで入力して下さい。
- 暗号は rsa, dsa, ecdsa を入力して下さい。
- 鍵サイズは RSA で 512 ～ 2048bit が推奨されます。
- P12 作成フラグは PKCS#12 ファイルを、0 が未作成、1 が作成です。
- パスワードは秘密鍵と p12 ファイルで同様のものを使います。

なお、既にシリアルナンバやサブジェクトが使用されていた場合は、エラーを表示後、次の処理へ移行します。

4.6. 証明書の失効

証明書を発行したユーザが秘密鍵を紛失したり、鍵の漏洩が起きた場合などは、証明書の失効を行う必要があります。

1 aica revoke オペレーションにより証明書の失効を行います。コマンドの形式は次のとおりです。

```
aica revoke num
一般オプション:
    -sv path      : “サーバ名:CA 名”を指定します
    -ssl          : リモート CA 接続に SSL を使用します
    -clid name    : SSL クライアント証明書の ID を指定します
    -u loginid    : CA ユーザ名を指定します（非 SSL 接続）
    -p passwd     : CA ユーザパスワードを指定します（非 SSL 接続）
```

2 証明書の失効は下記のとおりに行います。

```
bash$ aica revoke 3
CA PKCS#12 file open
Input PKCS#12 Password :  （パスワードを入力）
------
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]: （該当番号を入力）
success to revoke a certificate (sn:3)
```

証明書の失効を行う場合は、シリアル番号を指定します。指定した番号が存在した場合、証明書状態に失効のマークをつけます。もし指定した番号が存在しない場合は、その操作を無視します。

4.7. 証明書の失効解除

既に失効済みの証明書の状態から正常（失効を取り消した）の状態にします。

1 aica unrevoke オペレーションにより証明書の失効解除を行います。コマンドの形式は次のとおりです。

```
aica unrevoke num
一般オプション:
    -sv path      : “サーバ名:CA 名”を指定します
    -ssl          : リモート CA 接続に SSL を使用します
    -clid name    : SSL クライアント証明書の ID を指定します
    -u loginid    : CA ユーザ名を指定します（非 SSL 接続）
    -p passwd     : CA ユーザパスワードを指定します（非 SSL 接続）
```

2 証明書の失効解除は下記のとおりに行います。

```
bash$ aica unrevoke 3
CA PKCS#12 file open
Input PKCS#12 Password :  （パスワードを入力）
success to unrevoke a certificate (sn:3)
```

証明書の失効解除を行う場合は、シリアル番号を指定します。指定した番号が存在し証明書が失効されていた場合、証明書状態の失効マークを取り消します。もし指定した番号が存在しない場合は、その操作を無視します。

4.8. CRL の発行

証明書失効リスト（CRL: CertificateRevocationList）を発行します。CRL を発行することで、ユーザ証明書や署名の検証時に破棄状態のチェックを行えます。

1 aica crl オペレーションにより CRL の発行します。コマンドの形式は次のとおりです。

```
aica crl
一般オプション:
    -sv path      : “サーバ名:CA 名”を指定します
    -ssl          : リモート CA 接続に SSL を使用します
    -clid name    : SSL クライアント証明書の ID を指定します
    -u loginid    : CA ユーザ名を指定します（非 SSL 接続）
    -p passwd     : CA ユーザパスワードを指定します（非 SSL 接続）
```

2 CRL の発行は下記のとおりに行います。

```
bash$ aica crl
CA PKCS#12 file open
Input PKCS#12 Password :   （パスワードを入力）

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

CRL 発行を行うと、カレントディレクトリに 3 つの CRL ファイルが作成されます。それぞれの CRL は PEM 形式で保存されており、これらファイルには次のような違いがあります。

- out-CRL.crl
  ユーザ証明書の失効情報（シリアル番号と失効日時のリスト）を含んでいます。

- out-ARL.crl
  下位 CA 証明書の失効情報（シリアル番号と失効日時のリスト）を含んでいます。この CRL は特別に ARL（AuthorityRevocationList）と呼ばれます。

- out-CRL-All.crl
  CRL と ARL の双方の失効情報（シリアル番号と失効日時のリスト）を含んでいます。CRL を WEB 上に公開する場合などは一般的にこのファイルが使用されます。

4.9. 証明書と秘密鍵のエクスポート

CA には証明書と秘密鍵が保管されており、シリアル番号を指定することでユーザ証明書またはユーザ秘密鍵のエクスポートが行えます。また、CA の CSR キューに保管されている証明書要求のエクスポートも行えます。

1 aica export オペレーションにより証明書または秘密鍵をエクスポートします。コマンドの形式は次のとおりです。

```
aica export [オプション]
オプション:
      -sn num        : シリアル番号を指定します
      -o file        : “file”名で証明書ファイルを保存します
      -cert          : ユーザ証明書をエクスポートします（標準）
      -key           : ユーザ秘密鍵をエクスポートします
      -p12           : PKCS#12（秘密鍵＋証明書）をエクスポートします
      -csr num       : エクスポートする CSR の AcceptID 番号を指定します
一般オプション:
      -sv path       : “サーバ名:CA 名”を指定します
      -ssl           : リモート CA 接続に SSL を使用します
      -clid name     : SSL クライアント証明書の ID を指定します
      -u loginid     : CA ユーザ名を指定します（非 SSL 接続）
      -p passwd      : CA ユーザパスワードを指定します（非 SSL 接続）
```

2 証明書のエクスポートは下記のとおりに行います。

```
bash$ aica export –sn 2010 –o user.cer
CA PKCS#12 file open
Input PKCS#12 Password :  （パスワードを入力）
success to export a certificate (sn: 2010)
```

同様に、秘密鍵や PKCS#12 ファイルのエクスポートも可能ですが、ユーザ秘密鍵にアクセスする場合は、アクセスパスワードを入力が要求されます。また、ファイル出力時には出力用パスワードの入力も要求されます。

4.10. 秘密鍵のインポート

CA にはユーザ秘密鍵を保管する機能があり、発行した証明書と秘密鍵のペアがそろっている場合に、ユーザ秘密鍵のインポートが行えます。

1 aica import オペレーションにより CA にユーザ秘密鍵をインポートします。コマンドの形式は次のとおりです。

```
aica import [オプション]
オプション:
      -sn num        : シリアル番号を指定します
      -key file      : “file”名の秘密鍵ファイルを指定します
一般オプション:
      -sv path       : “サーバ名:CA 名”を指定します
      -ssl           : リモート CA 接続に SSL を使用します
      -clid name     : SSL クライアント証明書の ID を指定します
      -u loginid     : CA ユーザ名を指定します（非 SSL 接続）
      -p passwd      : CA ユーザパスワードを指定します（非 SSL 接続）
```

2 秘密鍵のインポートは下記のとおりに行います。

```
bash$ aica import –sn 3247 –key user.key
CA PKCS#12 file open
Input PKCS#12 Password :  （パスワードを入力）
get a private key from a file.
Open Private Key:  （パスワードを入力）

save a private key to the CA. input new password.
Save Private Key:  （パスワードを入力）
Verifying - Save Private Key:  （パスワードを入力）
success to import a private key (sn:3247)
```

ユーザの秘密鍵と指定したシリアル番号の証明書の鍵ペアは必ずマッチする必要があります。また、CA にユーザ秘密鍵をインポートするときには、必ずアクセスパスワードを設定します。

4.11. 秘密鍵の削除

CA には発行した証明書とユーザ秘密鍵を保管する機能があります。このうちユーザ秘密鍵は、CA の鍵ストアから削除することが可能です。また、CA の CSR キューに保管されている証明書要求も削除することが可能です。

1 aica delete オペレーションにより CA の鍵ストアからユーザ秘密鍵を削除します。コマンドの形式は次のとおりです。

```
aica delete [オプション]
オプション:
      -sn num       : シリアル番号を指定します
      -key          : “file”名の秘密鍵ファイルを指定します
      -csr num      : 削除する CSR の AcceptID 番号を指定します
一般オプション:
      -sv path      : “サーバ名:CA 名”を指定します
      -ssl          : リモート CA 接続に SSL を使用します
      -clid name    : SSL クライアント証明書の ID を指定します
      -u loginid    : CA ユーザ名を指定します（非 SSL 接続）
      -p passwd     : CA ユーザパスワードを指定します（非 SSL 接続）
```

2 ユーザ秘密鍵の削除は下記のとおりに行います。

```
bash$ aica delete –sn 3247 -key
CA PKCS#12 file open
Input PKCS#12 Password :  （パスワードを入力）
success to delete a private key (sn: 3247)
```

4.12. プロファイル設定の表示

CA が保持しているプロファイル設定の表示を行います。このコマンドを実行することで、CA の Issuer と Subject、指定したプロファイルのカレントシリアルナンバ、有効日数、拡張情報を表示します。

1 aica print オペレーションによりCAとプロファイルのカレント設定の表示を行います。コマンドの形式は次のとおりです。

```
aica print [オプション]
オプション:
      -pf name       : “name”プロファイルの設定を表示します
      -all           :  全てのプロファイル設定を表示します
一般オプション:
      -sv path       : “サーバ名:CA 名”を指定します
      -ssl           :  リモート CA 接続に SSL を使用します
      -clid name     : SSL クライアント証明書の ID を指定します
      -u loginid     : CA ユーザ名を指定します（非 SSL 接続）
      -p passwd      : CA ユーザパスワードを指定します（非 SSL 接続）
```

2 CA とプロファイル設定の表示は下記のとおりに行います。

```
bash$ aica print
CA PKCS#12 file open
Input PKCS#12 Password :  （パスワードを入力）
Issuer   :
   C=JP, O=nec corporation, OU=developer unit,
Subject :
   C=JP, O=nec corporation, OU=developer unit,
-----------------------------------
Certificate Profile : SMIME user
   certificate version    : 3
   current serial number: 20
   signature algorithm    : SHA1WithRSAEncryption
   certificate begin       : at signing time
   certificate end          : 7 days (604800 sec)
   certificate update      : 0 days (0 sec)
   subject template         :
      C=JP, O=nec corporation, OU=developer unit,
   profile working policy :
      reuse same subject DN ... allow
      reuse same public key ... allow
      replace subject DN with template ... allow
   CSR subject matching policy :
      C:option, ST:option, L:option, O:option
      OU:option, CN:option, UID:option, E:option
```

```
certificate extensions :
  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:…
```

プロファイル名を指定しない場合は、指定プロファイルを"SMIME user"として設定の表示を行います。

なお、各項目の内容は以下の通りです。

certificate version	証明書のバージョンです。
current serial number	この証明書プロファイルで次に割り当て予定のシリアル番号です。
signature algorithm	証明書の署名アルゴリズムです。
certificate begin	証明書の有効期限開始日時を表します。
certificate end	証明書の有効期限終了日時を表します。
certificate update	同一鍵、同一シリアル番号による証明書の有効期限更新に対して、更新禁止期間を設定できます。0 の場合、常に有効期限更新が可能です。証明書の有効期間と同じ場合、有効期限更新が行えなくなります。
subject template	CSR DN マッチングやサブジェクトリプレースで使用される証明書サブジェクト DN のテンプレートです。
profile working policy	証明書プロファイルのポリシ設定を表します。 このポリシにより、同一サブジェクトの禁止や同一鍵の禁止、サブジェクトのリプレースが行われます。
CSR subject matching policy	CSR DN マッチングのポリシ設定を表します。 CSR に含まれるサブジェクト DN と、DN テンプレートの各属性型（C,O,OU,CN 等）のマッチング設定を表します。
certificate extensions	証明書に追加する拡張情報の一覧を表示します。

次に、発行した証明書をリスト表示する aica list オペレーションについて説明します。

1 aica list オペレーションによりプロファイルが保持している証明書の発行リストを表示します。コマンドの形式は次のとおりです。

```
aica list [オプション]
オプション:
      -pf name       : “name”プロファイルの発行リストを表示します
一般オプション:
      -sv path       : “サーバ名:CA 名”を指定します
      -ssl           : リモート CA 接続に SSL を使用します
      -clid name     : SSL クライアント証明書の ID を指定します
      -u loginid     : CA ユーザ名を指定します（非 SSL 接続）
      -p passwd      : CA ユーザパスワードを指定します（非 SSL 接続）
```

2 プロファイルが保持している証明書の発行リストの表示は下記のとおりに行います。

```
bash$ aica list
CA PKCS#12 file open
Input PKCS#12 Password :  （パスワードを入力）

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,
```

プロファイル名を指定しない場合は、指定プロファイルを″SMIME user″として証明書の発行リストの表示を行います。なお、state は、'C'が CA 証明書、'R'が証明書失効、'E'が期限切れ、'K'が CA 側に秘密鍵を保管している状態を表します。また、表示時刻の形式は、″year/month/day hour:minute″ となっています。

4.13. プロファイル設定の更新

CA が保持しているプロファイル設定の更新を行います。このコマンドを実行することで、指定したプロファイルのカレントシリアルナンバ、有効日数の設定を行うことができます。

1 aica pfset オペレーションによりプロファイルの設定を行います。コマンドの形式は次のとおりです。

```
aica pfset [オプション]
オプション:
    -pf name       : “name”プロファイルの設定を行ないます
        （標準:SMIME user）
    -ver  num      : 証明書、CRLのバージョンを指定します
    -sn   num      : 証明書、CRLのシリアル番号を指定します
    -sec  num      : 証明書、CRLの有効期間（秒）を指定します
    -days num      : 証明書、CRLの有効期間（日）を指定します
    -upd num       : 証明書の更新禁止期間を指定します
    -start time    : 証明書、CRLの開始日時を指定します
        （フォーマットは”YYYYMMDDHHMMSSZ”で”.”にてクリアします）
    -end  time     : 証明書、CRLの終了日時を指定します
        （フォーマットは”YYYYMMDDHHMMSSZ”です）
    -hash alg      : 証明書、CRLの署名用ハッシュ関数を指定します
        （”sha1”,”md5”,”md2”を指定します）
    -sbjtmpl       : 証明書のサブジェクト DN テンプレートを設定します
    -pol           : 証明書の発行ポリシを設定します。
    -dnpol         : CSR サブジェクト DN のマッチングポリシを設定します
一般オプション:
    -sv path       : “サーバ名:CA 名”を指定します
    -ssl           : リモート CA 接続に SSL を使用します
    -clid name     : SSL クライアント証明書の ID を指定します
    -u loginid     : CA ユーザ名を指定します（非 SSL 接続）
    -p passwd      : CA ユーザパスワードを指定します（非 SSL 接続）
```

2 プロファイルが保持しているの有効日数の設定は下記のとおりに行います。

```
bash$ aica pfset –days 130
CA PKCS#12 file open
Input PKCS#12 Password :  （パスワードを入力）

Update Profile information.
```

プロファイル名を指定しない場合は、指定プロファイルを″SMIME user″として有効日数の設定を行います。また、CRL の有効日数を変更する場合は、それぞれ CRL のプロファイル名を指定して変更を行います。

aica pfset –pf CRL –days 3

aica pfset -pf ARL -days 7

aica pfset -pf CRL-All -days 3

上記の例では、発行する CRL,CRL-All の有効日数を 3 日とし、ARL の有効日数を 7 日とします。

3 サブジェクトテンプレートの設定は下記のとおりに行います。

```
bash$ aica pfset –sbjtmpl
CA PKCS#12 file open
Input PKCS#12 Password :  （パスワードを入力）
-----
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 プロファイルポリシの設定は下記のとおりに行います。

```
bash$ aica pfset –pol
CA PKCS#12 file open
Input PKCS#12 Password :  （パスワードを入力）
-----
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 CSR マッチングポリシの設定は下記のとおりに行います。

```
bash$ aica pfset –dnpol
CA PKCS#12 file open
Input PKCS#12 Password :  （パスワードを入力）
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. プロファイル拡張情報の更新

証明書拡張情報はプロファイルごとに設定がおこなえます。証明書を発行するときに、指定したプロファイルに設定してある拡張情報を証明書に追加するため、ここで設定した情報がそのまま証明書発行時に反映されます。プロファイルの拡張情報の設定は、スクリーンに表示される設定情報に従い、ステップバイステップで行うことができます。

1 aica pfext オペレーションによりプロファイルの証明書拡張情報の設定を行います。コマンドの形式は次のとおりです。

```
aica pfext [オプション]
オプション:
      -pf name       : “name”プロファイルの設定を行ないます
      -bscons        : Basic Constraints 拡張情報を設定します
      -keyusage      : Key Usage 拡張情報を設定します
      -extkeyusage   : Extended Key Usage 拡張情報を設定します
      -authkeyid     : Authority Key Identifier 拡張情報を設定します
      -sbjkeyid      : Subject Key Identifier 拡張情報を設定します
      -issaltname    : Issuer Alt Name 拡張情報を設定します
      -sbjaltname    : Subject Alt Name 拡張情報を設定します
      -certpol       : Certificate Policy 拡張情報を設定します
      -polmap        : Policy Mapping 拡張情報を設定します
      -crldp         : CRL Distribution Point 拡張情報を設定します
      -authinfo      : Authority Info Access 拡張情報を設定します
      -ocspnochk     : OCSP no check 拡張情報を設定します
      -nscrl         : Netscape CRL URL 拡張情報を設定します
      -nscomm        : Netscape comment 拡張情報を設定します
      -nstype        : Netscape cert type 拡張情報を設定します
      -crlnum        : CRL Number 拡張情報を設定します
      -issdp         : Issuing Distribution Point 拡張情報を設定します
      -crlreason     : CRL ReasonCode entry 拡張情報を設定します
一般オプション:
      -sv path       : “サーバ名:CA 名”を指定します
      -ssl           : リモート CA 接続に SSL を使用します
      -clid name     : SSL クライアント証明書の ID を指定します
      -u loginid     : CA ユーザ名を指定します（非 SSL 接続）
      -p passwd      : CA ユーザパスワードを指定します（非 SSL 接続）
```

2 拡張情報の変更を行う場合、下記のとおりに行います（例は BasicCostraints）。

```
aica pfext -bscons
CA PKCS#12 file open
Input PKCS#12 Password :  （パスワードを入力）

Add or delete X.509 extension for profile.
Select operation (a..add/d..delete/c..cancel)[a]:
```

パスワードの入力後、オプションで指定した拡張情報の追加（既にある場合は更新）、削除、または操作のキャンセルを問い合わせます。追加を選択した場合、指定した各拡張情報によってそれぞれの設定項目が表示されます。削除を指定した場合、すぐに操作が完了しますが、指定した拡張情報がない場合は何も変更されません。

- BasicConstraints の設定は下記のように行なわれます。

```
Configure detail information for BasicConstraints extension.
set critical flag (y/n)[y]: y       （クリティカルフラグの設定）
set CA flag    (y/n)[n]: n
set pathLength (-1 means 'NULL') [-1]: -1
'BasicConstraints' extension is added to 'SMIME user' profile.
```

上記の例では、ユーザ証明書向けの設定を行っており、CA フラグを OFF にし pathLength を無効（NULL）に指定しています。

- KeyUsage の設定は下記のように行なわれます。

```
Configure detail information for KeyUsage extension.
set critical flag (y/n)[y]: n       （クリティカルフラグの設定）
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.
```

上記の例では、ユーザ証明書向けの設定を行っており、署名、否認防止、鍵とデータの暗号化フラグを ON にし、証明書や CRL の発行（署名）を OFF にしています。

- ExtendedKeyUsage の設定は下記のように行なわれます。

```
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.
```

上記の例では、”1.2.33.44”のオブジェクト ID を拡張 KeyUsage として設定しています。OID が不正な場合は、処理を中止します。

- AuthorityKeyIdentifier の設定は下記のように行なわれます。

```
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.
```

AuthorityKeyIdentifier では、通常 key identifier のみを有効にして、それ以外は拡張情報として付加しないように設定します。

- SubjectKeyIdentifier は指定を行った時点で、鍵識別子の追加を設定します。

- IssuerAltName の設定は下記のように行なわれます。

```
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.
```

IssuerAltName では、証明書 Issuer の DN で表わせない CA の追加情報を設定します。上記の例では、Email アドレスを追加しています。

- SubjectAltName は指定を行った時点で、サブジェクト別名の追加を設定します。

- Certificate Policies の設定は下記のように行なわれます。

```
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.
```

上記の例では、1.2.33.44 のポリシを定義し、その内容を CPS Uri で示されるアドレスに配置しています。

- PolicyMappings の設定は下記のように行なわれます。

```
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.
```

上記の例では、証明書発行元のドメインポリシを 1.2.33.44 とし、証明書発行先のドメインポリシを 2.3.44.55 としてマッピングを行っています。マッピングは 1 つだけ設定することが可能です。

- CRL Distribution Point の設定は下記のように行なわれます。

```
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.
```

CRL の配布ポイントとして、GeneralName の URL を1つ指定します。また、複数の Distribution Point を加える場合は、最後の”Do you continue?”に”y”を入力し、引き続き CRL の配布ポイントを追加することが可能です。

- AuthorityInformationAccess の設定は下記のように行なわれます。

```
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.
```

上記の例では、OCSP の発行者情報を設定しており、その OCSP の CGI の URI をアドレスとして指定しています。

- OCSP no check フラグは指定を行った時点で、フラグの追加を設定します。
- CRL Number は指定を行った時点で、拡張情報の追加を設定します。
- CRL ReasonCode は指定を行った時点で、エントリ拡張情報の追加を設定します。
- IssuingDistributionPoint の設定は下記のように行なわれます。

```
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.
```

上記の例では、onlyContainsUserCerts を指定することで、CRL がユーザ失効情報のみをリストしていることが示されます。

この他のネットスケープ関連の拡張情報については、コメントフィールドまたは証明書用途の指定であり、表示内容に従って入力を行ってください。

4.15. プロファイルの追加、削除と名称変更

1つの CA で複数のプロファイルを保持することができ、証明書を発行するグループ毎にプロファイルを用意したり、必要のなくなったプロファイルを削除することができます。また、プロファイル名の変更を行うことができます。

1 aica prof オペレーションによりプロファイルの追加、削除と名称変更を行います。コマンドの形式は次のとおりです。

```
aica prof [オプション]
オプション:
      -add          : プロファイルを追加します
      -del          : プロファイルを削除します
      -rename       : プロファイル名を変更します
一般オプション:
      -sv path      : “サーバ名:CA 名”を指定します
      -ssl          : リモート CA 接続に SSL を使用します
      -clid name    : SSL クライアント証明書の ID を指定します
      -u loginid    : CA ユーザ名を指定します（非 SSL 接続）
      -p passwd     : CA ユーザパスワードを指定します（非 SSL 接続）
```

2 プロファイルの追加は下記のとおりに行います。プロファイルテンプレートを選択し、プロファイル名を付けて設定を保存します。

```
bash$ aica prof –add
CA PKCS#12 file open
Input PKCS#12 Password :  （パスワードを入力）
-------
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 templete number [0]: 5

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

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

3 プロファイルの削除は下記のとおりに行います。CA が保持しているプロファイルがリストされるので、削除するプロファイルの番号を指定し設定を保存します。

```
bash$ aica prof –del
CA PKCS#12 file open
Input PKCS#12 Password :  （パスワードを入力）
-------
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 プロファイルの名称変更は下記のとおりに行います。CA が保持しているプロファイルがリストされるので、プロファイルの番号を指定し、新しい名称を指定します。

```
bash$ aica prof –rename
CA PKCS#12 file open
Input PKCS#12 Password :  （パスワードを入力）
-------
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. オペレータの追加と削除

CAをローカルマシンにて起動し、直接操作を行う場合はCAのマスタパスワードを入力することで CA の操作が可能ですが、リモートコンソールから CA サーバに接続して使用する場合は、必ずユーザ(オペレータ)の認証が必要です。この CA サーバに接続可能なオペレータ証明書の発行、ID / Password やアクセス権限の設定が可能です。

1 aica user オペレーションによりオペレータを追加又は削除します。コマンドの形式は次のとおりです。

```
aica user [オプション]
オプション:
      -add          : オペレータを追加します
      -del          : オペレータを削除します
      -mod          : オペレータの情報を更新します
      -cpw          : オペレータのパスワードを更新します
オプション(オフラインのみ):
      -addop        : SSL 認証用のオペレータを追加します
      -addraop      : RA アドミニストレータ証明書を発行します
      -smlib        : PKCS#11 ライブラリを指定(RA アドミニストレータ用)
      -smlabel      : PKCS#11 ラベル名(RA アドミニストレータ用)
一般オプション:
      -sv path      : "サーバ名:CA 名"を指定します
      -ssl          : リモート CA 接続に SSL を使用します
      -clid name    : SSL クライアント証明書の ID を指定します
      -u loginid    : CA ユーザ名を指定します(非 SSL 接続)
      -p passwd     : CA ユーザパスワードを指定します(非 SSL 接続)
```

2 オペレータの追加は下記のとおりに行います。

```
bash$ aica user –addop
CA PKCS#12 file open
Input PKCS#12 Password :  (パスワードを入力)
-------
Add a new Operator to this CA.
.....oo
.oo
Input PASS Phrase:  (パスワードを入力)
Verifying - Input PASS Phrase:  (パスワードを入力)
success to modify CA user.
```

3　オペレータの削除は下記のとおりに行います。

```
bash$ aica user –del
CA PKCS#12 file open
Input PKCS#12 Password :　（パスワードを入力）
-------
Delete a CA user from this CA.

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

CA オペレータ証明書を発行すると、そのオペレータは CA に対して全ての操作が可能な Administrator 権限を持つことになります。このアクセス権を変更する場合は、以下のようにユーザ情報を変更します。なお、オペレータユーザ名が不明な場合は、ca.passwd ファイルを参照してください。

```
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.
```

アクセス権限には以下が指定できます。

B	Bind 要求。CA サーバに接続するためには、必須項目です。
S	Sign 操作要求。証明書の即時発行を許可します。
L	List 操作要求。証明書状態リストの取得を許可します。
A	Prof List 操作要求。CA が保持するプロファイル一覧の取得を許可します。
X	拡張操作要求。拡張オペレーションを許可します。通常この値は指定しません。
P [vadm]	Prof 操作要求。プロファイル一般情報の操作を許可します。v..表示、a..追加、d..削除、m..更新を表し、P の後に続けることで個々の操作を許可します。P 単独では全ての操作を許可します。
E [vadm]	Prof Ext 操作要求。証明書拡張情報の操作を許可します。v..表示、a..追加、d..削除、m..更新を表し、E の後に続けることで個々の操作を許可します。E 単独では全ての操作を許可します。
C [urne]	Cert 操作要求。ユーザ証明書の操作を許可します。u..更新、r..失効、n..失効取り消し、e..出力を表し、C の後に続けることで個々の

	操作を許可します。C 単独では全ての操作を許可します。
K [ied]	Key 操作要求。ユーザ秘密鍵の操作を許可します。i..保管、e..出力、d..削除を表し、K の後に続けることで個々の操作を許可します。K 単独では全ての操作を許可します。
Q [pedj]	CSR 操作要求。CSR キューの操作を許可します。p..POST、e..出力、d..削除、j..発行判定を表し、Q の後に続けることで個々の操作を許可します。Q 単独では全ての操作を許可します。
R [ie]	CRL 操作要求。CRL の操作を許可します。i..発行、e..出力を表し、R の後に続けることで個々の操作を許可します。R 単独では全ての操作を許可します。

5. その他の操作

証明書ビューアやコンバータなど、NAREGI CA コマンドのその他の操作について説明します。

5.1. 証明書ストア

aistore コマンドにより、証明書ストアに対する証明書･CRL のインポートやエクスポート操作を行います。証明書ストアは署名などの検証動作に必要であり、CA 証明書や CRL などを保管します。

証明書ストアの構成は、ROOT,SUB,OTHER,MY の四つのカテゴリに分けられ、ROOT,SUB,OTHER には、証明書と CRL を保管することができます。なお、ROOT はルート CA 証明書が保管され、SUB には中間 CA 証明書、MY には本人向けのユーザ証明書、OTHER には他人のユーザ証明書が保管されます。

- コマンドの形式は次のとおりです。

```
aistore [操作] [オプション]
操作:
      -i file          : “file”名のファイルをストアにインポートします
      -e file          : 指定のアイテムを“file”名でエクスポートします
      -l               : 指定のストアに保管されているアイテムを
                         リスト表示します。
      -p               : 指定のアイテムを表示します
      -d               : 指定のアイテムをストアから削除します
オプション:
      -st store        : “store”名のストアを指定します
                         （my, other, sub, root の1つを指定）
      -tp type         : ストアの“type”を指定します
                         （cert, crl の1つを指定）
      -id id           : アイテムのユニーク ID を指定します
      -ef format       : エクスポート時のファイル形式を指定します
                         （der, pem, pk7, pk12 の1つを指定）
      -pw pwd          : エクスポート時のパスワードを指定します
```

- **証明書、CRL のリスト表示**

MY ストアの証明書をリスト表示する場合は、下記のように指定します。

```
bash$ aistore -l
オプション:
[unique-id]      subject                        serialNumber
-------------------------------------------------------------------------
[mycert00] C=JP, O=test org, CN=t.yamada,    30000
```

オプションに何も指定しない場合は、デフォルトで MY ストアを指定したことになります。
リストの表示形式は、証明書アイテムのユニーク ID、サブジェクト、シリアルナンバを1つのセットとして表示します。
また、ROOT ストアの CRL をリスト表示する場合は、下記のように指定します。

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

- **証明書、CRL のインポート**

証明書の種類を自動的に判別し適当なストアへインポートすることができます。
"-id" オプションによって、明示的にストア内部でのユニークな名称を指定することができます。ID を指定しない場合は、証明書のサブジェクトより自動的にユニーク ID が決定されます。このときインポートできるファイル形式は、X.509 DER,PEM 証明書と PKCS#7, PKCS#12 ファイルです。

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

CRLをインポートする場合は、CRLを発行したCA証明書を先にインポートしてから操作を行ってください。

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

- **証明書、CRL の表示**

MY ストアに保管されている任意の証明書を表示する場合は、ストアの名称と証明書や CRL のユニーク ID を指定する必要があります。ユニーク ID はインポート時に指定、または自動的に割り当てられますが、ID が不明の場合は証明書ストアをリスト表示して確認を行ってください。なお、コマンドの実行は下記のように行います。

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

ROOT ストアに保管されている任意の CRL を表示する場合は、下記のように指定します。

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

- **証明書、CRL のエクスポート**

MY ストアに保管されている任意の証明書をエクスポートする場合は、ストアの名称と証明書や CRL のユニーク ID を指定する必要があります。コマンドの実行は下記のように行います。

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

ROOT ストアに保管されている任意の CRL をエクスポートする場合は、下記のように指定します。

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

- **証明書、CRL の削除**

  MY ストアに保管されている任意の証明書を削除する場合は、ストアの名称と証明書や CRL のユニーク ID を指定する必要があります。コマンドの実行は下記のように行います。

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

  ROOT ストアに保管されている任意の CRL を削除する場合は、下記のように指定します。

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

5.2. 証明書検証

certvfy コマンドにより、証明書の検証を行います。検証にに必要な CA 証明書や CRL などは、先に証明書ストアにインポートしておく必要があります。そして、X.509 PEM, DER の証明書ファイルを指定することで検証を行うことができます。

- コマンドの形式は次のとおりです。

```
certvfy [オプション] file
オプション:
    -depth num    : 検証時のパスの深さを設定します
    -ifcrl        : CRL が存在するときだけ失効チェックをします
    -nocrl        : 検証時に失効チェックを行いません
    -novfycrl     : CRL 自体の検証を行ないません
```

証明書ストアに CA 証明書が保存してあり、CRL が保存されていない状態で証明書検証をおこなうと下記のようになります。

```
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
```

オプションを何もつけない場合、完全な証明書検証を行います。上記の場合、depth=1（深さ1でエンドユーザ証明書）で証明書の失効状態を検証できなかった（CRL を証明書ストアから発見できなかった）ため、検証エラーが起きて終了しています。

上記と同じ状態で、-ifcrl オプションをつけて証明書検証をおこなうと下記のようになります。

```
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
```

-ifcrl オプションを付けると、CRL が存在する場合のみ失効状態を検証します。CRL がない場合は、証明書は失効されていないものとして検証を行い、上記のように検証が成功します。また、CRL が存在しても、-nocrl オプションによって失効状態の検証を行わないように指定することもできます。

5.3. 証明書ビューア

certview コマンドにより、各種証明書や秘密鍵、PKCS ファイルのテキスト表示を行います。先にコマンドの形式を解説し、その使用例を挙げます。

- コマンドの形式は次のとおりです。

```
certview [オプション] file1 file2 ...
オプション:
      -p passwd     : 入力ファイルに対するパスワードを指定します
```

- 表示可能なファイルの種類は次のとおりです。

```
証明書:          *.cer, *.pem, *.p7b
相互認証証明書:  *.ccp
秘密鍵:          *.key, *.pem
鍵パラメータ:    *.pem
CRL:             *.crl, *.pem, *.p7b
証明書要求:      *.req, *.pem, *.p10
PKCS#12:         *.p12, *.pfx
PKCS#8:          *.crtx, *.pem
```

入力されたファイルの種類を自動的に判別し、テキスト表示を行います。また、ファイルの形式（DER、PEM、BASE64）に関わらず表示を行うことができます。

5.4. 証明書変換

certconv コマンドにより、各種証明書や秘密鍵、PKCS ファイルの相互変換を行います。先にコマンドの形式を解説し、その使用例を挙げます。

- コマンドの形式は次のとおりです。

certconv [オプション] **file1 file2 …**
オプション:

-p12 **file**	: “file”名で PKCS#12 ファイルを保存します
-p7b **file**	: “file”名で PKCS#7 ファイルを保存します
-cert **file**	: “file”名で証明書ファイルを保存します
-crl **file**	: “file”名で CRL ファイルを保存します
-key **file**	: “file”名で秘密鍵ファイルを保存します
-crtp **file**	: “file”名で相互認証証明書ファイルを保存します
-p10 **file**	: “file”名で証明書要求ファイルを保存します
-p8 **file**	: “file”名で PKCS#8（鍵）ファイルを保存します
-pwin **pwd**	: 入力ファイルに対するパスワードを指定します
-pwout **pwd**	: 出力ファイルに対するパスワードを指定します
-pem	: 出力ファイルを PEM 形式にします（標準）
-der	: 出力ファイルを DER 形式にします
-depth **dp**	: “dp” の深さの証明書を出力します
-noenc	: 出力ファイルを暗号化しません（秘密鍵）

- **証明書，証明書要求, CRL, PKCS#7, PKCS#8 の PEM ←→ DER 形式変換**

```
certconv -der -cert out.cer in.cer
certconv -pem -p7b out.p7b in.p7b
```

"-der" オプションで DER 形式の出力を行います。
"-pem" オプションで PEM 形式の出力を行います(標準)。

- **チェイン有り証明書や CRL → PKCS#7 (*.p7b)**

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

入力は in.cer, in.crl, ca.cer で出力が out.p7b です。

- **PKCS#7 → 任意の深さの証明書**

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

この例では、深さ 0 (即ちチェインの最上位) の証明書と CRL を in.p7b ファイルから取り出しています。

- **チェイン有り証明書 + 秘密鍵 → 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

  入力は in.key, in.cer, ca.cer であり出力は out.p12 です。

  出力の PKCS#12 ファイルは Netscape 互換の形式をとっています。

- **PKCS#12 → 任意の深さの証明書、秘密鍵**

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

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

  入力は in.p12 であり出力は out.cer と out.key です。PKCS#12 (PFX) ファイルは Microsoft と Netscape の両方の形式が読み込み可能です。なお、1 番目の例では出力する証明書の深さを指定していませんが、この場合は最も深い階層、即ちユーザの証明書が出力されます。

- **PKCS#8 ←→ PEM 秘密鍵**

  certconv -noenc -p8 out.crtx in.key

  certconv -p8 out.crtx in.p12

  入力は in.key や in.p12 等の秘密鍵を含む形式となり、出力は PKCS#8 の out.crtx となります。この例のように、秘密鍵を "-noenc" オプションによって暗号化せずに出力することが可能です。

- **証明書 ←→ 相互認証証明書ペア**

  certconv –ccp out.ccp a2b.cer b2a.cer

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

  上の例では、入力は a2b.cer と b2a.cer であり、出力は out.ccp となります。 また、下の例では入力は in.ccp であり、相互認証証明書ペアの depth=1 である issuedByThisCA の証明書が出力されます。

5.5. 証明書要求作成

証明書取得コマンドである grid-certreq はシェルスクリプトで記述されています。このシェルスクリプトにて、証明書ライフサイクル管理コマンドを呼び出しており、この管理コマンド名を certreq と呼びます。
証明書に対するオペレーションとしては、-new オプション、-rvk オプション、-upd オプションがあります。-new オプションを指定すると、RA サーバに接続し、証明書の発行処理を行います。-rvk や-upd は証明書の失効、または更新処理を行いますが、必ず SSL クライアント認証を実行し、接続が確立した上でそのクライアント証明書に対して処理を行います。

- コマンドの形式は次のとおりです。

certreq [オプション]
オプション:
-algo **alg** : 生成する鍵ペアのアルゴリズムです。
rsa,dsa,ecdsa を指定できます（標準:rsa）
-size **num** : 鍵ペアの鍵長を指定します（標準:512bit）
-req **file** : “file”名で証明書要求ファイルを保存します
-key **file** : “file”名で秘密鍵ファイルを保存します
-kp **passwd** : 秘密鍵の保存用パスワードを指定します
-noenc : 秘密鍵を暗号化せずに保存します
-der : 証明書要求の保存形式を DER にします
-alt : Subject Alt Name を指定します
-s : サブジェクト入力モードをシンプルにします
リモート操作オプション:
-new **license** : 証明書要求作成し証明書を申請します
OneTime のライセンス ID を指定します
-rvk : SSL クライアント証明書を失効します
-upd : SSL クライアント証明書を更新します
-recv **accID** : “accID”（acceptID）の証明書を取得します
-in **file** : “file”名の CSR ファイルを入力します
-cer **file** : “file”名で証明書ファイルを保存します
-cacer **file** : “file”名で CA 証明書ファイルを保存します
-p12 **file** : “file”名で PKCS#12 ファイルを保存します
接続オプション:
-sv **path** : “サーバ名:CA 名”を指定します
-pt **port** : “port”ポート番号を指定します
-ssl : リモート CA 接続に SSL を使用します
-clid **name** : SSL クライアント証明書の ID を指定します
-clcer **file** : SSL クライアント証明書のファイル名を指定します
-clkey **file** : SSL クライアント秘密鍵のファイル名を指定します

5.6. Web Enroll 補助ツール

Web Enroll では、ユーザの認証に ID/Password や One Time の License ID を使用します。このとき、en.passwd ファイルや en.license ファイルの配置が必要であり、データ作成支援ツールとして aienrtool が存在します。先にコマンドの形式を解説し、その使用例を挙げます。

- コマンドの形式は次のとおりです。

```
aienrtool [オプション]
オプション:
      -csv file       : passwd 一括生成もしくはライセンス一括
                        メール通知用の CSV ファイルを指定します
[enroll passwd ファイルモード]
      -u name         : ユーザ名を指定します
      -p passwd       : パスワードを指定します
[enroll license ファイルモード]
      -lic            : ライセンス生成モードを指定します
      -n num          : 出力ライセンス数を指定します
      -s num          : スタート番号を指定します
      -hd text        : ライセンスヘッダ文字列を指定します
      -iv text        : 初期ベクトルを指定します
[generate license and send email mode]
      -lsend num      : ライセンス一括メール通知を行います
                        num は該当する RA RegInfo を指定します
```

- en.passwd 生成モード（1 行）

```
bash$ aienrtool -u user001 -p abcde
user001:XLvYdWJfeNdVxVq0fLG4K3WZl4Pn64o=
```

-u と-p でユーザ名とパスワードを入力し、標準出力に en.passwd ファイル形式データを出力します。ファイルとして保存する場合は、

aienrtool –u user001 –p abcde > en.passwd

のようにリダイレクトを行ってください。

- en.passwd 生成モード（CSV）

```
bash$ more myuser.csv
user001,abcde
user002,fghij
bash$ aienrtool -csv myuser.csv
user001:UmvvpBSF9vBogm6gTpgKqkHiO/nQ2EE=
user002:GsPapVqiV5+d3NFNUcjIuPCL484r0tA=
```

-csv で user,passwd の形式をもつ CSV ファイルを指定し、標準出力に en.passwd ファイル形式データを出力します。

- en.license 生成モード

```
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
```

-lic オプションを指定すると、ライセンス ID 生成モードになります。Web Enroll では、en.license ファイルに含まれる文字列とユーザから POST された文字列の単純なマッチングを行っているだけなので、このツール以外で生成したライセンスでも問題なく動作します。他に簡易にライセンス ID が生成できない場合は、本ツールをお使いください。
aienrtool は、まったくランダムな文字列を出力するのではなく、ユニークなライセンスを生成するようになっています。このため、ライセンス生成時に同じスタート番号やヘッダ文字列、初期ベクタを使用すると、同じライセンス番号が生成されます。
ユニークなライセンス ID を生成する場合、以下の手順により行ってください。

① ヘッダ文字列を決めます。上記の例では”MYID00”です。
② スタート番号を決めます。上記の例では 0 です。
③ 初期ベクタを決めます。上記の例では”randseed17asf4d17ds”です。

例えば、ライセンス ID を 1000 個出力する場合は、-n 1000 として出力します。次に、追加で 1000 個出力する場合は、-s 1000 としてスタート番号が重ならないようにすることで、新しくユニークなライセンス ID を生成できます。

- ライセンス一括メール通知モード（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
```

-csv でメールアドレスの形式をもつ CSV ファイルを指定し、指定の [RAd RegInfo *] 向けにライセンス一括メール通知と en.license もしくは LDAP エントリデータの更新を行います。

5.7. aica.cnf 設定補助ツール

CA サーバや CRL Publisher の自動起動を設定したい場合に aica.cnf に暗号化したパスワードの設定が必要です。この暗号パスワードの設定やシェルスクリプト用の補助ツールとして aiconftool が存在します。先にコマンドの形式を解説し、その使用例を挙げます。

- コマンドの形式は次のとおりです。

```
aiconftool [オプション]
[item オプション]
      -add          : アイテムを追加、又は更新します
      -del          : アイテムを削除します
      -mod          : アイテムが存在する場合に更新します
      -s section    : セクション名を指定します
      -i item       : アイテム名を指定します
      -v value      : アイテムの値を指定します
      -enc          : アイテムの値を暗号化します
[CA オプション]
      -addca        : CA をサーバに登録します
      -delca        : サーバから登録済み CA を削除します
      -addcrl       : CRL Publisher の設定を追加します
      -delcrl       : CRL Publisher の設定を削除します
      -addenr       : Web Enroll の設定を追加します
      -delenr       : Web Enroll の設定を削除します
      -ca name      : CA 名を指定します
      -path path    : CA のパスを指定します
      -pw passwd    : CA のパスワードを指定します
      -conf path    : aica.cnf のパスを指定します
```

- 自動起動用 CA パスワードの変更

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

“CAd”セクションの”capwd.0”アイテムに対して、値の更新を行います。この場合、パスワードの値である、”abcde”を暗号化して更新を行います。

aiconftool は操作結果などの出力を行わないため、操作を行ったあとは、直接 aica.cnf ファイルを表示し値が更新されているか確認してください。

5.8. aira オフライン CA 用コマンド

オフライン CA モードでは、証明書要求が RA サーバ上に保管され、手動で証明書の発行や配置を行う必要があります。このモードで使用する aira コマンドについて説明します。

- CSR の表示

```
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-----
MIIDRDCCAq0CAQAwZzELMAkGA1UEBhMCSlAxDTALBgNVBA
oTBFRFU1QxDTALBgNV
...
-----END CERTIFICATE REQUEST-----
```

csr –show オプションで acceptID 番号を入力して CSR の内容と PEM データを表示します。通常、acceptID 番号はメールにて、ユーザと RA 管理者に通知されます。

- CSR の破棄

```
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]:
```

csr –reject オプションで acceptID 番号を入力して CSR の内容を表示し、申請を拒否することが出来ます。aienroll プロセスが動作している場合、ユーザに申請拒否が通知されます。この操作を行うと、CSR キューからファイルが削除されます。

- 証明書の配置

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

csr –cert オプションで証明書を指定することで、同じ鍵ペアをもつ CSR が存在するかチェックし、問題なければ外部配布用の PKCS#7 形式に変換して証明書を配置します。aienroll プロセスが動作している場合、ユーザに証明書発行が通知されます。