opensslでオレオレ認証局を開局して証明書を発行する

2017/04/15
証明書の期限が来て作り直しした際に気づいた間違い等々を修正

2016/04/12
openssl で自己認証局を作る自分用メモ。

環境は FreeBSD 10.2 x86-64環境□pkgまたはportsからopenssl導入済みなのが前提。

証明書の中身って何？

- 証明書は、申請者の公開鍵と認証局の電子署名のペア。
- 電子署名は、申請者の署名要求を認証局の署名鍵で署名処理したデータ。
- 署名要求は、申請者の秘密鍵から生成した公開鍵と申請者を識別する情報のペア。

証明書の電子署名は、認証局の証明書に含まれる検証鍵(認証局の署名鍵から生成した公開鍵)で検証できる。
認証局は、認証局の証明書を公開する。

参照サイト

- WikiPedia - 証明書署名要求(CSR)
- WikiPedia - 公開鍵証明書(証明書)
- Wikipedia - 電子署名

電子署名は三つのフェーズからなります. 鍵生成, 署名, 検証です.
鍵生成フェーズでは署名に使う鍵と検証に使う鍵を作ります. ここでは署名に使う鍵を署名鍵, 検証に使う鍵を検証鍵と呼ぶことにします. 公開鍵/秘密鍵という言い方や, 暗号化鍵/復号鍵という言い方は避けましょう. これが混乱の元です.
署名フェーズでは署名鍵を持った署名者が文書に対して署名を行います.
検証フェーズでは検証者が検証鍵, 文書, 署名の3つに対して検証を行います.

なお, 公開鍵暗号方式を利用した署名と言った場合には, 鍵生成が暗号方式の鍵生成と共通であることが多いです. 署名方式の署名鍵と暗号方式の復号鍵が対応し, 署名方式の検証鍵と暗号方式の暗号化鍵が対応します.

186 @ hatenablog 暗号と署名の話より引用

オレオレ認証局開局の準備

生成するファイル

ファイル	説明
/etc/ssl/CA/private/cacert.key	オレオレ認証局の署名鍵。流出したら駄目なやつ。
/etc/ssl/CA/cacert.cert	オレオレ認証局の証明書。
/etc/ssl/CA/cacert.pem	オレオレ認証局の署名鍵と証明書の結合ファイル。流出したら駄目なやつ。
/etc/ssl/CA/cacert.der	オレオレ認証局の証明書を配布したい時の形式。 ブラウザにルートCA証明書として入れておきたい時などに。
とりあえず保持ファイル	**説明**
/etc/ssl/CA/private/cacert.withpass.key	オレオレ認証局の署名鍵パスフレーズ付き。流出したら駄目なやつ。
使用後は消すファイル	**説明**
/etc/ssl/CA/cacert.csr	オレオレ認証局の署名要求。

ディレクトリと初期ファイル作成

認証局 CA を開局するので /etc/ssl 以下に CA ディレクトリを作りそこに初期フォルダ、ファイルを配置。

```
root@hiroko:/etc/ssl # mkdir CA
root@hiroko:/etc/ssl/CA # cd CA
root@hiroko:/etc/ssl/CA # mkdir certs
root@hiroko:/etc/ssl/CA # mkdir crl         ←失効した証明書のリストがここに。
root@hiroko:/etc/ssl/CA # mkdir newcerts    ←署名した証明書のコピーが入る。証明書の失効処理でこれを使う。
root@hiroko:/etc/ssl/CA # mkdir private     ←ここの認証局の署名鍵をしまっておくよ！
root@hiroko:/etc/ssl/CA # touch index.txt
root@hiroko:/etc/ssl/CA # echo 01 > serial
root@hiroko:/etc/ssl/CA #
```

次に /etc/ssl/openssl.cnf を編集。編集個所は以下。適宜自分の環境様にカスタマイズする。

セクション	キー	値(変更後)
[CA_default]	dir	/etc/ssl/CA
	private_key	$dir/private/cacert.key
[req]	default_bits	2048
[req_distinguished_name]	countryName_default	JP
	stateOrProvinceName_default	hgotoh
	localityName	hgotoh
	0.organizationName_default	noname
	organizationalUnitName_default	noname

```
  ##################################################################
  [ CA_default ]

! dir           = /etc/ssl/CA           # Where everything is kept
  certs         = $dir/certs            # Where the issued certs are kept
  crl_dir               = $dir/crl              # Where the issued crl are
kept
```

```
  database      = $dir/index.txt        # database index file.

□
□

! private_key   = $dir/private/cacert.key # The private key

  ######################################################################
  [ req ]
! default_bits          = 2048
  default_keyfile       = privkey.pem

□
□

  [ req_distinguished_name ]
  countryName                   = Country Name (2 letter code)
! countryName_default           = JP
  countryName_min                       = 2
  countryName_max                       = 2

  stateOrProvinceName           = State or Province Name (full name)
! stateOrProvinceName_default   = hgotoh

! localityName                  = hgotoh

  0.organizationName            = Organization Name (eg, company)
! 0.organizationName_default    = noname

  organizationalUnitName                = Organizational Unit Name (eg,
section)
! organizationalUnitName_default        = noname
```

認証局の署名鍵を作る

認証局の署名鍵 /etc/ssl/CA/private/cacert.key を作成する。

```
root@hiroko:/etc/ssl/ # cd CA
root@hiroko:/etc/ssl/CA # openssl genrsa -aes256 -out private/cacert.key
2048
Generating RSA private key, 2048 bit long modulus
.....................................................................
.....................................................+++
.................................+++
e is 65537 (0x10001)
Enter pass phrase for private/cacert.key:             ←パスフレーズ入力
Verifying - Enter pass phrase for private/cacert.key:             ←再度パス
フレーズ入力
root@hiroko:/etc/ssl/CA #
```

署名鍵への署名要求を作る

認証局の署名鍵 /etc/ssl/CA/private/cacert.key への署名要求 /etc/ssl/CA/cacert.csr を作成する。

```
root@hiroko:/etc/ssl/CA # openssl req -new -key private/cacert.key -out
cacert.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [JP]:               ←そのまま改行
State or Province Name (full name) [hgotoh]:              ←そのまま改行
hgotoh []:              ←そのまま改行
Organization Name (eg, company) [noname]:               ←そのまま改行
Organizational Unit Name (eg, section) [noname]:              ←そのまま改行
Common Name (e.g. server FQDN or YOUR name) []:ca.hgotoh.jp     ←Common Name
は必須。ここでは認証局サーバのFQDN ca.hgotoh.jp を指定。
Email Address []:caadmin@hgotoh.jp              ←メールアドレス

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:              ←そのまま改行。入力しない。
An optional company name []:              ←そのまま改行。入力しない。
root@hiroko:/etc/ssl/CA #
```

自分で署名して証明書を作る

自分の署名要求 /etc/ssl/CA/cacert.csr に対して自分の署名鍵で電子署名を実施し証明書 /etc/ssl/CA/cacert.cert を作る。この例だと10年(3650日)有効な証明書が出来上がる。

```
root@hiroko:/etc/ssl/CA # openssl x509 -in cacert.csr -days 3650 -req -
signkey private/cacert.key -out cacert.cert
Signature ok
subject=/C=JP/ST=hgotoh/O=noname/OU=noname/CN=ca.hgotoh.jp/emailAddress=caad
min@hgotoh.jp
Getting Private key
root@hiroko:/etc/ssl/CA #
```

※自分の署名要求 cacert.csr に対して自分の署名鍵 cacert.key で署名を行っている。本来は外部の認証局に署名要求を送って、その認証局の署名鍵で署名するのね。

必要ならパスフレーズを取り除いた署名鍵 /etc/ssl/CA/private/cacert.key と証明書 /etc/ssl/CA/cacert.cert の結合ファイル /etc/ssl/CA/cacert.pem を作成。

```
root@hiroko:/etc/ssl/CA # mv private/cacert.key private/cacert.withpass.key
root@hiroko:/etc/ssl/CA # openssl rsa -in private/cacert.withpass.key -out
```

```
private/cacert.key
Enter pass phrase for private/cacert.withpass.key:                ←パスフレーズ入力
writing RSA key
root@hiroko:/etc/ssl/CA # cat private/cacert.key cacert.cert > cacert.pem
root@hiroko:/etc/ssl/CA #
```

必要なら認証局の証明書を配布するために /etc/ssl/CA/cacert.der を作成。

```
root@hiroko:/etc/ssl/CA # openssl x509 -inform pem -in cacert.pem -outform der -out cacert.der
root@hiroko:/etc/ssl/CA #

or

root@hiroko:/etc/ssl/CA # openssl x509 -inform pem -in cacert.cert -outform der -out cacert.der
root@hiroko:/etc/ssl/CA #
```

どちらでも同じファイルが作成される。 証明書発行する用意ができ、開局できる。

サーバ証明書発行

メールサーバ mail.hgotoh.jp の証明書を発行してみる。

ファイル	説明
imapserver.key	サーバの秘密鍵。流出したら駄目な奴。
imapserver.cert	オレオレ認証局の電子署名入りサーバ証明書。
imapserver.pem	サーバ秘密鍵とサーバ証明書の結合ファイル。流出しちゃダメなやつ。
とりあえず保持ファイル	**説明**
imapserver.withpass.key	サーバの秘密鍵パスフレーズ付き。流出したら駄目な奴。
使用後は消しとけファイル	**説明**
imapserver.csr	サーバの署名要求。

署名要求のCommonNameで指定したFQDNと、実際にこのサーバ証明書を使うサーバのURLが不一致だったりするとエラーになっちゃう。
サーバ証明書はそのサーバでしか使えないものなので、よそのサーバに持って行ったら駄目よ。

サーバの作業

サーバの秘密鍵と署名要求を作る

サーバ mail.hgotoh.jp の秘密鍵 imapserver.key を作る。

```
root@amane:/etc/ssl/imap # openssl genrsa -aes256 -out imapserver.key 2048
Generating RSA private key, 2048 bit long modulus
.....................................+++
.....................+++
```

```
e is 65537 (0x10001)
Enter pass phrase for imapserver.key:              ←パスフレーズ入力
Verifying - Enter pass phrase for imapserver.key:              ←パスフレーズ入力
root@amane:/etc/ssl/imap #
```

サーバ mail.hgotoh.jp のサーバ証明書の署名要求 imapserver.csr を作る。

```
root@amane:/etc/ssl/imap # openssl req -new -key imapserver.key -out imapserver.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [JP]:              ←認証局のものに合わせる
State or Province Name (full name) [hgotoh]:              ←認証局のものに合わせる
hgotoh []:              ←認証局のものに合わせる
Organization Name (eg, company) [noname]:              ←認証局のものに合わせる
Organizational Unit Name (eg, section) [noname]:              ←認証局のものに合わせる
Common Name (e.g. server FQDN or YOUR name) []:mail.hgotoh.jp   ←Common Name は必須。証明書を配置するサーバのFQDN mail.hgotoh.jp を指定。
Email Address []:imapadmin@hgotoh.jp              ←メールアドレス

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:              ←そのまま改行。入力しない。
An optional company name []:              ←そのまま改行。入力しない。
root@amane:/etc/ssl/imap #
```

認証局へ署名要求を送る

署名要求 imapserver.csr を認証局に送り署名をした証明書 imapserver.cert の提供を受ける。

認証局で署名された証明書をサーバに配置する

必要ならパスフレーズを消したサーバの秘密鍵 imapserver.key とサーバ証明書 imapserver.cert の結合ファイル imapserver.pem を作成。

```
root@amane:/etc/ssl/imap # mv imapserver.key imapserver.withpass.key
root@amane:/etc/ssl/imap # openssl rsa -in imapserver.withpass.key -out imapserver.key
Enter pass phrase for imapserver.withpass.key:              ←パスフレーズ入力
writing RSA key
root@amane:/etc/ssl/imap # cat imapserver.key imapserver.cert >
```

```
imapserver.pem
root@amane:/etc/ssl/imap #
```

サーバの秘密鍵は流出しちゃいけないものなので、imapserver.key、imapserver.pem は配布しちゃだめよ。
証明書と秘密鍵を一つにまとめた形式を入力とするサーバプログラムのために imapserver.pem を作成する。たとえば apache、courier-imap、とか。

認証局の作業

認証局で署名する

認証局に送られてきたサーバの署名要求 imapserver.csr を処理してサーバ証明書 imapserver.cert を作る。 特に指定しないとデフォルトの1年が有効期間となる。

```
root@hiroko:/etc/ssl/ # openssl ca -config /etc/ssl/openssl.cnf -policy
policy_anything -in /etc/ssl/tmp/imapserver.csr -out
/etc/ssl/tmp/imapserver.cert
Using configuration from /etc/ssl/openssl.cnf
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 4 (0x4)
        Validity
            Not Before: Apr 10 20:17:27 2016 GMT
            Not After : Apr 10 20:17:27 2017 GMT
        Subject:
            countryName               = JP
            stateOrProvinceName       = hgotoh
            organizationName          = noname
            organizationalUnitName    = noname
            commonName                = mail.hgotoh.jp
            emailAddress              = imapadmin@hgotoh.jp
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx
            X509v3 Authority Key Identifier:
DirName:/C=JP/ST=hgotoh/O=noname/OU=noname/CN=ca.hgotoh.jp/emailAddress=caad
min@hgotoh.jp
                serial:xx:xx:xx:xx:xx:xx:xx:xx

Certificate is to be certified until Apr 10 20:17:27 2017 GMT (365 days)
Sign the certificate? [y/n]:y
```

```
1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
root@hiroko:/etc/ssl/ #
```

証明書の返送

出来上がったサーバ証明書 imapserver.cert を要求元のサーバに送る。

クライアント証明書発行

メールサーバアクセスで使うクライアント証明書を発行する。

ファイル	説明
imapclient.key	クライアントの秘密鍵。流出注意なやつ。
imapclient.cert	オレオレ認証局の電子署名入りクライアント証明書。
imapclient.pem	クライアント証明書と認証局証明書の結合ファイル。流出注意な奴。
imapclient.p12	imapclient.pemをPKCS12形式にしたもの。流出注意な奴。
とりあえず保持ファイル	**説明**
imapclient.withpass.key	クライアントの秘密鍵パスフレーズ付き。流出注意な奴。
使用後は消しとけファイル	**説明**
imapclient.csr	クライアントの署名要求。

クライアントの秘密鍵は流出しちゃいけない。 同じ証明書をサブPCでも使いたい、とかになるとimapclient.pemをPKCS12形式で持ち出す事になる。もちろんサブPC外に流出したらアウトなのでよく考えて。

サブPC用にクライアント証明書を作ればサブPCから流出があってもサブPCの範囲で影響はクローズするから、可能ならクライアント毎に作った方がいいかもね。

クライアント証明書って具体的に何さ？

クライアント証明書はもう少し突っ込むと、ユーザ証明書とかコンピュータ証明書とかに分かれる。 ここで説明するクライアント証明書を Thunderbirdの証明書マネージャからインポートしてみると、

- あなたの証明書
- 認証局証明書

にインポートがされている imapclient.cert が " あなたの証明書 " で有効になっているみたい cacert.cert が " 認証局証明書 " で有効になっているみたい。

…ユーザ証明書ってことでいいのかな？

クライアントの作業

クライアントの秘密鍵と署名要求を作る

クライアント h.kuma の秘密鍵 imapclient.key を作る。

```
$ openssl genrsa -aes256 -out imapclient.key 2048
Generating RSA private key, 2048 bit long modulus
..............................+++
.....................................................+++
e is 65537 (0x10001)
Enter pass phrase for imapclient.key:              ←パスフレーズ入力
Verifying - Enter pass phrase for imapclient.key:              ←パスフレーズ入力
$
```

クライアント h.kuma の署名要求 imapclient.csr を作る。

```
$ openssl req -new -key imapclient.key -out imapclient.csr
Enter pass phrase for imapclient.key:
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:JP              ←認証局のものに合わせる
State or Province Name (full name) [Some-State]:hgotoh              ←認証局のものに合わせる
Locality Name (eg, city) []:hgotoh              ←認証局のものに合わせる
Organization Name (eg, company) [Internet Widgits Pty Ltd]:noname
←認証局のものに合わせる
Organizational Unit Name (eg, section) []:noname              ←認証局のものに合わせる
Common Name (e.g. server FQDN or YOUR name) []:h.kuma              ←Common Nameは必須。ここでは自分の名前を指定。
Email Address []:h.kuma@hgotoh.jp              ←メールアドレス


Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:              ←そのまま改行。入力しない。
An optional company name []:              ←そのまま改行。入力しない。
$
```

認証局へ署名要求を送る

署名要求 imapclient.csr を認証局に送り署名をした証明書 imapclient.cert の提供を受ける。

認証局で署名された証明書を配布可能にする

必要ならパスフレーズを消したクライアントの秘密鍵 imapclient.key を作る。

```
$ mv imapclient.key imapclient.withpass.key
$ openssl rsa -in imapclient.withpass.key -out imapclient.key
Enter pass phrase for imapclient.withpass.key:              ←パスフレーズ入力
writing RSA key
$
```

必要ならクライアントの秘密鍵 imapclient.key と クライアント証明書 imapclient.cert の結合ファイル imapclient.pem を作る。

```
$ cat imapclient.key imapclient.cert > imapclient.pem     ←imapclient.certは
認証局証明書！
$
```

必要ならPKCS12形式の配布ファイル imapclient.p12 を作成する。

```
$ openssl pkcs12 -export -in imapclient.pem -out imapclient.p12 -name
"kuma's key"
Enter Export Password:              ←配布時に問われる"kuma's key"に対するパスワー
ドを入力。パスフレーズじゃないよ。
Verifying - Enter Export Password:                ←配布時に問われる"kuma's key"に
対するパスワードを入力。パスフレーズじゃないよ。
$
```

imapclient.p12 はメールクライアントなどに証明書をインポートする際使用する。

認証局の作業

認証局で署名する

認証局に送られてきたクライアント h.kuma の署名要求 imapclient.csr を処理してクライアント証明書 imapclient.cert を作る。 特に指定しないとデフォルトの1年が有効期間となる。

```
root@hiroko:/etc/ssl # openssl ca -config /etc/ssl/openssl.cnf -policy
policy_anything -in /etc/ssl/tmp/imapclient.csr -out
/etc/ssl/tmp/imapclient.cert
Using configuration from /etc/ssl/openssl.cnf
Check that the request matches the signature
Signature ok
Certificate Details:
        Serial Number: 5 (0x5)
        Validity
            Not Before: Apr 10 19:35:39 2016 GMT
            Not After : Apr 10 19:35:39 2017 GMT
        Subject:
            countryName               = JP
```

```
            stateOrProvinceName       = hgotoh
            organizationName          = noname
            organizationalUnitName    = noname
            commonName                = h.kuma
            emailAddress              = h.kuma@hgotoh.jp
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
                xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx:xx
            X509v3 Authority Key Identifier:
DirName:/C=JP/ST=hgotoh/O=noname/OU=noname/CN=ca.hgotoh.jp/emailAddress=caad
min@hgotoh.jp
                serial:xx:xx:xx:xx:xx:xx:xx:xx

Certificate is to be certified until Apr 10 19:35:39 2017 GMT (365 days)
Sign the certificate? [y/n]:y


1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
root@hiroko:/etc/ssl #
```

証明書の返送

出来上がったクライアント証明書 imapclient.cert を要求元のクライアントに送る。

証明書の期限延長

以下はクライアント証明書の例。サーバ証明書でも同じ。

変更と変更後の証明書

"-days 1460" は365 × 4 = 1460 で4年の指定。2019-01-01開始なのは更新日が2019-01-01だから。※更新前は2018-05-04開始だった

```
$ openssl x509  -days 1460 -signkey imapclient.key -in imapclient.cert -out
imapclient.cert.new
Getting Private key
$ mv imapclient.cert imapclient.cert.old
$ mv imapclient.cert.new imapclient.cert
$ openssl x509 -noout -text -in  imapclient.cert
Certificate:
    Data:
```

```
        Version: 3 (0x2)
        Serial Number: 12 (0xc)
    Signature Algorithm: sha256WithRSAEncryption
        Issuer: C=JP, ST=hgotoh, L=hgotoh, O=noname, OU=noname,
CN=h.kuma/emailAddress=hgotoh@hgotoh.jp
        Validity
            Not Before: Jan  1 03:57:49 2019 GMT
            Not After : Dec 31 03:57:49 2022 GMT
        Subject: C=JP, ST=hgotoh, L=hgotoh, O=noname, OU=noname,
CN=h.kuma/emailAddress=hgotoh@hgotoh.jp
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                Public-Key: (2048 bit)
                Modulus:
                    □
                    □
                    □
$
```

技術資料, openssl, CA証明書, **サーバ**証明書, **クライアント**証明書, SSL証明書