実践で学ぶ

一歩進んだサーバー構築・運用術

written by 仙石 浩明

第6回 「stone(後編)」

アプリケーション・レベルでTCPやUDPのパケットを中継するソフト「stone」の紹介の2回目です。今回はstoneを使って，SSLで暗号化したしたデータを送受信するための具体的な手順を解説します。これで安心してメールを転送できるようになります。

会社設立に向けて走り出してから4カ月，ようやく創業メンバーがそろってきました*1。当初はほとんどの仕事を自分一人でこなす必要があったのですが，人数が増えるに従って役割分担が可能になり，少しゆとりが出てきました。

というわけで一週間休暇*2を取ってハワイへ行ってきました。当初の予定では，会社を設立して一段落したころに行くつもりで6月下旬を選んだのですが，諸般の事情で設立が遅れています。まあ，よくあることですね。

休暇中ではあっても，朝晩2度のメール・チェックは欠かせません。日経Linux編集部から先月号の原稿の校正用原稿がファクシミリで送られてきたりして，SOHO in Hawaiiの様相を呈していました。

私が出先でメールを読むときは，現地のプロバイダへ接続した後，自宅のマシンへSSL(セキュア・ソケット・レイヤー)で暗号化したUUCP over TCP/IP(すなわち，UUCP over SSL)で接続し，メールを転送しています。メールを一気に送受信できるので，接続時間を最小にできます*3。また，SSLで暗号化しているので仕事上の機密事項を含んだメールでも安心です*4。

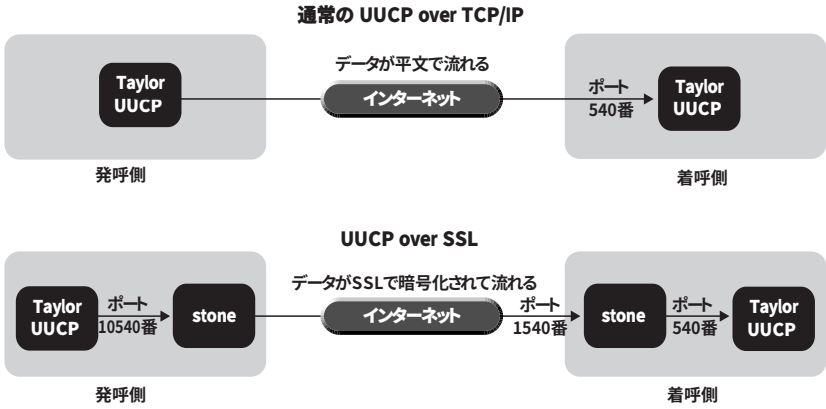

図1 stoneを使用した場合のUUCP over SSL
「Taylor UUCP」と呼ぶUUCPの最新バージョンとstoneを使って，データを暗号化して送信できる。

UUCP over SSL

stoneでSSLを使うには，まずOpenSSL*5をインストールする必要があります。ここではディレクトリの/usr/local/ssl以下に，OpenSSL 0.9.5aがインストールされているものとします。

また，UUCP over SSLを利用するには，UUCP over TCP/IPが行える状態である

*1 皆さん，前職を急には辞められないので，なかなかすぐにはメンバーがそろわないのです。
*2 五月連休を休まず働いたので，その代休という扱いです。
*3 米国の場合，フラットレート（ホテルの部屋の電話を使う場合に一回75セントで何時間でも接続可）なので，接続時間が短くてもあまりメリットはありません。現地プロバイダでなく，海外ローミング・サービスを利用する場合だと，接続時間が短いのは大きなメリットでしょう。
*4 会社から自宅のマシンあてのメールは，通常のSMTPではなく，SSLで暗号化して転送しています。
*5 OpenSSLについては，http://www.openssl.org/を参照してください。

```
-l -dd
-f 2
-o 99 -g 99

asao.gcd.org:1540/ssl   localhost:10540
```

図2　stoneをSSL暗号通信を要求するクライアントとして動作させるための設定ファイル例

```
call-login      *
call-password   *
time            Any
commands        rnews rmail

system          gcd
alias           gcd.org
port            SSLasao
address         localhost
```

図3　sysファイルの例
ポート「SSLasao」は，stone の受けポート10540番。

```
port            SSLasao
type            tcp
service         10540
```

.図4　portファイルの例

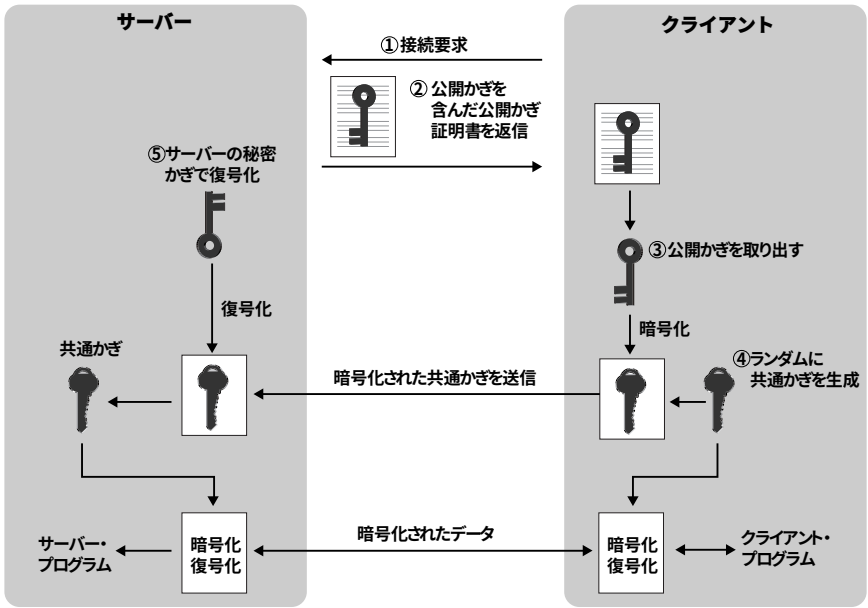

図5　共通かぎを使って通信を暗号化する仕組み

ことが必要です*6。stoneを使ったUUCP over SSLは，**図1**のような構成になります。

発呼側の設定

stoneをSSL暗号通信を要求するクライアントとして動作させるには，stoneの引数の「あて先ホスト・ポート」に「/ssl」を付けます。言い換えると，中継を設定する形式の「あて先ホスト：あて先ポート 受けポート」のあて先ポートに「/ssl」を付けます。例えば，stoneの設定ファイルを**図2**のようにすることにより，ポート10540番に届いたUUCPパケットをSSLで暗号化して，asao.gcd.orgのポート1540番に転送します。

UUCPの設定ファイル「sys」（**図3**）および「port」（**図4**）において，接続先ホスト「asao.gcd.org」ポート540番（UUCP over TCP/IP の標準のポート番号）の代わりに，stoneの受けホスト（localhost）・ポート（10540番）を設定します。

公開かぎと秘密かぎ

Web用の暗号化方式として有名なSSLは，暗号通信に先だってまずRSA暗号*7を使ってサーバー・クライアント間で共通かぎ*8を交換し，この共通かぎを使って通信を暗号化します。その仕組みを簡単に**図5**に示しました。より詳しく知りたい方は，日本ベリサインのホームページのFAQ（http://www.verisign.co.jp/repositor/faq/SSL/）を参照すると良いでしょう。

クライアントから接続要求（1）があると，サーバーは公開かぎ証明書（2）を返します。クライアントは，この公開かぎ証明書からサーバーの公開かぎ（3）を取り出します。次にクライアントはランダムな共通かぎ（4）を生成し，この共通かぎ（4）をサーバーの公開かぎ（3）を使って暗号化してサーバーへ送ります。

サーバーは，自身で持っている秘密かぎ（5）を使って，クライアントから送られてきた共通かぎ（3）を復号化します。以上で，クライアントとサーバーの双方が同じ共通かぎを持つことになりので，この共通かぎを使って暗号通信を行います（6）。

したがってSSL通信を行うサーバーは，公開かぎ証明書（2）と秘密かぎ（5）を持っている必要があります*9。公開かぎ証明

書は，日本ベリサインなどの認証局（Certification Authority, CA）に発行してもらう*10ほか，後述するように自前の認証局をでっち上げれば，自分で発行することもできます。

秘密かぎの作成と登録申請書の作成

認証局に公開かぎ証明書を発行してもらうために必要なのが，登録申請書です。OpenSSLのreqコマンドを使って，秘密かぎと登録申請書を生成する例を**図6**に示します。登録申請書には，サーバーが属する組織名や所在地，メール・アドレスが含まれますから,それらのデータを入力しています。入力し終えると，秘密かぎのファイル「key.pem」と登録申請書ファイル「newreq.pem」がカレント・ディレクトリに作成されます。

ここで注意することは，生成するRSA暗号かぎのビット長です。デフォルトでは，1024ビットになっているのですが，国内で使われているWebブラウザの多くは，512ビットを超えるかぎを扱うことができません。SSL通信の対象として不特定多数のWebブラウザを想定しているならば，かぎ長を512ビットにしておいた方が無難です。

かぎ長を512ビットにするには，reqコマンドの引数に「-newkey rsa:512」オプションを付け加えるか，設定ファイル「/usr/local/ssl/openssl.cnf」において，「default_bits」の行を，

```
default_bits            = 512
```

に変更します。図6でreqコマンドの出力の2行目が，「Generating a 512 bit RSA private key」となっていることを確認してください。

秘密かぎ「key.pem」はサーバー・プログラム以外から読み出せないようにパーミッションを変更し，/usr/local/ssl/privateディレクトリに，ファイル名「www.pem」で保管します(**図7**)。

生成した登録申請書「newreq.pem」の例を**図8**に示します。これを認証局に送れ

```
% openssl req -new -nodes -keyout key.pem -out newreq.pem ↵
Using configuration from /usr/local/ssl/openssl.cnf
Generating a 512 bit RSA private key
..............................+++++
.....+++++
writing new private key to 'key.pem'

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]: ↵

State or Province Name (full name) [Some-State]:Kanagawa ↵
Locality Name (eg, city) []:Kawasaki ↵
Organization Name (eg, company) [Internet Widgits Pty Ltd]:GCD ↵
Organizational Unit Name (eg, section) []: ↵

Common Name (eg, YOUR name) []:www.gcd.org ↵
Email Address []:webmaster@gcd.org ↵

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []: ↵

An optional company name []: ↵
```

↵このマークで改行

図6　OpenSSLのreqコマンドを使って秘密鍵と登録申請書を生成する例

```
# chown root.sslkey key.pem ↵
# mv key.pem /usr/local/ssl/private/www.pem ↵
```

図7　秘密かぎを保管する

```
-----BEGIN CERTIFICATE REQUEST-----
MIIBMzCB3gIBADB5MQswCQYDVQQGEwJKUDERMA8GA1UECBMIS2FuYWdhd2ExETAP
BgNVBAcTCEthd2FzYWtpMQwwCgYDVQQKEwNHQ0QxFDASBgNVBAMTC3d3dy5nY2Qu
b3JnMSAwHgYJKoZIhvcNAQkBFhF3ZWJtYXN0ZXJAZ2NkLm9yZzBcMA0GCSqGSIb3
DQEBAQUAA0sAMEgCQQDev3CZ7ldEMJlPBYKxJhty1EdBl+HFs0a6r5oahBuqSYE3
nSUmwfP0pinBWkZ9+WQflwUHRe1lK5sqfZNv9pTRAgMBAAGgADANBgkqhkiG9w0B
AQQFAANBAMq02r3LCWP6IDZ8MwWxfVcC2u9C2A2BWgGxnp0EPFLIvK+Tohql7J+M
/rHr+/4fZwc2vVRBq9L4ft7c31/4L9Q=
-----END CERTIFICATE REQUEST-----
```

図8　登録申請書「newreq.pem」の例

＊6　本連載の第4回「UUCP の活用」(2000年7月号) を参照してください。
＊7　RSA暗号代表的な公開かぎ暗号。米RSA Security社の暗号化技術である。
＊8　共通かぎは，暗号化と復号化に同じかぎを用いる対称暗号方式におけるかぎを指す。
＊9　図5から分かるように，クライアント側は事前にかぎを用意する必要がありません。
＊10　日本ベリサインの場合は年間12万6000円かかります。

```
[ CA_default ]

dir            = /usr/local/ssl/CA      # Where everything is kept
```

図9　/usr/local/ssl/openssl.cnfの設定例

```
CATOP=/usr/local/ssl/CA
```

図10　/usr/local/ssl/misc/CA.shの設定例

```
# /usr/local/ssl/misc/CA.sh -newca ↵
CA certificate filename (or enter to create)

Making CA certificate ...
Using configuration from /usr/local/ssl/openssl.cnf
Generating a 512 bit RSA private key
...........+++++
.............+++++
writing new private key to '/usr/local/ssl/CA/private/cakey.pem'
Enter PEM pass phrase:XXXXXXXXXXXXXXX（伏せ字） ↵
Verifying password - Enter PEM pass phrase:XXXXXXXXXXXXXXX（伏せ字） ↵

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]: ↵

State or Province Name (full name) [Some-State]:Kanagawa ↵
Locality Name (eg, city) []:Kawasaki ↵
Organization Name (eg, company) [Internet Widgits Pty Ltd]:GCD ↵
Organizational Unit Name (eg, section) []: ↵

Common Name (eg, YOUR name) []:Hiroaki Sengoku ↵
Email Address []:sengoku@gcd.org ↵
```

図11　/usr/local/ssl/misc/CA.sh -newcaを実行してパスフレーズ，認証局の名称，所在地，メール・アドレスを入力する

```
# /usr/local/ssl/misc/CA.sh -sign ↵
Using configuration from /usr/local/ssl/openssl.cnf
Enter PEM pass phrase:XXXXXXXXXXXXXXX（伏せ字） ↵
Check that the request matches the signature
Signature ok
The Subjects Distinguished Name is as follows
countryName           :PRINTABLE:'JP'
stateOrProvinceName   :PRINTABLE:'Kanagawa'
localityName          :PRINTABLE:'Kawasaki'
organizationName      :PRINTABLE:'GCD'
commonName            :PRINTABLE:'www.gcd.org'
emailAddress          :IA5STRING:'webmaster@gcd.org'
Certificate is to be certified until Jul  2 04:36:48 2001 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
```

図12　公開かぎ証明書「newcert.pem」を作成する(その1)

ば[*11]，公開かぎ証明書を得ることができます。

認証局の作成

ベリサインの場合，法人格のない団体への公開かぎ証明書の発行は行っていないので，個人で立ち上げているサーバーの場合は自前で認証局を立てた方が簡単です。

認証局の役割は，

(1) 申請内容の確認
**　申請者が当該ドメイン名の正規の所有者であることを確認する**
(2) 公開かぎ証明書の発行
**　認証局の秘密かぎで電子署名を施す**

ですが，自前で認証局を立てる場合は，申請者は自分自身あるいは周囲の人であることがほとんどでしょうから，(1)は確認するまでもないでしょう。

(2)を行うためには，認証局の秘密かぎが必要になります。まず，認証局のためのディレクトリを決めます。ここでは，/usr/local/ssl/CAとします。このディレクトリ名を/usr/local/ssl/openssl.cnfの「dir =」の行と（**図9**），/usr/local/ssl/misc/CA.shの「CATOP=」の行に設定します（**図10**）。

次に

```
/usr/local/ssl/misc/CA.sh -newca
```

を実行します。認証局の秘密かぎを守るためのパスフレーズを入力してください。そして，認証局の名称，所在地，メール・アドレスを入力します（**図11**）。

これで，/usr/local/ssl/CA/private/cakey.pemに認証局の秘密かぎが，/usr/local/ssl/CA/cacert.pemに認証局の証明書がそれぞれ作成されます。

公開かぎ証明書の発行

次に, 登録申請書「newreq.pem」のあるディレクトリで,「/usr/local/ssl/misc/CA.sh -sign」を実行します(**図12**)。これでカレント・ディレクトリに, 公開かぎ証明書「newcert.pem」が作成されます。これを/usr/local/ssl/certsディレクトリに,「www.pem」というファイル名で保管します(**図13**)。

着呼側の設定

stoneをSSL暗号通信を受け付けるサーバーとして動作させるには, -zオプションで秘密かぎと公開かぎ証明書のパスを設定してから,「受けポート」に「/ssl」を付けます。例えば, ポート1540番でUUCP over SSLを受け付けるには, **図14**のように実行します。図14において,「-z key=ファイル名」が秘密かぎのパスの設定,「-z cert=ファイル名」が公開かぎ証明書のパスの設定です。ただし, localhostのポート540番(uucpポート)でUUCP over TCP/IPを受け付ける設定になっているものとします。

stone利用の実例

GCD*12で, 実際にstoneがどのように使われているかを紹介します。GCDのゲートウエイ(asao.gcd.org)では, stoneの設定ファイル「/etc/rc.d/stone」を**図15**のように記述しておき, **図16**のシェル・スクリプト*13でstoneを呼び出しています

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 1 (0x1)
        Signature Algorithm: md5WithRSAEncryption
        Issuer: C=JP, ST=Kanagawa, L=Kawasaki, O=GCD,
        CN=Hiroaki Sengoku/Email=sengoku@gcd.org
        Validity
            Not Before: Jul  2 04:36:48 2000 GMT
            Not After : Jul  2 04:36:48 2001 GMT
        Subject: C=JP, ST=Kanagawa, L=Kawasaki,
        O=GCD, CN=www.gcd.org/Email=webmaster@gcd.org
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (512 bit)
                Modulus (512 bit):
                    00:de:bf:70:99:ee:57:44:30:99:4f:05:82:b1:26:
                    1b:72:d4:47:41:97:e1:c5:b3:46:ba:af:9a:1a:84:
                    1b:aa:49:81:37:9d:25:26:c1:f3:f4:a6:29:c1:5a:
                    46:7d:f9:64:1f:97:05:07:45:ed:65:2b:9b:2a:7d:
                    93:6f:f6:94:d1
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Comment:
                OpenSSL Generated Certificate
            X509v3 Subject Key Identifier:
               B1:C5:6B:5A:83:3D:B4:42:13:37:EC:CC:C3:FF:14:C5:86:AB:19:AC
            X509v3 Authority Key Identifier:
              keyid:80:7B:D3:A7:F1:D4:33:46:1F:5E:FB:23:70:30:87:53:33:A5
              :EB:4F
                DirName:/C=JP/ST=Kanagawa/L=Kawasaki/O=GCD/CN=
                Hiroaki Sengoku/Email=sengoku@gcd.org
                serial:00

    Signature Algorithm: md5WithRSAEncryption
        3e:c2:c1:e7:95:79:25:c9:32:dd:48:d9:10:10:3e:6f:50:47:
        bd:f2:a8:62:7e:dd:62:a7:78:21:4e:3b:af:60:d9:49:e2:25:
        6f:93:b3:c7:c0:50:99:ab:ac:3c:69:ba:87:2d:ed:ab:f4:df:
        84:81:f0:c6:e3:85:57:26:92:8b
-----BEGIN CERTIFICATE-----
MIIC6zCCApWgAwIBAgIBATANBgkqhkiG9w0BAQQFADB7MQswCQYDVQQGEwJKUDER
MA8GA1UECBMIS2FuYWdhd2ExETAPBgNVBAcTCEthd2FzYWtpMQwwCgYDVQQKEwNH
Q0QxGDAWBgNVBAMTD0hpcm9ha2kgU2VuZ29rdTEeMBwGCSqGSIb3DQEJARYPc2Vu
Z29rdUBnY2Qub3JnMB4XDTAwMDcwMjA0MzY0OFoXDTAxMDcwMjA0MzY0OFoweTEL
MAkGA1UEBhMCSlAxETAPBgNVBAgTCEthbmFnYXdhMREwDwYDVQQHEwhLYXdhc2Fr
aTEMMAoGA1UEChMDR0NEMRQwEgYDVQQDEwt3d3cuZ2NkLm9yZzEgMB4GCSqGSIb3
DQEJARYRd2VibWFzdGVyQGdjZC5vcmcwXDANBgkqhkiG9w0BAQEFAANLADBIAkEA
3r9wme5XRDCZTwWCsSYbctRHQZfhxbNGuq+aGoQbqkmBN50lJsHz9KYpwVpGfflk
H5cFB0XtZSubKn2Tb/aU0QIDAQABo4IBBDCCAQAwCQYDVR0TBAIwADAsBglghkgB
hvhCAQ0EHxYdT3BlblNTTCBHZW5lcmF0ZWQgQ2VydGlmaWNhdGUwHQYDVR0OBBYE
FLHFa1qDPbRCEzfszMP/FMWGqxmsMIGlBgNVHSMEgZ0wgZqAFIB706fx1DNGH177
I3Awh1MzpetPoX+kfTB7MQswCQYDVQQGEwJKUDERMA8GA1UECBMIS2FuYWdhd2Ex
ETAPBgNVBAcTCEthd2FzYWtpMQwwCgYDVQQKEwNHQ0QxGDAWBgNVBAMTD0hpcm9h
a2kgU2VuZ29rdTEeMBwGCSqGSIb3DQEJARYPc2VuZ29rdUBnY2Qub3JnggEAMA0G
CSqGSIb3DQEBBAUAA0EAPsLB55V5Jcky3UjZEBA+b1BHvfKoYn7dYqd4IU47r2DZ
SeIlb5Ozx8BQmausPGm6hy3tq/TfhIHwxuOFVyaSiw==
-----END CERTIFICATE-----
Signed certificate is in newcert.pem
```

図12　公開かぎ証明書「newcert.pem」を作成する(その2)

*11　14日間無料のテスト用公開かぎ証明書が次のURLから取得できるので, とりあえず試してしてみたい方は利用してみると良いでしょう。http://www.verisign.co.jp/landing/go_vsi.htm

*12　筆者が運営する任意団体です。個人で引いたOCNエコノミーの費用の一部を賄うために, 多岐に渡るサービスを提供しています。本連載では, このGCDでのサーバー構築, 運用法を例に挙げています。

*13　説明のため, 実際に使用しているシェル・スクリプトより簡略化しています。

```
# mv newcert.pem /usr/local/ssl/certs/www.pem ↵
```

図13　作成した「newcert.pem」を「www.pem」というファイル名で/usr/local/ssl/certsディレクトリに保管する

```
stone -z key=/usr/local/ssl/private/www.pem \
      -z cert=/usr/local/ssl/certs/www.pem \
      localhost:uucp 1540/ssl &
```

図14　stoneをSSL暗号通信を受け付けるサーバーとして動作させるための実行例

```
 1  #define LOCALS      192.168.1.0/255.255.255.0
 2
 3  #if HOST_asao_gcd_org
 4  -l -dd
 5  -f 2
 6  -z key=/usr/local/ssl/private/www.pem
 7  -z cert=/usr/local/ssl/certs/www.pem
 8  -o 65534 -g 120
 9  -a /var/log/stone.log
10  -t /var/root
11
12  $INNER:uucp         $OUTER:1540/ssl                          --
13  localhost:http      $OUTER:443/ssl                           --
14  $OUTER:telnet       $OUTER:10023/ssl                         --
15  $INNER:smtp         $OUTER:465/ssl          mx.klab.org      --
16  mx.klab.org:465/ssl localhost:10025                          --
17  $OUTER:pop/apop     $INNER:pop              LOCALS           --
18  #endif
```

図15　GCDのゲートウエイにおけるstoneの設定ファイル「/etc/rc.d/stone」

```
#!/bin/sh
INNER="asaogt.gcd.org"
OUTER="asaogw.gcd.org"
export INNER OUTER
/usr/local/sbin/stone -C /etc/rc.d/stone &
```

図16　stoneを呼び出すためのシェル・スクリプトの例

```
openssl x509 -outform DER -out cacert.der < /usr/local/ssl/CA/cacert.pem
```

図17　認証局の証明書「cacert.der」を作成する

図15の設定ファイル中,「$OUTER」など「$」で始まる語は環境変数の参照です。「$OUTER」は外向けのグローバル・アドレス（asaogw.gcd.org),「$INNER」は内部LAN向けのプライベート・アドレス（asaogt.gcd.org）です。

設定ファイルはstoneで読み込まれる前にCプリプロセッサで処理されるので, C言語のソース・プログラムと同様に「#define」でマクロを設定できます。そしてstoneが実行されたマシンのホスト名が「asao.gcd.org」のときは,「#define HOST_asao_gcd_org 1」が自動的に設定されるため,「#if HOST_asao_gcd_org」～「#endif」部分が有効になります。

設定ファイルの中の8行目の文は, stoneを実行するユーザーIDとグループIDを指定しています。グループIDの120番はsslkeyで, 前述したように秘密かぎを読めるのはrootとsslkeyグループだけですから, このように設定しています。

10行目は, stoneを犠牲パーティションで実行するための指定です。stoneはパケットの受け付け動作に入る前に, chroot /var/rootを行います。もし万一, stoneにセキュリティ・ホールがあったとしても, 侵入者の自由になるのは, /var/root以下だけに限定できます。

12行目以降は, 中継の設定です。前述したようにUUCP over SSLのための設定を行っています。

13行目は, WebサーバーをSSL対応にするための設定です。

これで, Webブラウザでhttps://www.gcd.org/にアクセスすると, stoneがSSL通信を復号化し, localhostのポート80番に中継します。

あらかじめWebブラウザに（自前でででっち上げた）認証局の証明書をインストールしておくと良いでしょう。**図17**のように, OpenSSLのx509コマンドを実行することにより, DER形式の証明書「cacert.der」を作成し, Webブラウザでインポートします。

14行目はSSLに対応したtelnetクライアントで接続するための設定です。

15行名と16行目は, MTA間でSMTP通信を暗号化するための設定です。双方のMTAは相手のSMTPポートに接続する代わりに, stoneの受けポートに接続するように設定してあります。この設定により, klab.org,とgcd.orgの間では外部に情報が漏れることなく, メールを送受信できるようになります。

17行目は, GCDの内部向けにPOPサービスを提供するための設定です。asao.gcd.orgにAPOPサーバーはありますが, POPサーバーはありません。その代わり内部LAN上のPOPクライアントからのアクセスを, stoneがAPOPに変換して中継しています。