OpenSSL と Apache を用いた

CamelliaでSSL通信できるSSLサーバの構築方法 V2.1

マルチプラットフォーム型共通鍵ブロック暗号

NTT 情報流通プラットフォーム研究所　2009/3/19 版

この資料は、共通鍵ブロック暗号“Camellia”のユーザーズガイドです。Camellia は、NTT と三菱電機が共同で開発した暗号で、ソフトウェア実装、ハードウェア実装を問わず、さまざまな環境で世界トップクラスの安全性・処理性能を実現します。その Camellia が OpenSSL に搭載されたことで、手軽に利用できるようになりました。OpenSSL の次期メジャーバージョンアップ版である OpenSSL0.9.9 系では、Camellia がデフォルトで有効になる予定です。OpenSSL0.9.8 系では、まだ Camellia がデフォルトで有効になっていませんが、Apache Lounge※1 では、OpenSSL のディストリビュータに対して OpenSSL0.9.8 系でもヨーロッパや日本では有効にすることを推奨しています。

本ガイドでは、暗号利用を検討されているシステム構築担当者やシステム開発担当者の方を対象として、OpenSSL＋Apache（1.x 系または 2.x 系）を用いた SSL サーバの構築方法を説明します。本ガイドを参考に、Camellia で SSL 通信ができる SSL サーバを構築してみて下さい。

既存 OpenSSL の Camellia 有効確認

OS と一緒にすでに OpenSSL がインストールされている場合があるため、インストール済みの OpenSSL で Camellia が有効かどうかを確認します。SSL サーバを構築しようとしている OS と OS のバージョンを確認し、**Camellia が有効な OpenSSL を同梱している OS** に含まれているかを確認してください。含まれていない場合は、以下のコマンドで確認をしてください。

◆OpenSSL で使用できる暗号の確認

```
$ openssl ciphers
```

コマンドの実行結果として表示された暗号の中に Camellia が含まれていない場合は、Camellia が有効になっていません。OpenSSL のインストールをする必要がありますので、次章の **OpenSSL のインストール** から実行してください。

Camellia が含まれている場合には、Camellia が有効になっていますので、Apache のバージョンを確認してください。Apache のバージョンが 2.x 系であった場合には、**Camellia を最優先で選択させるための設定** から進めてください。

Apache のバージョンは以下のコマンドで確認をすることができます。

◆Apache のバージョンの確認

```
$ httpd -v
```

※1：（http://www.apachelounge.com/forum/viewtopic.php?t=1992）

OpenSSL のインストール

Apache1.x 系、Apache2.x 系とも、はじめに Camellia が搭載されている OpenSSL をインストールします。

OpenSSL のダウンロード

Camellia は OpenSSL0.9.8c 以降に搭載されていますので、以下のサイトから最新(2008 年 8 月現在、0.9.8h が最新)のアーカイブを入手します。

■OpenSSL:　http://www.openssl.org/source/

※次節以降は、0.9.8e を例として記述します。

アーカイブの展開

OpenSSL のアーカイブを展開します。

◆OpenSSL のアーカイブを展開

```
$ tar xvfz openssl-0.9.8e.tar.gz
```

※Windows でダウンロードした場合のサーバへのファイル転送手順等については省略します。

コンパイルとインストール

次の手順で、OpenSSL のコンパイルおよびインストールを行います。OpenSSL0.9.8 系では、コン

パイル時に Camellia を有効にするためのオプションが必要となります。

◆OpenSSL のコンパイルとインストール

```
$ cd openssl-0.9.8e          ← OpenSSL の展開先
$ ./config enable-camellia   ← Camellia のコンパイルオプションをオンにする
  --prefix=/usr/local/ssl shared
$ make depend
$ make
$ su
# make install               ← root 権限でインストール
```

インストール結果の確認

インストールが正常に終了したならば、次のコマンドで Camellia が利用可能であることを確認します(Camellia の名称が表示されていれば OK)。

◆Camellia が利用可能か確認

```
# cd /usr/local/ssl/bin
# ./openssl ciphers            ← 今回インストールした OpenSSL*1 を実行
        :（省略）
..:DHE-RSA-CAMELLIA-256-SHA:DHE-DSS-...
```

*1：OS に同梱されていて、すでにインストールされている OpenSSL や Apache にパスが張られている場合がありますので、上記のように、今回インストールした OpenSSL を指定してコマンドを実行して下さい。

秘密鍵とサーバ証明書の作成

SSL 通信を行なうために必要な秘密鍵とサーバ証明書を作成します。

秘密鍵の作成

パスワードで保護された秘密鍵を、Camellia 等の暗号を使って作成する方法を次に示します。

この場合、Apache を起動するたびにパスワードを確認してきますが、このことでシステムをより堅牢にすることができます。

◆Camellia で暗号化された 2048 ビット長の秘密鍵の作成

```
# cd /usr/local/ssl/bin                    ← 作成される秘密鍵
# ./openssl genrsa -camellia128 2048 > server.key
        :（省略）                          ← パスフレーズ(パスワード)入力
Enter PEM pass phrase:
Verifying password - Enter PEM pass phrase:
                                           ← パスフレーズ再入力
```

サーバ証明書の作成

公的な SSL サーバを構築する場合は、CSR(証明書署名要求)を作成し、ベリサイン社等の信頼された証明書発行機関によって署名された証明書を発行してもらいます。

本ガイドでは、Camellia の動作確認を目的としているため、公的認証局の署名ではなく、手軽に用意できる“自己署名付き証明書”を作成することにします。その結果、クライアントからサーバにアクセスすると、信頼されたサイトではない旨の警告が表示されますが、問題ありません。

自己署名付き証明書を作成するコマンドの例を以下に示します。

◆自己署名付き証明書の作成

```
# ./openssl req -new -days 365 -key server.key   ← 作成済みの秘密鍵
    -x509 -out server.crt                        ← 作成されるサーバ証明書
```

コマンドパラメータの“-days 365”は証明書の有効期限が 1 年間であることを表していて、省略した場合は 30 日間となります。

本コマンドを入力すると、サーバ情報として、国の識別子(「JP」)、行政区域名、組織名、サーバ名等を入力するよう求められますので、メッセージに従って入力して下さい。

以上で、秘密鍵に対応した証明書(公開鍵と付加情報)が作成されました。

Apache＋mod_SSL のインストール

Apache と mod_SSL をインストールします。ただし、Apache2.x 系では mod_SSL が既に組み込まれているため、Apache だけをインストールします。

Apache1.x 系の場合

Apache と mod_SSL のダウンロード

Apache と mod_SSL は、それぞれ以下のサイトから入手できます。

■Apache：　http://www.apache.org

■mod_SSL:　http://www.modssl.org

ここでは、Apache は“apache_1.3.37”を、mod-SSL は“mod_ssl-2.8.28-1.3.37[*1]”をダウンロードします。

*1：接続する Apache のバージョンに合ったもの(下線部の数値が同じもの)をダウンロードします。

アーカイブの展開

Apache と mod_SSL のアーカイブを展開します。

◆Apache1.x 系と mod_SSL のｱｰｶｲﾌﾞを展開

```
$ tar xvfz apache_1.3.37.tar.gz
$ tar xvfz mod_ssl-2.8.28-1.3.37.tar.gz
```

コンパイルとインストール

コンパイルを実行する前に、mod_SSL を展開したディレクトリに移動し、次のような処理(configure)を行ないます。

◆configure の実行

```
$ cd mod_ssl-2.8.28-1.3.37        mod_SSL の展開先
$ ./configure
    --with-apache=../apache_1.3.37
    --with-ssl=/usr/local/ssl
```

次に、Apache を展開したパスに移動し、コンパイルおよびインストールを実行します。

◆ｺﾝﾊﾟｲﾙとｲﾝｽﾄｰﾙの実行

```
$ cd ../apache_1.3.37        Apache の展開先
$ make
$ su
# make install
```

Apache の設定

Apache を正しく動作させるためには、設定ファイル“/usr/local/apache/conf/httpd.conf”の編集が必要となります。

このうち、秘密鍵とサーバ証明書の所在[*1]、名称 (パス)の指定形式についてのみ例示します。その他の設定内容[*2] については、関連するサイト等をご覧になって下さい。

◆証明書と秘密鍵の設定例 (httpd.conf)

```
SSLCertificateFile        証明書格納先のﾊﾟｽを指定
  /usr/local/apach/conf/ssl.crt/server.crt
SSLCertificateKeyFile     秘密鍵格納先のﾊﾟｽを指定
  /usr/local/apach/conf/ssl.key/server.key
```

*1：作成済みの秘密鍵と証明書を上記ディレクトリに移動していることを前提とします。

*2：User、Group、ServerName 等

ライブラリの確認

OpenSSL のライブラリが Apache にリンクされていることを確認します。リンクされていない場合は、環境変数(LD_LIBRARY_PATH)を正しく設定します。

◆ﾗｲﾌﾞﾗﾘの確認

```
# ldd /usr/local/apache/bin/httpd
            :
libssl.so.0.9.8=>/usr/local/ssl/lib/libssl.so.‥
            :
```

以上で、Apache1.x 系での Apache と mod_SSL のインストールが終了しました。

設定ファイルの変更を有効にするためには Apache の(再)起動が必要となります。

Apache2.x 系の場合

Apache のダウンロード

Apache1.x 系と同じサイトから入手できます。ここでは、“httpd-2.2.4”をダウンロードします。

アーカイブの展開

Apache のアーカイブを展開します。

◆Apache2.x 系のｱｰｶｲﾌﾞを展開

```
$ tar xvfz httpd-2.2.4.tar.gz
```

コンパイルとインストール

以下の手順により、コンパイルおよびインストールを実行します。

◆コンパイルとインストールの実行

```
$ cd httpd-2.2.4
$ ./configure
  --with-ssl=/usr/local/ssl
  --enable-ssl
$ make
$ su
# make install
```

Apache の展開先

Apache の設定

Apache1.x 系同様、Apache を正しく動作させるためには、設定ファイル"/usr/local/apache2/conf/extra/http-ssl.conf"の編集が必要となります。

なお、秘密鍵とサーバ証明書の所在、名称 (パス)の指定形式については、Apache1.x 系に準じます。

◆httpd-ssl.conf の編集内容

```
SSLCertificateFile
/usr/local/apache2/conf/server.crt
SSLCertificateKeyFile
/usr/local/apache2/conf/server.key
        :
<VirtualHost _default_:443>
ServerName xxxxxxxx:443
```

証明書格納先のパスを指定

秘密鍵格納先のパスを指定

Apache をインストールしたサーバの名称を指定

◆httpd.conf の編集内容

```
ServerName xxxxxxxx:80
        :
        :
#Include conf/extra/httpd-ssl.conf
```

Apache をインストールしたサーバの名称を指定

コメント(#)を外し、有効行にする。

ライブラリの確認も、Apache1.x 系と同様に行ないます。

◆ライブラリの確認

```
# ldd /usr/local/apache2/bin/httpd
```

以上で、Apache のインストールが終了しました。

設定ファイルの変更を有効にするために Apache の再起動を行います。

Apache の起動と停止

Apache1.x 系の場合

Apache1.x 系の起動・停止方法を次に示します。

Apache の起動方法

Apache を SSL 無しで起動する方法と、SSL 有りで起動する方法を以下に示します。

◆Apache の起動

```
【SSL 無しの場合】
# /usr/local/apache/bin/apachectl start
【SSL 有りの場合】
# /usr/local/apache/bin/apachectl startssl
```

Apache の起動確認

Apache が起動していることを確認します。「SSL 無し」で起動した場合は"http://xxxx..../ "(xxxx はサーバ名)をブラウザのアドレス欄に、「SSL 有り」の場合は、プロトコルの部分を"https"に置き換えてサーバの URL を指定します。

画面に"インストールが無事終了した"旨の内容が表示されればインストールは OK です。

なお、「SSL 有り」で起動した場合は、秘密鍵を作成した時のパスフレーズを確認されますので入力して下さい。

Apache の停止方法

Apache を停止する方法を以下に示します。

◆Apache の停止

```
# /usr/local/apache/bin/apachectl stop
```

Apache2.x 系の場合

Apache2.x 系の起動・停止方法を以下に示します。

Apache の起動方法

httpd2.2.x では、処理中のプロセスが終わってか

ら終了あるいは再起動させるために、“graceful”という指定が追加されましたが、ここではApache1.x系同様、“start”による起動方法を示します。

◆Apache の起動

```
# /usr/local/apache2/bin/apachectl start
```

Apache 起動時に次のようなエラーメッセージが出力された場合、80 番ポートが既に使用されていることが原因ですので、ps コマンドで httpd のプロセスが存在するか確認し、存在したならば kill コマンドで削除してから Apache を再度起動して下さい。

◆起動時のエラーメッセージ

```
(98)Address already in use:make_sock:could not bind to address xxxxxxx:80 no listening sockets available, shutting down
```

Apache の停止方法

Apache を停止する方法を以下に示します。

◆Apache の停止

```
# /usr/local/apache2/bin/apachectl stop
```

Camellia を最優先で選択させるための設定

Apache2.x 系において、Camellia を最優先で選択させるための設定について説明します。

※後述しますが、Apache1.x 系では最優先に選択させる指定方法はありません。

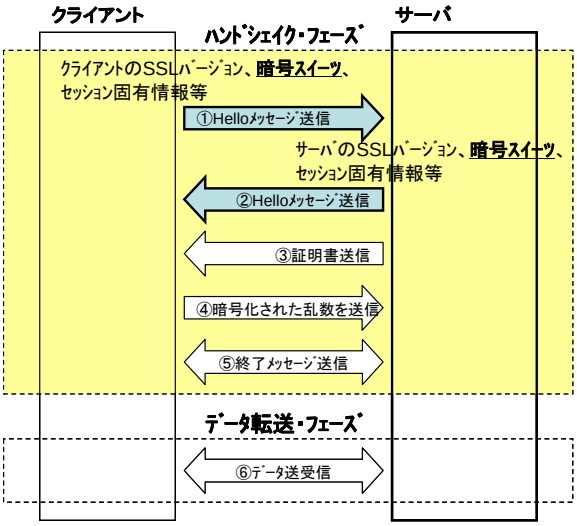
SSL ハンドシェイクについて

Camellia を優先的に使用する方法を説明する前に“SSL ハンドシェイク”について、簡単に説明します。

“SSL ハンドシェイク”とは、クライアント(ブラウザ)と Web サーバとの間で SSL セッションを確立するプロセスのことで、使用する暗号の決定、サーバーおよびクライアントの認証、データを保護するための暗号鍵の設定等を行います。この時点で、優先的に使用する暗号が決まります。

SSL ハンドシェイクの処理イメージを以下に示します(詳細な説明は省略します)。

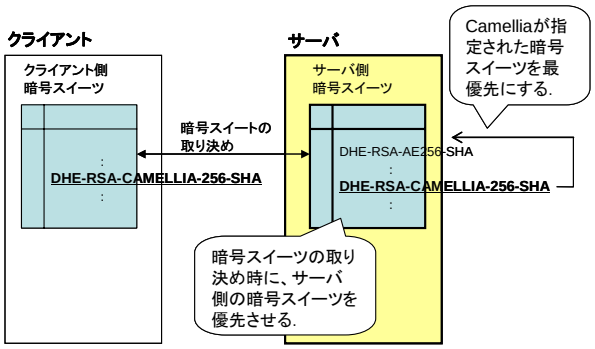
◆SSLﾊﾝﾄﾞｼｪｲｸの処理ｲﾒｰｼﾞ



Camellia を優先的に選択させる設定

SSL ハンドシェイクの中で、サーバとクライアント間で暗号スイーツ[*1] の取り決めが行なわれます。

この時、通常はクライアント側の暗号スイーツが優先されますが、サーバ側の暗号スイーツを操作(設定を変更)することで、Camellia を優先的に選択させることができます。



具体的には、設定ファイル“/usr/local/apache2/conf/extra/httpd-ssl.conf”のディレクティブを以下のように編集します。

◆httpd-ssl.conf の編集

```
:
SSLHonorCipherOrder  on
:
SSLCipherSuite  CAMELLIA:ALL…
:
```

サーバ側の暗号スイーツを優先させる

現在の設定の先頭に”CAMELLIA”を追記する

ただし、Apache1.x 系では、上記のような機能がサポートされていませんので、Camellia を最優先で選択させるには Apache2.1 以降で SSL サーバを構築する必要があります。

*1：SSL コネクションが認証、鍵交換、ストリーム暗号化を行うために使う、アルゴリズムの組合せ。

Camellia が使用されていることを確認する方法

SSL 通信において、実際に Camellia が使用されていることを確認する方法を以下に示します。

◆Camellia が使用されていることを確認

```
# cd /usr/local/ssl/bin
# ./openssl s_client -connect xxxxxxxx:443
  -cipher 'CAMELLIA:ALL'
    :
SSL-Session:
  Protocol:TLSv1
  Cipher:ADH-CAMMELIA256-SHA
    :
```

Apacche をインストールしたホスト名を指定

“s_client”は、サーバと SSL/TLS 接続を確立するテスト用のコマンドで、“-connect”オプションで接続する相手のホストとポートを指定します。

以上で、Camellia＋Apache による SSL サーバの構築がすべて終了しました。

Camellia を利用できる OSS

OS カーネル

OS	Ver.	利用できる機能
Linux	2.6.21 以降	IPsec
Fedora Core	7 以降	IPsec
FreeBSD 7 系	7.0 以降	IPsec、パーティション暗号化(geli)
FreeBSD 6 系	6.4 以降	IPsec

ライブラリ

製品名	Ver.	備考
OpenSSL	0.9.8c 以降	0.9.8 系は、コンパイルオプションで enable-camellia を指定する必要あり
NSS	3.12 以降	
Crypto++	5.4 以降	
GNU TLS	2.2.0 以降	libgcrypt 1.3.0 以降が含まれていること
BouncyCastle	1.30 以降	
camellia-rb	-	Ruby 向け Camellia ライブラリ

アプリケーション

製品名	Ver.	利用できる機能
Firefox	3.0 以降	SSL
GnuPG	2 以降	OpenPGP
ipsec-tools	0.7 以降	IPsec、IKE

Camellia が有効な OpenSSL が同梱されている OS

製品名	Ver.
Fedora Core	9 以降
OpenSUSE	10.3 以降
Gentoo Linux	2008.0 以降
FreeBSD	7.0 以降
FreeBSD ports	2007/6/12 以降

Camellia に関する情報の入手方法

Camellia に関する最新情報や紹介記事等は、次のサイトから入手することができます。

■サイト名：NTT の暗号要素技術 > Camellia

http://info.isl.ntt.co.jp/crypt/camellia/index.html

■Camellia に関する情報

- Camellia に関するニュースリリース・関連記事
- Camellia の紹介
- 標準化情報(Camellia を認定した標準化団体等)
- Camellia を採用したセキュリティ製品の紹介
- Camellia 仕様書等の技術情報

■本件問い合わせ先：camellia@lab.ntt.co.jp