

---

# Linux インターネットサーバーの SSL 証明書管理

～Kousec Server Certificate Manager の導入効果の検証～

第二回 サーバー上での SSL 設定作業



Copyright 2009 Kousec Software, Inc. All rights reserved.  
All company names and product names are trademarks of their respective holders.



---

## シリーズインデックス

[http://www.kousec.com/prod\\_cm\\_ja.html](http://www.kousec.com/prod_cm_ja.html)

第一回： 検証構成と SSL 証明書の取得・配備計画

PDF: [http://www.kousec.com/tj/tj\\_review\\_S1.pdf](http://www.kousec.com/tj/tj_review_S1.pdf)

第二回： サーバー上での SSL 設定作業

PDF: [http://www.kousec.com/tj/tj\\_review\\_S2.pdf](http://www.kousec.com/tj/tj_review_S2.pdf)

第三回： 証明書仕様の定義と証明書の取得

PDF: [http://www.kousec.com/tj/tj\\_review\\_S3.pdf](http://www.kousec.com/tj/tj_review_S3.pdf)

第四回： 証明書の配備と監視

PDF: [http://www.kousec.com/tj/tj\\_review\\_S4.pdf](http://www.kousec.com/tj/tj_review_S4.pdf)

第五回： セキュリティと運用のベストプラクティス

PDF: [http://www.kousec.com/tj/tj\\_review\\_S5.pdf](http://www.kousec.com/tj/tj_review_S5.pdf)

---

## Table of Contents

サーバー上での SSL 設定作業 .....	5
Linux OS のインストールと各サーバーソフトウェアのインストール .....	6
証明書関連ファイルのサーバー上での配置場所は？ .....	6
サーバーソフトウェア間での証明書関連ファイルは共有していいの？ .....	8
各サーバーソフトウェアの SSL 証明書の初期設定を行う .....	9
自己署名証明書の生成 .....	9
OpenLDAP (slapd) .....	10
Apache2 (httpd) .....	12
FTP Server (vsftp) .....	13
SMTP Server (Postfix) .....	14
POP/IMAP Server (Dovecot) .....	15
Tomcat6 .....	17
サーバー作業の確認 .....	19

---

## サーバー上での SSL 設定作業

第一回では、サーバー上にインストールするサービスとそれに伴う SSL サーバー証明書の要件を定義しました。今回からは実際にサーバーソフトウェアのインストール・設定、証明書の取得配備といった作業を行っていきます。

検証の全体作業の流れは以下のとおりです。

1. **SSL 証明書の取得・配備計画の作成**
2. **Linux OS のインストール・各サーバーソフトウェアのインストール**
3. **各サーバーソフトウェアのコンフィギュレーション**  
このコンフィギュレーション内で、まず自己署名証明書が作成され、それを用いて SSL を設定しておきます。
4. **Kousec Server Certificate Manager 上で証明書定義を作成**  
ステップ 1 で決めた証明書要件を証明書定義として作成していきます。
5. **Kousec Server Certificate Manager 上で証明書取得プロセスを実行**  
証明書を CA から取得し証明書リポジトリに保存します。今回は各証明書をビルトインプライベート CA から発行します。
6. **Kousec Server Certificate Manager 上で証明書配備プロセスを実行**  
証明書リポジトリに保存された証明書を定義されたサーバーに配備していきます。具体的には各サービスに対して証明書のインストールを行います。ここで、ステップ 3 で作成された自己署名証明書を CA からの証明書で置き換えることになります。
7. **Kousec Server Certificate Manager 上で配備した証明書の監視設定を実施**

それぞれのステップを回を分けて説明していきます。

第二回： ステップ 2、ステップ 3

第三回： ステップ 4、ステップ 5

第四回： ステップ 6、ステップ 7

第五回： 運用ベストプラクティス、まとめ

また、今回から本文中で Kousec Server Certificate Manager の略称として 「CertMgr」 や 「Kousec CertMgr」 を使います。

---

## Linux OS のインストールと各サーバーソフトウェアのインストール

今回の検証では Ubuntu Server 9.04 を使用します。

OS のインストール時には、Software selection にて LAMP server, Mail server, OpenSSH server, Tomcat Java server を選択しました。また足りないパッケージについては Ubuntu Server Guide に従い apt-get コマンドで追加インストールしていきます。

下記ガイドを参考にしてインストールを行います。

Ubuntu Server Guide 9.04 <https://help.ubuntu.com/9.04/serverguide/C/index.html>

## 証明書関連ファイルのサーバー上での配置場所は？

各サーバーソフトウェアの SSL コンフィギュレーションを行うまえに、Linux サーバー上での証明書関連ファイルの配置場所について考えておく必要があります。

Ubuntu Server や CentOS 5.x では SSL サーバー証明書を、各サーバーソフトウェアの設定ファイルとは別に、ある特定のディレクトリ配下に配置することを推奨しているようです。Ubuntu Server では下記ディレクトリに証明書関連ファイルを置いています。

`/etc/ssl/certs/` : サーバーおよび CA の SSL 証明書

`/etc/ssl/private/` : サーバー証明書の秘密鍵

CentOS 5.x でも同様です。

`/etc/pki/tls/certs/` : サーバーおよび CA の SSL 証明書

`/etc/pki/tls/private/` : サーバー証明書の秘密鍵

しかし、将来のサーバー拡張に備えて、各サーバーソフトウェアが提供するサービスはサーバーマシンと出来るだけ疎な関係にしておき、サーバーソフトウェアの関連ファイルだけを別サーバーに簡単に持っていけるようにしておきたいと考えます。上記の慣習に従った場合、具体的には以下のような問題が出ます。

- ・ 秘密鍵ファイルのパーミッションの設定が複雑になる。  
秘密鍵ファイルはサーバーソフトウェアだけが読めるように設定しなければなりません

---

ん。多くのサーバーソフトウェアはまだ root ユーザーで実行中の間に秘密鍵を読み込み、その後、指定された一般ユーザーまで権限を落とすという動作を行うので、root ユーザーだけが秘密鍵を読み込めればよいのですが、そうではないサーバーソフトウェアもあります。例えば Openldap は openldap 用一般ユーザーで秘密鍵を読みに行きます。その場合、秘密鍵ファイルおよびそのディレクトリ private は root だけでなく openldap ユーザーでも読めるように設定しておかなければなりません。Tomcat6 も同様に tomcat6 ユーザーで秘密鍵ファイルを読みに行きます。したがって private ディレクトリは tomcat6 ユーザーでも読めるように設定する必要があります。他のサーバーソフトウェアと private ディレクトリや秘密鍵ファイル自体を共有した環境でこのような設定を行うのは多くのミス招きやすいものです。<sup>1</sup> (読めるべきものが読めなかったり、読めないはずのものが読めたりする。)

- あるサーバーソフトウェアを別サーバーに移行する際などでサーバーソフトウェア環境をバックアップすると他サービスの秘密鍵までバックアップ・転送されてしまう。例えば FTP サービスを別マシンに移行するのに LDAP サービスの秘密鍵まで間違えてコピーされ移行先サーバーにおかれるというミスがおきやすくなります。
- あるサーバーソフトウェアが侵入された場合他のサーバーソフトウェア用の秘密鍵も危機にさらされる。

上記理由から、サービスに付随すべき SSL 証明書はそのサーバーソフトウェアの他の設定ファイルと同じ場所に配置しておき、パーミッションを root だけ、必要ならそのサーバーソフトウェア専用のユーザーだけに与えるように設定していくこととします。具体的には、例えば Apache が使う SSL 証明書であれば、`/etc/apache2/`、OpenLDAP であれば`/etc/ldap/`の下に `ssl/certs` と `ssl/private` ディレクトリを作りそこに格納します。

証明書ファイル: `/etc/<サーバーソフトウェア名>/ssl/certs/<証明書名>.cert`

秘密鍵ファイル: `/etc/<サーバーソフトウェア名>/ssl/private/<証明書名>.key`

証明書チェーンファイル: (サーバー証明書が中間 CA に署名されている場合のみ)

`/etc/<サーバーソフトウェア名>/ssl/certs/<証明書名>-chain.cert`

トラスト CA ファイル: (多くの場合 SSL クライアント認証のときだけ必要)

`/etc/<サーバーソフトウェア名>/ssl/certs/trusted-ca.cert`

ファイルの場所に関する注意: Ubuntu Server 上でデフォルトで稼働している AppArmor があるサーバーソフトウェアに対して有効になっている場合、そのサーバーソフトウェアは AppArmor で設定されたファイル・ディレクトリに対してしか読み書きができないよう

---

<sup>1</sup> Ubuntu Server では、`ssl-cert` というグループを作り private ディレクトリおよび秘密鍵ファイルのグループを `ssl-cert` に設定、そして openldap ユーザーなどの補助グループとして `ssl-cert` を設定しています。

になります。上記のようにファイルの格納場所を検討する場合は AppArmor のプロファイルも確認する必要があります。apparmor\_status コマンドで現在のプロファイルが確認できます。Ubuntu Server 9.04 では slapd(OpenLDAP)に AppArmor プロファイルが適用されています。

下記に今回使用するサービスと証明書の場合・パーミッションなどを整理した表を掲載しておきます。

なお、後ほど使用する CertMgr では各サーバーソフトウェアの設定ファイルから証明書関連ファイルの場所を読み出すようになっており、これらの場所を一つずつ入力していく必要はありません。

表 1 証明書関連ファイルの場所とセキュリティ設定

サービス	秘密鍵の場所 証明書の場所	秘密鍵の owner:group	秘密鍵の permission
OpenLDAP	/etc/ldap/ssl/private/ldap.example.com.key /etc/ldap/ssl/certs/ldap.example.com.crt	root:openldap	0640
Apache2	/etc/apache2/ssl/private/www.example.com.key /etc/apache2/ssl/certs/www.example.com.crt	root:root	0600
FTP Server	/etc/vsftpd/ssl/private/www2.example.com.key /etc/vsftpd/ssl/certs/www2.example.com.crt	root:root	0600
Postfix	/etc/postfix/ssl/private/smtp.example.com.key /etc/postfix/ssl/certs/smtp.example.com.crt	root:root	0600
Dovecot	/etc/dovecot/ssl/private/pop.example.com.key /etc/dovecot/ssl/certs/pop.example.com.crt	root:root	0600
Tomcat6	/etc/tomcat6/ssl/private/www2.example.com.jks	root:tomcat6	0640

### サーバーソフトウェア間での証明書関連ファイルは共有していいの？

Ubuntu Server のデフォルトでは単一の自己署名証明書のファイルを物理的にも複数のサーバーソフトウェア間で共有しています。証明書の更新作業の削減のためだと思われそうですが、これだと証明書を共有しているサービス全てを停止してから証明書の更新が必要になります。

サーバーとサービスを分離するという原則からは、たとえばマルチドメイン証明書を複数の

---

サービスが使う場合でも物理的には別ファイルにしておくほうがよいでしょう。サービスの独立性が向上し、サービス毎に証明書の更新等が可能になります。

今回の検証では、POP と IMAP、FTP と WWW2 がそれぞれマルチドメイン証明書を使います。POP と IMAP は Dovecot という単一のサーバーソフトウェアの制御下で共有されるので物理的にファイルも共有します。FTP と WWW2 はそれぞれ vsftpd と tomcat6 が使用するのでファイルは別にしておきます。(ちなみに tomcat6 では JKS という証明書形式を使うためそもそも vsftpd と証明書ファイルを共有することはできません)

## 各サーバーソフトウェアの SSL 証明書の初期設定を行う

さて、証明書の配置方針が決まったら、各サーバーソフトウェアの設定を行っていきます。Ubuntu Server 9.04 ではソフトウェアパッケージのインストールと同時に基本的な設定も自動的に行われます。ただし SSL とその証明書の設定は多くのケースでほとんど行われないので、Ubuntu Server の公式マニュアルである Ubuntu Server Guide にしたがって、SSL の設定を行っていきます。ここでは、SSL と SSL 証明書に関するポイントを記載します。

なお、記載しているコマンドはすべて root になって実行します。(sudo -s で root でシェルが起動する)

## 自己署名証明書の生成

Ubuntu Server Guide では、まず自己署名のサーバー証明書を一つ作り、全てのサーバーソフトウェアがそれを共有するという形でのコンフィギュレーションを説明しています。Ubuntu Server 9.04 では OS インストール時に一つでも SSL を使うサーバーソフトウェアをインストールしていると自動的に自己署名したサーバー証明書が下記の場所に作成されています。

<code>/etc/ssl/certs/ssl-cert-snakeoil.pem</code>	サーバー証明書
<code>/etc/ssl/private/ssl-cert-snakeoil.key</code>	サーバー秘密鍵

我々は SSL の初期設定としてこの自己署名証明書を各サーバーソフトウェア用にコピーして証明書の設定を行っていきます。

---

万が一この自己署名証明書を再作成したい場合は、Ubuntu Server Guide の [Creating a Self-Signed Certificate](#) での手順に従うか下記コマンドを実行すればよいでしょう(よりコマンド数を削減した手順です)。

```
> cd /tmp
> openssl req -x509 -nodes -days 365 -newkey rsa:1024 -keyout server.key -out
server.crt
[DNの情報を入力。Common Nameにldap.example.comを入力。後はデフォルトにしたがってEnter
を押していけばよい。]
> cp server.crt /etc/ssl/certs/ssl-cert-snakeoil.pem
> cp server.key /etc/ssl/private/ssl-cert-snakeoil.key
```

## OpenLDAP (slapd)

Ubuntu Server Guide の OpenLDAP Server 章にしたがってインストールと設定を行っていきます。apt-get install でインストールと基本的な設定が終わりますので、基本動作確認後、SSL の設定を行います。

Ubuntu Server Guide とは異なる場所にある SSL 証明書を設定します。SSL 証明書パラメーターと設定すべきパスは以下のとおりです。

```
olcTLSCertificateFile /etc/ldap/ssl/certs/ldap.example.com.crt
olcTLSCertificateKeyFile /etc/ldap/ssl/private/ldap.example.com.key
olcTLSCACertificateFile /etc/ldap/ssl/certs/trusted-ca.crt
```

ポイントは以下です。

- ・ 鍵ファイルは root ユーザーではなく openldap ユーザーで読み込まれるので openldap ユーザーに read パーミッションを与えなければならない。
- ・ slapd では、SSL クライアント認証を行わない場合でもトラスト CA ファイルを指定する必要がある。自己署名証明書を使う場合、その証明書だけを格納しておく。

以下のコマンドで自己署名証明書を slapd 用にコピーしてきます。

```
>mkdir -p /etc/ldap/ssl/certs
>mkdir /etc/ldap/ssl/private
>cd /etc/ldap/ssl
>chgrp openldap private ; chmod 0750 private
>cp /etc/ssl/certs/ssl-cert-snakeoil.pem certs/ldap.example.com.crt
>cp /etc/ssl/certs/ssl-cert-snakeoil.pem certs/trusted-ca.crt
>cp /etc/ssl/private/ssl-cert-snakeoil.key private/ldap.example.com.key
>chgrp openldap private/* ; chmod 0640 private/*
```

あとは Ubuntu Server Guide に従い、ldapmodify コマンドで cn=config に証明書関連ファイルの場所を設定します。なお Ubuntu Server Guide では ldif ファイルを作らず直接入力する方法を示しているが ldif ファイルを作成したほうがミスが少なくなるでしょう。参考に ldif ファイルと必要なコマンドを載せておきます。

下記内容を certfile.ldif というファイル名で保存する。

```
dn: cn=config
changetype: modify
replace: olcTLSCertificateFile
olcTLSCertificateFile: /etc/ldap/ssl/certs/ldap.example.com.crt

dn: cn=config
changetype: modify
replace: olcTLSCertificateKeyFile
olcTLSCertificateKeyFile: /etc/ldap/ssl/private/ldap.example.com.key

dn: cn=config
changetype: modify
replace: olcTLSCACertificateFile
olcTLSCACertificateFile: /etc/ldap/ssl/certs/trusted-ca.crt
```

## コマンド

```
> ldapmodify -x -D cn=admin,cn=config -W -f certfile.ldif
```

ここで 1 点注意しなければならないのは、SSL 証明書関連ファイルの設定が間違っていると通常の方法では slapd が起動しなくなることです。しかし slapd が起動しなければ ldapmodify コマンドでパラメーターを変更できません。その状態に陥った場合は、

---

openldap ユーザーで/etc/ldap/slapd.d/cn=config.ldif を vi で開き、olcTLSCertificateFile など 3 行を削除すれば起動するようになります。

そうなる前に、ldapmodify コマンドを打つ前に下記の確認方法で証明書関連ファイルの整合性を確認しておくのがよいでしょう。

#### 確認方法

```
> su -s /bin/bash openldap
[ パーMISSIONも確認するため openldap ユーザーで実行する ]
> openssl verify -CAfile certs/trusted-ca.crt certs/ldap.example.com.crt
[ certs/ldap.example.com.crt: OK と表示されれば OK ]
> openssl x509 -noout -modulus -in certs/ldap.example.com.crt | openssl md5
[ MD5 のハッシュ値(1)が表示される。 ]
> openssl rsa -noout -modulus -in private/ldap.example.com.key | openssl md5
[ ここで表示される MD5 ハッシュ値が(1)の値と一致すれば OK ]
```

あとは、Ubuntu Server Guide どおり、/etc/default/slapd で”ldaps”を有効にし、slapd を再起動させます。

## Apache2 (httpd)

Ubuntu Server Guide の Apache2 Web Server 章にしたがってインストールと設定を行っていきます。Ubuntu Server 9.04 では、Apache2 をインストールするとデフォルトの SSL 仮想サーバーの設定ファイルまで自動的に作成されます。使用される証明書は/etc/ssl/certs にある自己署名証明書です。

これを Apache2 専用に配置した証明書を使うように設定ファイルを書き換えます。

```
証明書関連設定ファイル /etc/apache2/sites-available/default-ssl
```

パラメータの値は以下のとおり。

```
SSLCertificateFile /etc/apache2/ssl/certs/www.example.com.crt
SSLCertificateKeyFile /etc/apache2/ssl/private/www.example.com.key
(SSLCertificateChainFile は自己署名証明書の場合設定しない)
```

---

以下のコマンドで自己署名証明書を apache2 用にコピーしてくる。

```
>mkdir -p /etc/apache2/ssl/certs
>mkdir /etc/apache2/ssl/private
>cd /etc/apache2/ssl
>chgrp root private ; chmod 0700 private
>cp /etc/ssl/certs/ssl-cert-snakeoil.pem certs/www.example.com.crt
>cp /etc/ssl/certs/ssl-cert-snakeoil.pem certs/trusted-ca.crt
>cp /etc/ssl/private/ssl-cert-snakeoil.key private/www.example.com.key
>chgrp root private/* ; chmod 0700 private/*
```

そして、もしまだなら、“a2enmod ssl ; a2ensite default-ssl” を実行し SSL を有効にします。最後に Apache2 を再起動して動作確認を行います。

## FTP Server (vsftpd)

Ubuntu Server Guide の vsftpd の章にしたがってインストールと設定を行っていきます。Ubuntu Server 9.04 では、vsftpd をインストールするとデフォルトの自己署名証明書を使うよう設定されます。

これを vsftpd 専用に配置した証明書を使うように設定ファイルを書き換えます。

証明書関連設定ファイル /etc/vsftpd.conf

パラメータの値は以下。

```
rsa_cert_file=/etc/vsftpd/ssl/certs/www2.example.com.crt
rsa_private_key_file=/etc/vsftpd/ssl/private/www2.example.com.key
```

vsftpd では/etc 配下に専用のディレクトリを持っていません。今回 SSL 証明書を格納するために/etc/vsftpd というディレクトリを作成します。

以下のコマンドで自己署名証明書を vsftpd 用にコピーします。

```
>mkdir -p /etc/vsftpd/ssl/certs
>mkdir /etc/vsftpd/ssl/private
```

---

```
>cd /etc/vsftpd/ssl
>chgrp root private ; chmod 0700 private
>cp /etc/ssl/certs/ssl-cert-snakeoil.pem certs/www2.example.com.crt
>cp /etc/ssl/private/ssl-cert-snakeoil.key private/www2.example.com.key
>chgrp root private/* ; chmod 0700 private/*
```

vsftpd を再起動して動作確認を行います。

## SMTP Server (Postfix)

今回構築する SMTP サーバーには 3 つの機能が必要です。

- A) 社内 PC からの社外向け送信メールを受け取る (認証付 SMTP Server 機能)
- B) 社外向けメールをインターネットへ転送する (SMTP Client 機能)
- C) インターネットから社内向けメールを受信する (公開 SMTP Server 機能)

この中で SSL サーバー証明書が頻繁に使われるのは A のケースだけです。SMTP 認証を行うときの暗号化と相手認証のために SSL が使用されます。ここでは A に関連する SSL 証明書についてのみ説明します。

A の機能の仕様は以下のとおり。

- SMTP 認証でメール送信を要求するユーザーを認証する。
- 送信専用ポート(submission port) TCP/587 で受け取る
- TLS で暗号化とサーバー認証を行う

Ubuntu Server 9.04 では、Postfix をインストールするとデフォルトの自己署名証明書が設定されたコンフィグファイルが作られます。また Ubuntu Server Guide に従い **dovecot-postfix** というパッケージをインストールすると、SMTP ユーザー認証を POP/IMAP サーバーである Dovecot 内のユーザー認証機能を使って実現するようになります。(ちなみに **dovecot-postfix** をインストールするとその延長で Dovecot 自体がインストールされます)。また、送信専用ポート 587 番はデフォルトでは有効になっていないので、`/etc/postfix/master.cf` ファイルを開き”submission”の行を有効にします。

---

ではここから、SSL サーバー証明書の設定を行って行きましょう。Postfix 専用に配置した証明書を使うように設定ファイルを書き換えます。

証明書関連設定ファイル `/etc/postfix/main.cf`

パラメータの値は以下を設定します。

```
smtpd_tls_cert_file=/etc/postfix/ssl/certs/smtp.example.com.crt
smtpd_tls_key_file=/etc/postfix/ssl/private/smtp.example.com.key
```

また、`main.cf` を修正するときに、ついでに下記パラメーターもコメントアウトしておきます（Ubuntu 9.04 の `dovecot-postfix` パッケージのバグのようです）

```
smtpd_tls_mandatory_ciphers = medium, high
```

では、以下のコマンドで自己署名証明書を Postfix 用にコピーします。

```
>mkdir -p /etc/postfix/ssl/certs
>mkdir /etc/postfix/ssl/private
>cd /etc/postfix/ssl
>chgrp root private ; chmod 0700 private
>cp /etc/ssl/certs/ssl-cert-snakeoil.pem certs/smtp.example.com.crt
>cp /etc/ssl/private/ssl-cert-snakeoil.key private/smtp.example.com.key
>chgrp root private/* ; chmod 0600 private/*
```

なお、Postfix が SMTP Client として動作する場合に使用できる SSL クライアント証明書は設定しません。SMTP で SSL を利用する主な目的は、SMTP 認証を行うときの暗号化と相手認証です。

最後に Postfix を再起動して動作確認を行います。

---

Ubuntu Server 9.04 では、Dovecot をインストールするとデフォルトの自己署名証明書が設定されたコンフィグファイルが作られます。なお Ubuntu Server Guide では Postfix のインストールにおいて SMTP-AUTH を有効にするために、**dovecot-postfix** というパッケージのインストールを推奨しています。これをインストールするとその延長で Dovecot 自体もインストールされますので既に Dovecot はインストールされているはずですが。

また Ubuntu Server Guide の Dovecot インストールの章に書かれていないのですが、**dovecot-postfix** パッケージをインストールした場合、Dovecot のメインのコンフィグファイルは `/etc/dovecot/dovecot.conf` ではなく `/etc/dovecot/dovecot-postfix.conf` になります。Postfix もインストールした環境において、Ubuntu Server Guide を見ながら Dovecot の設定を行う場合 `dovecot.conf` を `dovecot-postfix.conf` と読み替える必要があります。

では SSL 証明書の設定を行っていきましょう。Dovecot 専用に配置した証明書を使うようにコンフィグファイルを書き換えます。

証明書関連設定ファイル `/etc/dovecot/dovecot-postfix.conf`  
(`dovecot-postfix` パッケージを導入していない場合は `/etc/dovecot/dovecot.conf`)

パラメータの値は以下。

```
ssl_cert_file = /etc/dovecot/ssl/certs/pop.example.com.crt
ssl_key_file = /etc/dovecot/ssl/private/pop.example.com.key
```

以下のコマンドで自己署名証明書を Dovecot 用にコピーします。

```
>mkdir -p /etc/dovecot/ssl/certs
>mkdir /etc/dovecot/ssl/private
>cd /etc/dovecot/ssl
>chgrp root private ; chmod 0700 private
>cp /etc/ssl/certs/ssl-cert-snakeoil.pem certs/pop.example.com.crt
>cp /etc/ssl/private/ssl-cert-snakeoil.key private/pop.example.com.key
>chgrp root private/* ; chmod 0600 private/*
```

最後に Dovecot を再起動して動作確認を行います。

---

---

## Tomcat6

最後のサーバーソフトウェアは Tomcat 6 です。

Ubuntu では Tomcat をインストールするのに 2 つの方法があります。一つ目は他のサーバーソフトウェアと同様にシステムにインストールし OS の起動にあわせて起動される方法です(System-wide installation)。もう一つは開発者向けに開発者のプライベートなインスタンスとしてインストールし可能する方法です(Using private instances)。今回は Tomcat も運用向けに使用しますので、System-wide installation の方法でインストールを行います。Ubuntu Server Guide に従いインストールします。

SSL 証明書の設定方法は、Ubuntu Server Guide には記載されていません。しかし Apache Tomcat のマニュアル(<http://tomcat.apache.org/tomcat-6.0-doc/ssl-howto.html>)を見ると簡単に設定できます。参考にここにも記載しておきます。

Tomcat では他の Unix サーバーソフトウェアと異なり、Java Key Store (JKS) というファイルに証明書と秘密鍵を格納します。下記の場所に JKS ファイルを置きます。

```
JKS ファイル /etc/tomcat6/ssl/private/www2.example.com.jks
```

なお、tomcat は root で起動して tomcat6 ユーザーに権限を落としてからこの JKS ファイルを読みに行きます。したがって下記コマンドで private ディレクトリとその中のファイルには下記パーミッションに設定しておきます。

```
owner:group root:tomcat6 permission: 0640
```

注意: Tomcat 6.0 からは、オプションとして OpenSSL を使った SSL エンジンも使用可能になったようです。今回はこちらは使用しませんのでマニュアルを読む際注意してください。

JSEE ベースの SSL コネクター: 今回使用する

APR/OpenSSL ベースの SSL コネクター: 使用しない

---

SSLの初期設定時点では、このJKSファイルに自己署名証明書を生成・格納しておきます。  
そして CertMgr を使って、正式な証明書と置き換えることにします。

1. 下記コマンドで自己署名証明書を生成しておきます。

```
>mkdir -p /etc/tomcat6/ssl/private  
>keytool -genkey -alias tomcat -keyalg RSA -keystore  
/etc/tomcat6/ssl/private/www2.example.com.jks
```

パスワードは keystore password と key password という二つのパスワードの入力が求められます。Tomcat ではこの2つを一致させる必要がありますので同じものを入力してください。Tomcat のデフォルトは changeit です、それを入力します。  
あとは自己署名証明書を格納する DN に関する情報です。これについてはあとで CertMgr から正式な証明書で置き換えるのでここではデフォルトのまま進めてください。

最後に JKS ファイルの権限を設定します。

```
>cd /etc/tomcat6/ssl  
>chgrp tomcat6 private ; chmod 0750 private  
>chgrp tomcat6 private/* ; chmod 0640 private/*
```

2. /etc/tomcat6/server.xml の SSL コネクタの設定を編集します。

- (ア) SSL コネクタ全体がコメントアウトされているのでそれを有効にする。
- (イ) keystoreFile パラメータで JKS ファイルへのパス名を追加
- (ウ) keystorePass で上記で設定したパスワードの追加

```
<!-- Define a SSL HTTP/1.1 Connector on port 8443  
This connector uses the JSSE configuration, when using APR, the  
connector should be using the OpenSSL style configuration  
described in the APR documentation -->  
<!--  
<Connector port="8443" protocol="HTTP/1.1" SSLEnabled="true"  
maxThreads="150" scheme="https" secure="true"  
clientAuth="false" sslProtocol="TLS"  
keystoreFile="/etc/tomcat6/ssl/private/www2.example.com.jks"  
keystorePass="changeit" />
```

---

-->

最後に tomcat6 を再起動して動作確認を行います。

## サーバー作業の確認

ここで各サーバーソフトウェアがサービスを提供する TCP ポート番号とプロトコルを表にまとめて整理しておきます。これに従い Firewall の設定を行きましょう。

表 2 今回設定したサービスのポート番号一覧

ポート番号	サーバーソフトウェア	プロトコル	Over SSL / STARTTLS	
389	OpenLDAP (slapd)	LDAP	STARTTLS	
636	OpenLDAP (slapd)	LDAP	over SSL	
80	Apache2 (httpd)	HTTP	なし	
443	Apache2 (httpd)	HTTP	over SSL	
21	vsftpd	FTP	STARTTLS	
25	Postfix	SMTP	STARTTLS	
587	Postfix	SMTP	STARTTLS	
110	Dovecot	POP3	STARTTLS	
995	Dovecot	POP3	over SSL	
143	Dovecot	IMAP	STARTTLS	
993	Dovecot	IMAP	over SSL	
8443	Tomcat6	HTTP	over SSL	

Over SSL/STARTTLS の列は SSL 化の手法を示しています。**over SSL** とはアプリケーションプロトコルを SSL でカプセル化したもので接続と同時に SSL/TLS 通信が始まります。この場合 SSL 専用のポートを必要とします。**STARTTLS** では通常と同じ（SSL 化されていない）ポートに接続して、アプリケーションプロトコルの命令の一つを使い TLS 通信に切り替えます。クライアントの設定手順作成時やトラブルシューティングのときにこれを意識しておくことが重要です。

参考に、今回のサービスに関連するポートで今回のソフトウェアでは設定しなかったポート番号の一覧も載せておきます。

---

ポート番号	サーバーソフトウェア	プロトコル	Over SSL / STARTTLS	
465	(Postfix)	SMTP	over SSL	
990		FTP	over SSL	

ポート 465 は smtps (SMTP over SSL)であり、多くのメーラーでサポートされています。Postfix でもサポートされており、Ubuntu Server では/etc/postfix/master.cf ファイル中の、”smtps”の行を有効にすれば使えるようになります。より多くのクライアントに対応し、またクライアント PC の設定を容易にするためこのポートも使用したほうがよいでしょう。ポート 990 は ftps (FTP over SSL - Implicit)と呼ばれています。Ubuntu Server 9.04 に含まれる vsftpd ではサポートされていませんが、SSL 対応の FTP クライアントの設定では出てくる用語なので認識しておいたほうがよいでしょう。

#### トラブルシューティングについて

##### ・ 確認方法

SSL 証明書の設定を行いサービスが正常に起動しても証明書の設定があっているかどうかは分かりません。まずは下記で基本的な確認を行っていきます。

- 該当ポートをサーバープロセスがリスニングしていることを確認する。

```
netstat -untap
```

- ローカルマシンから、openssl コマンドで実際にポートに接続しに行く。

```
例 openssl s_client -connect host:port -showcerts [-starttls
<smtp|pop3|imap|ftp>]
```

エラーになったりハングした場合は、syslog や各サーバーソフトウェアのログ (/var/log/\*)をチェックします。

- 次に別のマシンから、telnet で直接、該当の TCP ポートを叩く。

Connection refused になったり Connection timedout になる場合は Firewall の設定を見直します。

##### ・ よくあるミス

秘密鍵が開けない。秘密鍵ファイルのパーミッションを見直す。AppArmor のプロファイルを見直す。

秘密鍵が暗号化されている。openssl コマンドで解除したものを配置する。

証明書と秘密鍵が対応していない。

この確認がとれた後、実際のクライアントや CertMgr から接続を試行できます。そこで発生しうるより上位レイヤー(SSL/PKI 等)での問題については、第 5 章で CertMgr を使ってトラブルシューティングを行っていきます。

---

今回の作業をまとめます。

- 各サーバーソフトウェアを Ubuntu Server Guide に従いインストール・設定する
- 各サーバーソフトウェアは自動生成された自己署名証明書を使って SSL を有効にする
- ただし SSL 証明書の場所については、サーバーソフトウェア毎にディレクトリを作成しそこに自己署名証明書をコピーし、そこを指すように設定ファイルを変更する

このように SSL 証明書を配置し、サービスの SSL 証明書をサーバーマシンや他のサービスから独立させ、保守性を向上、将来のサーバー拡張に備えます。

以上が、サーバー上での SSL 設定作業です。

次回からは、実際に Server Certificate Manager を使用し本番用の SSL 証明書を取得・配備していきます。