サーバ証明書の有効期限を調べるプログラムの作成

○内藤　茂樹 A)
自然科学研究機構分子科学研究所技術推進部 A)

１．はじめに

近年 HTTPS や SMTPS 等の需要が増えており、それに伴い認証機関から発行を受けたサーバ証明書も増えている。サーバ証明書は更新作業が必要であることから、有効期限が近づいたらメールで知らせるプログラムを作成した。使用したプログラミング言語は Python3 で、各サーバに対して実際に SSL/TLS 接続を行って入手したサーバ証明書の有効期限を調べている。今回の発表ではそのプログラムについて報告する。

２．方針

サーバの証明書の有効期限を調べるのに、各サーバにエージェントを入れるか、リモートで証明書を収集して調べるかを考察した。

まずエージェントタイプであるが、数十台あるサーバ全てにインストールするのは手間であり、かつ OS 毎にエージェントを作り直す必要があるかも知れない。そして通知をメールで行うのなら各サーバ単位で出すか、または管理用のサーバを別途立ててそこに情報を集約して送信するかのどちらかが必要となる。一方のリモートタイプであるが、証明書を取得するのに数十台のサーバに対してアクスセスを行う必要がある。しかしエージェントは不要であることからアプライアンス機へも対応可能であるし、通知メールを出すのも 1 台だけに絞ることができる。

このような考察からリモートタイプで行うことにした。

３．HTTPS(port443)への対応

まず port443 を使う一般的な HTTPS サーバの証明書を調べることから始めた。リモートタイプで行う場合、サーバ証明書を各サーバからダウンロードする必要がある。サーバ証明書は TCP ハンドシェイク後の TLS ハンドシェイクで、クライアントが送信する Client Hello に対するサーバの返答である Server Hello で送られてくる。したがって HTTP コマンドまで打たずに TLS ハンドシェイクが終わった時点で接続を終了すればよい(図 1)。ただし実際の

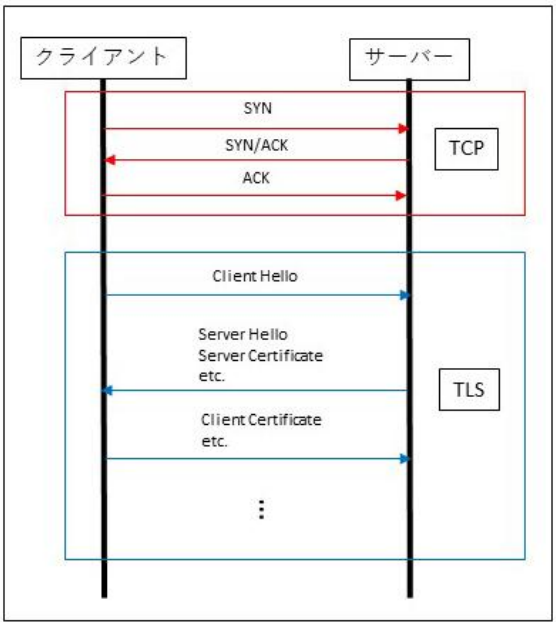

図１　TCP と TLS のハンドシェイク

プログラミングでは、Python3 のモジュールを使うとサーバ証明書を簡単に取得できる。

Python3 でサーバ証明書の有効期限を調べる手順を説明する。まず ssl モジュールを使って、“dataA=ssl.get_server_certificate((サーバ,443))”で変数 dataA として対象となるサーバのサーバ証明書を取得する。この時点では PEM(base64)形式なので、OpenSSL モジュールを使い“dataB = OpenSSL.crypto.load_certificate(OpenSSL.crypto.FILETYPE_PEM,dataA.encode('utf-8'))” で dataA を X.509 オブジェクトに変換したものを dataB として取得する。続いて dataB から有効期限を取り出すのだが、そのままでは UTC なので datetime モジュールを使って JST に変換する。“dataC=datetime.datetime.strptime(str(dataB.get_notAfter())[2:16],'%Y%m%d%H%M%S')+datetim

e.timedelta(hours=9)”で有効期限を JST に変換したものを dataC として取得する。

対象となるサーバを一行に一台記述したリストファイルを用意し、それを配列として読み込んで一台ずつサーバ証明書を取得することにした。

４．HTTPS(port 443 以外)への対応

対象となるサーバの中には HTTPS の port を 443 以外に設定してあるものがあった。そのためプログラムを異なる port 番号にも対応させることにした。サーバ証明書を取得するための関数はそのままで、port 番号を固定から変数にかえるだけで良い。port 番号はリストファイルを csv 形式にして、一行に“サーバ名,port 番号”の組み合わせで記述することにした。そしてプログラム側では csv モジュールを使い“DictReader()”関数で辞書として読み込み、サーバ名をキーに port 番号を値とした。

５．STARTTLS への対応

サーバ証明書はウェブ(HTTPS)だけでなくメール(SMTPS や STARTTLS)でも使われる。分子研でも STARTTLS に対応したメールサーバがあるので、それへの対応を行った。

STARTTLS ではまず port25 を使う普通の SMTP 接続をクライアントが行う。クライアントが送信する EHLO コマンドの返信に“250-STARTTLS”があれば、クライアントが STARTTLS を要求して TLS ネゴシエーションが始まる。その TLS ネゴシエーションでサーバからサーバ証明書が送られてくる(図 2)。したがってプログラムでも SMTP 通信から開始して、途中で STARTTLS を要求することになる。まず smtplib モジュールの SMTP インスタンスを使ってメールサーバに port25 で接続する。続いて“ehlo_or_helo_if_needed()”にて EHLO を送信する。その返信中の STARTTLS の有無を“has_estn(‘STARTTLS’)”で確認し、あれば“starttls()”でサーバに STARTTLS を送信し、“starts TLS negotiation”を開始する。サーバ証明書の取得は“sock.getpeercert(True)”で行うが、HTTPS のサーバ証明書と形式を揃えるため、ssl モジュールの“DER_cert_to_PEM_cert()”インスタンスを使って PEM 形式にした。

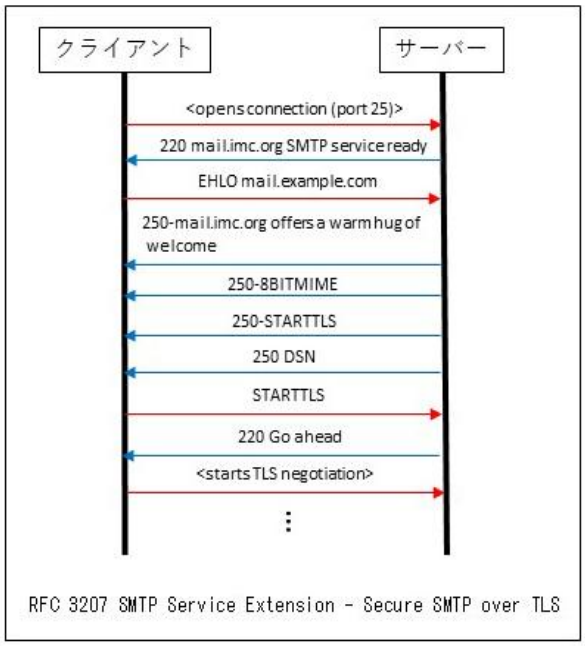

図２　STARTTLS

STARTTLS 形式に対応するためには、サーバのリストファイルも複数プロトコルに対応する必要がある。csv 形式のリストファイルにプロトコルを追加して一行に“サーバ名,port 番号,プロトコル”を記述することにした。リストファイルの 1 行目には“name,port,protocol”と各列の情報を記述した。それを next()関数を使ってヘッダとして読み込み、index()を使ってそれぞれ配列の何番目の要素かを変数に格納した。そして格納した要素番号を使って各サーバの port 番号やプロトコルを取り出し、対応する接続方法でサーバに接続することにした。

また保守などで一時的にサーバを停止しているときにエラーとならないようにする必要がある。そこで停止しているサーバの行の 1 文字目に“#”を記入して、その“#”が付いている場合は当該行を読み飛ばすことにした。同様に空行も読み飛ばすことにした。

６．その他

突発的な障害等でサーバに接続できなかった場合にエラーでプログラムが停止してしまい、残りのサーバへ通信できなくなることがないように、“try”“except”を使って例外処理を施した。

有効期限が設定した日数内に当てはまる場合は、該当するサーバをメールで知らせることにした。メールに記載する内容は、サーバ名、証明書の有効期限及び証明書が切れるまでの残日数とした。また例外処理の対象となった接続できなかったサーバは、エラーメッセージを記載したメールを送信することにした。

７．まとめ

今回作成したプログラム自体は簡単な物で作るのにさほど苦労はしなかった。それは使用した言語がPython3であることも大きいと思われる。Python3には便利なモジュールが豊富にあり、また利用者も多いのでサンプルや解説記事も揃っている。ドキュメントも公開されているので、サンプルで判らない関数などはドキュメントで確認することもできる。

完成したプログラムは１日１回“cron”で動かし、有効期限の迫ったサーバや通信できなかったサーバをメールで知らせている。

今回作成したプログラムのソースコードは技術情報共有サイトで公開する予定である。

・技術情報共有サイト
https://tech-share.ims.ac.jp/

謝辞

分子科学研究所技術推進部計算情報ユニットの方々にはプログラミングに関して助言をいただきました。ありがとうございました。