情報学群実験4C　第 4 回レポート
WWW システムと認証および暗号化

学籍番号 1190319
楠田 健太
グループ4

平成 29 年 7 月 22 日

1　目的

WWW では，現在多くの Web ページが公開されており，Web サーバ，Web ブラウザとも多くのプログラムが提供されている．一方で，インターネットが社会一般に広まったことにより，悪意を持つ利用者も現れることから，適切な情報の発信が重要となる．仮に，公開すべきでない情報を公開してしまった場合には，この情報を秘匿することは困難となる．また，一部の利用者のみに限定した情報公開を行ないたい場合もある．

Web サーバのアクセス制限や認証の機能を用いることにより，これらの問題に対して柔軟に対応することが可能となる．

2　内容

本文書では，Apache を利用して ローカルサーバ上に Web サーバを設定する．ローカルサーバ用のマシンに，パッケージシステムを利用することにより，Apache を導入する．

次に，コンテンツ（HTML）を用意し，公開用の適切なディレクトリに設置する．このとき，ブラウザで用意したコンテンツが見れることを確認する．また，2 種類のアクセス制限（HTTP の接続元の IP アドレスによるアクセス制限 と Basic 認証によるユーザ認証）を設け，それらの通信についてモニタを行なう．一部のページに対しては，2 種類のアクセス制限を異なるディレクトリへの設置を考える．1 つはグループ内の IP アドレス上のみ接続可能な設定，もう 1 つについては，ユーザ認証をクリアしたユーザのみ接続可能な設定を行なう．

それらを設定した後，Wireshark を使用して，HTTP のリクエストとレスポンスの過程，Basic 認証におけるパケット内のパスワードの取り扱いを確認する．

次に，HTTP 通信の暗号化を SSL/TLS を利用することによって行ない，Wireshark にて通信の過程をモニタする．

ここからは，OpenSSL を利用することで，正式な証明書と秘密鍵を用意し，実際に運用する．サーバにて認証局を配置後，認証局の証明書をルート証明書としてクライアントに導入することにより，サーバ証明書を認証局で署名して利用する．

最後に，再び Wireshark を用いて， サーバへの HTTPS 通信をモニタする．

3　要素技術

3.1　HTTP

HTTP（Hypertext Transfer Protocol）は，ブラウザとサーバの間の通信において用いられるプロトコルである．WWW では，HTTP が OSI 参照モデルのアプリケーション層のプロトコルにあたる [1][4].

ユーザがブラウザに Web ページの URI を入力すると，HTTP の処理が始まる．HTTP では，通常 80 番のポート番号が利用されている．まず，クライアントからサーバーへポート 80 番で TCP のコネクション確立が施行され，その通信路を利用することで，コマンドや応答，データからなるメッセージの送受信が可能となる [1].

HTTP では主に HTTP1.0 と HTTP1.1 の 2 つのバージョンが使用されている．HTTP 1.0 までは 1 つの「コマンド」,「応答」をやり取りするたびに TCP コネクションを確立，切断していたのに対し，HTML 1.1 からは 1 つの TCP コネクションにより複数の「コマンド」,「応答」が可

能となった．これによって，TCP のコネクションの確立や切断によるオーバヘッドの削減を実現し，効率が上昇した [1].

3.2 HTML

HTML（HyperText Markup Language）とは，Web ページを記述するためのデータ形式である．WWW では，HTML がプレゼンテーション層のプロトコルにあたる．ブラウザの画面に表示する文字の大きさや位置，色などを指定可能である．さらに，画像や動画を張り付ける設定，音楽を鳴らす設定が可能である [1].

HTML はハイパーテキストという機能をもっており，画面に表示された文字や絵にリンクを張ることにより，そこがクリックされた際に別の情報を提示することを可能とする [1].

また，HTML は WWW 共通のデータ表現プロトコルであり，仕様の異なるコンピュータ間であっても，HTML に従ってデータを用意しておくことにより，ほぼ同じように表示することが可能である [1].

3.3 SSL

SSL（Secure Sockets Layer）とは，Web の HTTP 通信を暗号化する仕組みである．Netscape 社が最初に提案した時の名称は SSL であったが，標準化に際して TLS（Transport Layer Security）という名称に変更された．両方を総称して SSL と呼ぶ場合もある．TLS/SSL を使った HTTP 通信のことを HTTPS と呼び，共通鍵暗号方式で暗号化が行われる．この共通鍵を送信する際には，公開鍵暗号方式が利用される．以下に通信の手順を示す [1].

1. クライアント : Web サーバに HTTPS でアクセス.（利用可能な暗号化アルゴリズムを通知）
2. Web サーバ : Web サーバの「公開鍵」と「証明書」を送信.（使用するアルゴリズムを通知）
3. クライアント : 電子証明書を確認後，公開鍵がアクセスしたサーバの所有物であるかを確認.
4. クライアント : 共通鍵暗号化方式の共通鍵を作成．Web サーバの公開鍵を用いて暗号化し，送信.
5. web サーバ： 暗号を複合し，共通鍵を認知.
6. 共通鍵方式による暗号化通信が可能となる.

公開鍵の正誤の確認は，認証局（CA）から発行された証明書を扱う．主な認証局の情報はあらかじめ Web ブラウザに組み込まれているが，Web ブラウザに組み込まれていない認証局の証明の場合は，警告が表示されるため，認証局を信頼するかどうかを利用者が判断する [1].

3.4 CA

CA（認証局）は，公開鍵基盤において信頼を提供する役割を担う．CA は公開鍵の電子的な身分証明書であるディジタル証明書を発行し，管理する [2].

インターネット上において，CA は一般的に，信頼できる第 3 社組織が運用しており，証明書発行要求の登録や発行，失効リストの管理と発行，署名に使用する秘密鍵の管理などを行なう [2].

認証局を運用する第 3 者組織には，運用規定とディジタル証明書の発行基準が求められる．認証局自身の身元を証明するための認証局の公開鍵と秘密鍵のペアが生成され，認証局の発行したディジタル証明書の署名を検証するため，認証局の公開鍵証明書を生成する．この認証局の公開鍵証明書のディジタル署名は上位の認証局が署名するが，最上位のルート認証局の場合は，自己署名が行われる [2][3].

認証局のディジタル証明書をクライアントの証明書ストアに保存し，その認証局の発行したディジタル証明書の署名の複合と検証に使用する [2].

また，互いに何らかの信頼関係を持っているコミュニティでは，互いの合意により，独自に認証局を立て，そのコミュニティ内で「プライベート認証局」として運用することができる [2].

4 作業記録

4.1 Web サーバの構築

Ubuntu に Apache を インストールする．

1. 以下のコマンドを実行し，プロキシの環境変数を設定する．

プロキシ環境変数設定

```
# export http_proxy=http://192.168.0.1:7999
# export https_proxy+http://192.168.0.1:7999
```

2. 以下のコマンドを実行し，Ubuntu に Apache をインストールする．

Apache のインストール

```
# apt install apache2
```

以上で Apache のインストールは完了である．

4.2 コンテンツの編集と公開

HTML を用いてコンテンツを編集し，公開用ディレクトリ（/ar/www/html）に設置する．また，コンテンツがブラウザから表示可能であることを確認する．

1. 以下のコマンドを実行し，コンテンツを編集する．

コンテンツの編集

```
# vi index.html
```

以上でコンテンツの編集と公開は完了である．

4.3 Web サーバでの認証設定

Basic 認証によるユーザ認証，HTTP の接続元の IP アドレスによるアクセス制限 の 2 種類のアクセス制限の設定を行ない，その通信をモニタする．

4.3.1 Basic 認証によるユーザ認証の設定

1. 以下のコマンドを実行し，ユーザ「testuser」の設定を行なう．

ユーザの設定

```
# htpasswd -c /etc/apache2/.htpasswd testuser
```

2. パスワードを尋ねられるため,「root00」と入力する．

3. 以下のコマンドを実行し，/var/www/html の下にディレクトリ「basic」を作成する．

ディレクトリ「basic」の配置

```
# mkdir basic
```

4. 以下のコマンドを実行後，ファイル編集を行ない，ディレクトリ「basic」へのアクセス制限を設ける．

ディレクトリ「basic」へのアクセス制限の設定

```
# vi /etc/apache2/sites-abailable/000-default.conf

（下記を＜ VirtualHost *:80 ＞ディレクティブ内に追加する）
＜ Directory /var/www/html/basic ＞
        AuthType Basic
        AuthUserFile /etc/apache2/.htpasswd
        AuthName “ basic auth ”

        Satisfy any
        Order deny，allow
        Deny from all
        Require valid-user
＜/Directory ＞
```

入力後，Shift + ZZ を入力して保存する．

以上で Basic 認証によるユーザ認証の設定は終了である．

4.3.2 HTTP の接続元の IP アドレスによるアクセス制限の設定

1. 以下のコマンドを実行し，/var/www/html の下にディレクトリ「inside」を作成する．

ディレクトリ「inside」の作成

```
# mkdir inside
```

2. 以下のコマンドを実行後，ファイル編集を行ない，ディレクトリ「inside」へのアクセス制限を設ける．

ディレクトリ「inside」へのアクセス制限の設定

```
# vi /etc/apache2/sites-available/000-default.conf

（下記を＜ VirtualHost *:80 ＞ディレクティブ内に追加）

＜ Directory /var/www/html/inside ＞
        Order deny，allow
        Deny from all
        Allow from 自グループのサブネット/プレフィックス
＜/Directory ＞
```

入力後，Shift + ZZ を入力して保存する．

以上で HTTP の接続元の IP アドレスによるアクセス制限の設定は完了である．

4.4 HTTPS による暗号通信設定

Ubuntu 上の Apache において，SSL/TLS を有効にするために，必要なモジュールを設定する．

1. 以下のコマンドを実行し，モジュール mod_ssl を有効化する．

モジュール mod_ssl の有効化

```
# a2enmod ssl
```

2. 以下のコマンドを実行し，conf ファイルを用いて SSL を有効にする．

conf ファイルの有効化

```
# a2ensite default-ssl
```

3. 以下のコマンドを実行し，Apache を再起動する．

Apache の再起動

```
systemctl restart apache2
```

以上で HTTPS による暗号通信設定は完了である．

4.5 Apache における鍵の指定

まずは秘密鍵を作成し，公開鍵から正しい情報を付加した後，正式な機関に署名してもらった証明書を作成して，/etc/apache2/sites-available/default-ssl.conf において，以下のように設定を行なう．

default-ssl.conf の設定

ServerName：サーバの FQDN（ブラウザからアクセスするホスト名）
ServerAdmin：Web サーバ管理者に連絡可能なメールアドレス
SSLCertificateFile：サーバ証明書ファイル
SSLCertificateKeyFile：サーバ秘密鍵

入力後，Shift + ZZ を入力し，保存する．
以上で，Apache における鍵の指定は完了である．

4.6 認証局の作成

ルート CA の作成は，OpenSSL のコマンドにて行なう．

1. 以下のコマンドを実行し，CA のディレクトリを作成する．

CA ディレクトリの作成

```
# cd /etc/ssl
# mkdir CA
```

2. CA が証明書などを保存するディレクトリを作成する．

証明書などの保存用ディレクトリの作成

```
# mkdir newcerts
```

3. ディレクトリ「CA」に，証明書を番号づけて管理する上で必要となる，シリアルファイルを作成する．

シリアルファイルの作成

```
# cd CA
# echo 01 > serial
```

4. 証明書発行情報が書かれるデータベースファイル「index.txt」を作成する．

index.txt の作成

```
# touch index.txt
```

5. /etc/ssl/openssl.cnf の CA のセクションの編集を行なう．

#

```
cd ..
# vi openssl.cnf

[CA_default]
dir = /etc/ssl
database = $dir/CA/index.txt
certificate = $dir/certs/cacert.pem
serial = $dir/CA/serial
private_key = $dir/private/cakey.pem
```

以上の編集が完了後，Shift + ZZ を入力し，保存する．

以上で認証局の生成は完了である．

4.7 CA 用証明書と秘密鍵の生成

CA の証明書と秘密鍵を生成する．これはルート認証局となるので，自己署名を行なう．

1. 秘密鍵は openssl genrsa コマンドを用いる．-out で鍵ファイル名を設定し，鍵の長さは 2048 とする．

秘密鍵の生成

```
# openssl genrsa -out cakey.pem 2048
```

2. 対応する証明書署名要求を作成する．以下のコマンドを実行すると，証明書の内容を尋ねられるため，今後用いられる予定の除法を入力する．

証明書署名要求の作成

```
# openssl req -new -key cakey.pem -out careq.pem

（以下は証明書の内容である）
Country Name : JP
State or Province Name : Kochi
Locality : Kami-shi
Organization Name : Kochi University of Technology
Common Name : server4.g4.info.kochi-tech.ac.jp
（残りの内容は Enter キーでスキップする）
```

3. 秘密鍵を /etc/ssl/private に配置し，careq.pem に自己署名を行なう．

careq.pem へ自己署名

```
# mv cakey.pem private/
#openssl ca -in careq.pem -out cacert.pem -selfsign -days 365
-cxtensions v3_ca -batch
```

4. 作成された CA のルート証明書 cacert.pem を PEM 形式から DER 形式に変換する．

ルート証明書の形式変換

```
# openssl x509 -in cacert.pem -outform DER -out cacert.der
```

5. PEM 形式の証明書は /etc/ssl/certs へ移動させる．

PEM 形式証明書の移動

```
# mv cacert.pem certs
```

以上で CA 用証明書と秘密鍵の生成は完了である．

4.8 ルート証明書のインストール

ブラウザにて証明書のインストールを行なう．

1. ブラウザから http:/g4.info.kochi-tech.ac.jp/cacert.der にアクセスし，ルート認証局の証明書としてインストールする．
2. インターネットエクスプローラーを開き，URL 検索欄にて,「http:/g4.info.kochi-tech.ac.jp/cacert.der」と入力し，検索ボタンをクリックする．
3. Windows の場合，文字や数の羅列が表示されるため，右上の「ツール」をクリックし，ファイル＞「名前を付けて保存」をクリックする．
4. ファイル名を「cacert_der」から「cacert.der」へと変更し，ファイルの種類を「テキストファイル（*txt)」に設定した後，保存先をデスクトップにして「保存」をクリックする．
5. 再度「ツール」をクリックし，インターネットオプション＞コンテンツ＞証明書＞信頼されたルート証明機関を指定し,「インポート」をクリックする．
6. 「証明書のインポート ウィザードの開始」と表示されるため,「次へ」をクリックする．
7. インポートする証明書ファイルを尋ねられるため,「参照」をクリックし，デスクトップを指定，参照を「すべてのファイル」とすると,「cacert.der」が表示されるため，ダブルクリックする．
8. ファイルを指定後,「次へ」をクリックする．
9. 「証明書をすべて次のストアへ配置する：信頼されたルート証明機関」のように，もう一度確認されるため,「次へ」をクリックする．
10. インポート完了後,「完了」をクリックする．

以上で，証明書のインストールは完了である

4.9 Apache のサーバ証明書の作成

ここでは，証明書と秘密鍵を正しく作成する．

1. 以下のコマンドのように OpenSSL コマンドを使用する．

秘密鍵の生成

```
# openssl genrsa -out server.key 2048
# openssl req -new -key server.key -out server.req
```

2. 続けて，証明書の内容を尋ねられるため，CA 用証明書と時と同様に入力するが,「Common Name」は「www.g4.info.kochi-tech.ac.jp」とする．

3. CA でサーバの証明書署名要求 server.req に署名し，server.crt を作成する．

server.crt の作成

```
# openssl ca -policy policy_anything -out server.crt -infiles server.req
```

4. もし，著名に失敗した際は，再度シリアルと index.txt を戻すなどの操作，あるいは，別のカギを作成，指定する必要がある．

5. Apache にて，秘密鍵，証明書の設定を以下のように行ない，再起動する．

秘密鍵，証明書の設定

```
# mv server.key /etc/ssl/private/
# mv server.crt /etc/ssl/certs/
# rm server.req

# vi /etc/apache2/sites-available/default-ssl.conf
SSLCertificateFile /etc/ssl/certs/server/crt
SSLCertificateKeyFile /etc/ssl/private/server.key
```

以上の編集が終了後，Shift + ZZ で保存する．

以上で Apache のサーバ証明書の作成と設定は完了である．

4.10 Wireshark によるパケットモニタ

Wireshark を用いて，HTTP，HTTPS 通信のリクエスト・レスポンスの過程や，BASIC 認証の際のパスワードのパケット内での取り扱いを確認する（パケットモニタを行なう）．

1. インターネットエクスプローラーを開き，検索欄にて「Wireshark」と入力し，検索ボタンをクリックする．

2. 検索結果のうち，上から 3 番目の「Wireshark・Download」をダブルクリックする．

3. 「Stable Release (2.2.7)」の「Windows Installer (64 bit)」をダブルクリックする．

4. 「1.as.dl.wireshark.org から Wireshark-win64-2.2.7.exe (47.1 MB) を保存または実行しますか？」と尋ねられるため,「保存」をクリックする.

5. ダウンロード完了後,「フォルダを開く」をクリックする.

6. 「Wireshark-win64-2.2.7」をダブルクリックする.

7. 「次のプログラムにこのコンピューターへの変更を許可しますか？」と尋ねられるため,「はい」をクリックする.

8. セットアップに入るため,「Next ＞」をクリックする.

9. 「License Agreement」に入るため,「I Agree」をクリックする.

10. 「Choose Components」にて項目すべてにチェックを入れ,「Next ＞」をクリックする.

11. 「Select Additional Tasks」の「Create Shortcuts」にて,「Wireshark Desktop Icon」にチェック,「File Extensions」にて,「Associate trace file extensions to Wireshark」にチェックを入れ,「Next ＞」をクリックする.

12. 「Choose a directory in which to install Wireshark.」と尋ねられるため，そのまま「Next ¿」をクリックする.

13. 「Install WinPcap?」と尋ねられるため，インストールにチェックを入れ,「Next ＞」をクリックする.

14. 「Install USBPcap?」と尋ねられるため，インストールにチェックを入れ,「Install」をクリックする.

15. WinPcap のセットアップに入るため,「Next ＞」をクリックする.

16. 「License Agreement」にて,「I Agree」をクリックする.

17. 「Installation options」にて，オプションにチェックが入っていることを確認し,「Install」をクリックする.

18. 「Finish」をクリックする.

19. ダウンロードが開始し，途中で USBPcap について尋ねられるため,「OK」をクリックする.

20. ダウンロード完了後,「Next」をクリックする.

21. Wireshark のセットアップに戻るため,「Run Wireshark 2.2.7 (64 bit)」にチェックが入っていることを確認し,「Finish」をクリックする.

22. Wireshark が起動後，ローカルエリア接続をクリックする.

23. パケットモニタに取り掛かる前に，プロキシの設定を外す.

24. スタートメニュー＞コントロールパネル＞ネットワークとインターネット＞インターネットオプション＞接続＞ LAN の設定＞ LAN にプロキシサーバーを使用する　のチェックを外し,「OK」をクリックする.

25. インターネットエクスプローラーを開き，URL 欄に「http://www.g4.info.kochi-tech.ac.jp/basic/g4.html」と入力後，検索ボタンをクリックする．

26. Wireshark にて,「http」でパケットを絞り込み，172.21.14.6 から 172.21.14.2 宛ての通信かつ，info 欄に「Get /basic」と記載されているパケットをダブルクリックする．

27. パケットの詳細情報のうち，上から 4 番目の「Hypertext Transfer Protocol」の矢印マークをクリックして，詳細表示を行なう．

28. 詳細項目のうち，上から 8 番目の「Authorization: Basic dGVzdHVzZXI6cm9vdDAw\r \n」を確認する．

以上で，HTTP 通信によるパケットモニタは終了である．続けて，HTTPS 通信によるパケットモニタを行なう．

1. デスクトップの Wireshark をダブルクリックし，起動後，ローカルエリア接続をクリックする．

2. インターネットエクスプローラーを開き，URL 欄に「https://www.g4.info.kochi-tech.ac.jp/basic/g4.html」と入力後，検索ボタンをクリックする．

3. 「ssl」でパケットを絞り込むと，TLS のパケットが表示されるため，172.21.14.6 から 172.21.14.2 宛てのパケットをダブルクリックし，詳細表示を行なう．

4. 詳細表示に「Secure Sockets Layer」という項目があり，http パケットのように「Hypertext Transfer Protocol → Authorization」が存在しないことを確認する．

以上で，Wireshark を用いた https のパケットモニタは完了である．

5 考察

SSH の公開鍵認証では，認証局による署名の仕組みはなく，安全で信頼性が確保できる手段であらかじめ公開鍵を送ることを求められる．具体的には，サーバの管理者宛てに，自分の公開鍵を電子メールで添付し，サーバの管理者はその文面ややりとり，電話などの別の手段での確認などを経て，サーバの所定の保存場所に，認定された公開鍵として保存される．仮にその伝達手段を悪意をもった第 3 者から見られていた場合，望ましくない状況へと陥る可能性がある．したがって，大切な情報を扱う際には，電子媒体ではなく，手紙や口頭で伝達すべきであり，電子メールを利用する場合には，一通のメールを盗み見られてもわからないように，数回に分けた送受信を経て，伝達するべきである．

参考文献

[1] 竹下隆史，村山公保，荒井透，苅田幸雄，“ マスタリング TCP/IP — 入門編— 第 5 版，” 株式会社オーム社，2014．.

[2] 鈴木朋夫，“ Get! Comp TIA Security+ セキュリティ社会の必修科目（試験番号 : SY0-401）”，翔泳社，2015.

[3] 岩田彰，“インターネット暗号化技術 -PKI，RSA，SSL，S/MIME，etc．-”，株式会社ソフト・リサーチ・センター，2002.

[4] シスコシステムズ，“シスコ ネットワーキングアカデミー CCNA1 受講ガイド，”ソフトバンクパブリッシング株式会社，2005.