**Activity 4 Guide: Wireshark Network Packet Capture Demonstration**

Table of Contents

[Wireshark Network Packet Capture Demonstration 2](#_Toc175302330)

[Build the Login Application 2](#_Toc175302331)

[Create a New Spring Boot Project 3](#_Toc175302332)

[Create a UserModel 8](#_Toc175302333)

[Create a User Controller 9](#_Toc175302334)

[Create Two Template Pages 10](#_Toc175302335)

[Run and Test the Application 11](#_Toc175302336)

[Wireshark Instructions 12](#_Toc175302337)

[How You Would Use Wireshark on a Network (Not Just Localhost) 20](#_Toc175302338)

[About Internet Protocol Exchanges 21](#_Toc175302339)

[Other Common Protocols 24](#_Toc175302340)

[How to Fix the Application's Vulnerability 24](#_Toc175302341)

[What Are SSL Certificates? 24](#_Toc175302342)

[How SSL Certificates Help Encrypt Data 25](#_Toc175302343)

[How Do We Know if an SSL Certificate is Valid? 26](#_Toc175302344)

[Error Messages from Self-Signed Certificates 27](#_Toc175302345)

[Have SSL/TLS Certificates Ever Been Compromised? 28](#_Toc175302346)

[Generate a Certificate 28](#_Toc175302347)

[Java Path Side Note 28](#_Toc175302348)

[About the SSL Create Statement 29](#_Toc175302349)

[About These Settings 31](#_Toc175302350)

[Repeat the Wireshark Process with the SSL-Enabled Application 35](#_Toc175302351)

[Explanation of the Wireshark Data 37](#_Toc175302352)

[The SSL Certificate as a government backdoor 38](#_Toc175302353)

[Current SSL Security Practices 39](#_Toc175302354)

[End-to-End is More Secure Than SSL/TLS 39](#_Toc175302355)

[What You Learned 39](#_Toc175302356)

[Check for Understanding 40](#_Toc175302357)

[Deliverables 43](#_Toc175302358)

## Wireshark Network Packet Capture Demonstration

**Overview**

Application security is both an internal and external challenge. The way in which an application sends data across a network will determine how secure the data remains. This lesson will demonstrate that unencrypted network requests are vulnerable to packet sniffers, as seen in Figure 1.

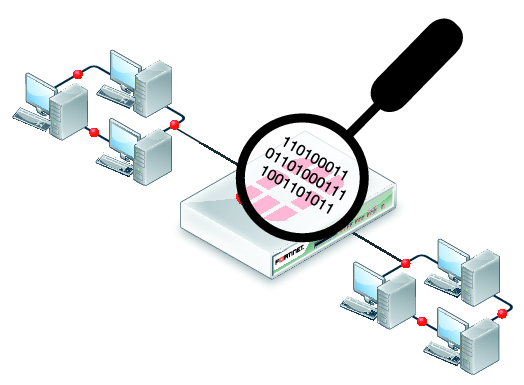


Figure 1 Packet Sniffers capture all traffic on a network and save it to a log that can be examined.

Wireshark is a popular hacking tool as well as network engineer's diagnostic tool for viewing communication over a computer network. For engineers, seeing error messages, response times and traffic routes can be valuable to improving the performance of network devices. For hackers, network traffic may reveal sensitive information.

1. Tools Needed: (a) Java IDE (b) Wireshark application.
2. Using this tutorial, create a very simple application that has a login form and form processor. The Java application will run a web server that operates a login screen and results screen.
3. We will capture the network traffic using Wireshark and steal credentials from an unencrypted login session.
4. Finally, we will convert the unencrypted http traffic to an encrypted https protocol and compare the traffic captured by Wireshark.

# Build the Login Application

## Create a New Spring Boot Project

1. Create a new Java project with Spring Boot Start Web and Spring Boot Dev Tools. Depending on your IDE, your project creation process will be slightly different. Here are three Spring Boot application starter routines shown using one of three options.
   1. (a) VS Code
   2. (b) Ellipse
   3. (c) the online Spring Boot Initializer

**Build a Spring Boot Application Using VS Code**

1. In VS Code, ensure Java is installed and proper extensions are installed.
   1. You can confirm that Java is installed by typing **java --version** at the command prompt, as seen in Figure 2.

A black screen with white text

Description automatically generated

Figure 2 Java version 17 is installed on this Windows computer.

* 1. In Visual Studio Code, the "Extension Pack for Java" from Microsoft will provide a set of popular extensions for Java, as seen in Figure 3.

A screen shot of a computer

Description automatically generated

Figure 3 Microsoft's Extension Pack for Java will install a set of popular extensions that enable development.

1. Click the Create Java Project button shown in Figure 4.

A screenshot of a computer

Description automatically generated

Figure 4 Once the Java Extension Pack is installed, VS Code has a "Create Java Project" button in the main window.

1. Choose Spring Boot (provided by Spring Initializer), as shown in Figure 5.

A screenshot of a computer

Description automatically generated

Figure 5 Starting a Spring Boot project in VS Code.

1. Select Maven Project, as shown in Figure 6.

A screenshot of a computer

Description automatically generated

Figure 6 Selecting Maven for the build tool.

1. Choose the current version of Spring Boot, as shown in Figure 7.

A screenshot of a computer

Description automatically generated

Figure 7 Selecting newest version of Spring Boot.

1. Select Java, as shown in Figure 8.

A screenshot of a computer

Description automatically generated

Figure 8 Selecting Java.

1. Give your Group Id a name, use a reverse internet domain name convention, seen in Figure 9. Use your name or a business name you own.

A screenshot of a computer

Description automatically generated

Figure 9 Choosing your name for the Group ID.

1. Give the Artifact Id a name that identifies the name of this application, as shown in Figure 10.

A screenshot of a computer

Description automatically generated

Figure 10 Using "loginapp" for the artifact id.

1. Select Jar package type, as shown in Figure 11.

A screenshot of a computer

Description automatically generated

Figure 11 Selecting Jar as the package type.

1. Select the Java version installed on your computer, as shown in Figure 12.

A screenshot of a computer

Description automatically generated

Figure 12 Selecting 17 as the Java version installed on this computer.

1. Choose **Spring Web**, **Thymeleaf**, and **DevTools** dependencies, as shown in Figure 13.

A screenshot of a computer

Description automatically generated

Figure 13 Three dependencies chosen.

If you completed the VS Code setup, you may skip this section.

**Eclipse project creation.**

1. Start Eclipse for Java.
2. File > New ? Project, as shown in Figure 14.

A screenshot of a computer

Description automatically generated

Figure 14 Choosing new project in Eclipse.

1. Choose Maven Project, as seen in Figure 15.

A screenshot of a project

Description automatically generated

Figure 15 Selecting Maven project type.

1. Maven-archetype-quickstart, as seen in Figure 16.

A screenshot of a computer

Description automatically generated

Figure 16 Searching for maven archetype QuickStart.

1. Name the Group ID, Artifact ID, as seen in Figure 17.

A screenshot of a project

Description automatically generated

Figure 17 Naming the Group Id and Artifact Id values.

1. Confirm the project creation, as shown in Figure 18.

A screenshot of a computer program

Description automatically generated

Figure 18 Responding Y in the command prompt.

1. As success message will show up, as seen in Figure 19.

A screen shot of a computer code

Description automatically generated

Figure 19 Viewing the finished results of building a new project.

1. Manually add the 3 dependencies (Spring Boot Starter Web, Spring Boot Devtools, Thymeleaf) the POM.XML file by searching Maven Central for the dependencies, as seen in Figure 20.

A screenshot of a computer program

Description automatically generated

Figure 20 A look at the POM file with new dependencies.

**Online Spring Boot Initializer**

You can create the entire Spring Boot project using the online initializer tool regardless of the IDE you are using, as seen in Figure 21.

1. Navigate to <http://start.spring.io>
2. Choose the following options:
   1. Maven
   2. Java
   3. Newest version of Spring Boot
   4. Group: edu.yourname
   5. Artifiact: loginapp
   6. Name: loginapp
   7. Description: App to test ssl certificates
   8. Package name: edu.yourname.loginapp
   9. Packaging: Jar
   10. Java: 17
   11. Dependencies: Spring Boot DevTools, Spring Web, Thymeleaf

A screenshot of a computer

Description automatically generated

Figure 21 Starting a Spring Boot project using the online Spring Initializer service at <http://start.spring.io> Spring Boot Dev Tools , ThymeLeaf and Spring Web dependencies are selected.

1. The starter project is zipped and automatically downloaded.
2. Unzip the project and open it with your favorite IDE.

# Application Development

Now that you have a completed starting project, continue with the following steps:

## Create a UserModel

1. Create a new file UserModel.java
2. Give it properties id, username, and password.
3. Create getters, setters, constructor, and toString methods, as seen in Figure 22.

A screenshot of a computer

Description automatically generated

Figure 22 UserModel used in the login app.

## Create a User Controller

1. Create a new file UserController.java
2. Annotate the class as a @controller
3. Create two methods showLoginForm and loginUser, as shown in Figure 23.
4. showLoginForm displays the loginform.html template.
5. loginUser checks the credentials and returns the results to loginresult.html template.

A screenshot of a computer screen

Description automatically generated

Figure 23 UsersController has two methods – showLoginForm (GET) and loginUser (POST).

## Create Two Template Pages

1. loginform.html displays a data entry form for the user object, as seen in Figure 24.

A screenshot of a computer

Description automatically generated

Figure 24 Login Form HTML is bound to the UserModel class.

1. loginresult.html displays the user data with a message, as seen in Figure 25.

A screenshot of a computer

Description automatically generated

Figure 25 The loginresult page shows both the username and password properties used in the login form.

## Run and Test the Application

1. The login form displays a data entry form, as shown in Figure 26.

A screenshot of a computer

Description automatically generated

Figure 26 Login form at localhost:8080.

1. The result page shows a success or failure plus displays the credentials that were entered, as seen in Figure 27.

A screenshot of a computer

Description automatically generated

Figure 27 Successful login with hard-coded user and password.



* Take a screenshot of the running application at this point.
* Paste the image into a Microsoft Word document.
* Put a caption below the image explaining what is being demonstrated.

### Wireshark Instructions

1. Download and install [Wireshark](http://wireshark.org).
2. The application might ask you to install **ChmodBPF** and relaunch the app.
3. Windows will need a module called npcap. [Npcap](http://npcap.com) is a packet capture library for Windows that includes a "loopback adapter" that allows you to capture loopback traffic with Wireshark. During Wireshark installation options, you probably will be asked if you wish to add npcap, as seen in Figure 28.

A screenshot of a computer error message

Description automatically generated

Figure 28 Npcap installation option is checked by default.

1. Launch the app.
2. Select a network interface to monitor.

* **Loopback for Mac** - Select the network card that you want to monitor. In this case, you want to monitor "**Loopback**" because the website we are looking at is **localhost**. The loopback interface is a virtual network interface used by a computer to send traffic to itself. It's typically associated with the IP address 127.0.0.1 for IPv4 and ::1 for IPv6. On most systems, the loopback interface is named lo (Linux) or lo0 (macOS). It is commonly used for testing and development purposes. For example, when a web server running on your machine is accessed via http://localhost, the traffic is handled by the loopback interface.
* **Windows** Unlike Linux and macOS, Windows does not provide a direct, easily accessible loopback interface in network monitoring tools like Wireshark. The **Npcap** application provides a work-around for the loopback feature.

1. Set the capture for the loopback interface, as shown in Figures 29 and 30.

A screenshot of a computer

Description automatically generated

Figure 29 Windows version of Wireshark shows "Adapter for loopback traffic."a

A screenshot of a computer

Description automatically generated

Figure 30 Mac version shows a "Loopback: lo0" interface.

1. If the localhost web server is running, you will likely see some network traffic results, as seen in Figure 31.

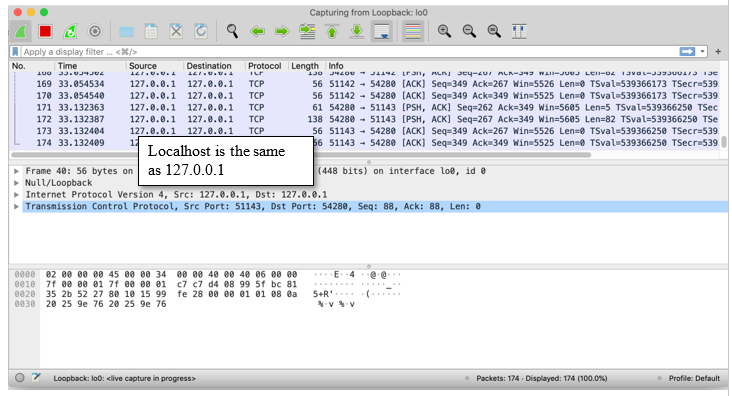


Figure 31 Localhost traffic is being tracked.

1. Click the shark fin to start capturing packets, as shown in Figure 32.

A screenshot of a computer

Description automatically generated

Figure 32 Start button for Wireshark tracking.

1. While Wireshark is capturing packets, switch to the local server website and perform a login, as aseen in Figure 33.

A screenshot of a computer

Description automatically generated

Figure 33 Login form at localhost:8080.

1. The result page, as seen in Figure 34, shows a success or failure, and displays the credentials that were entered.

A screenshot of a computer

Description automatically generated

Figure 34 Successful login with hard-coded user and password.

1. Return to Wireshark and stop capturing by clicking on the red stop button, seen in Figure 35.



Figure 35 Wireshark button to stop monitoring.

1. You should see a long list of text. Each line in the report represents a packet of communications between your computer and the localhost web server, as seen in Figure 36.

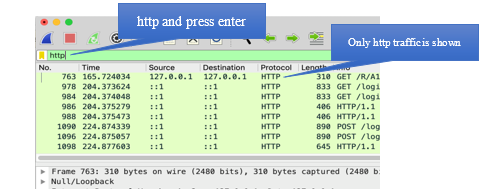


Figure 36 Applying the http filter results in hiding many of the captured packets.



* Take a screenshot of the running application at this point.
* Paste the image into a Microsoft Word document.
* Put a caption below the image explaining what is being demonstrated.

1. Apply a filter to show only "http" packets.
2. Select one of the http packets with a 200 result and text/html content. In this example, capture number 9 contains the results of the GET request that displays the login form.
3. Expand the "Line-based text data" to see the contents of this request, as seen in Figure 37.

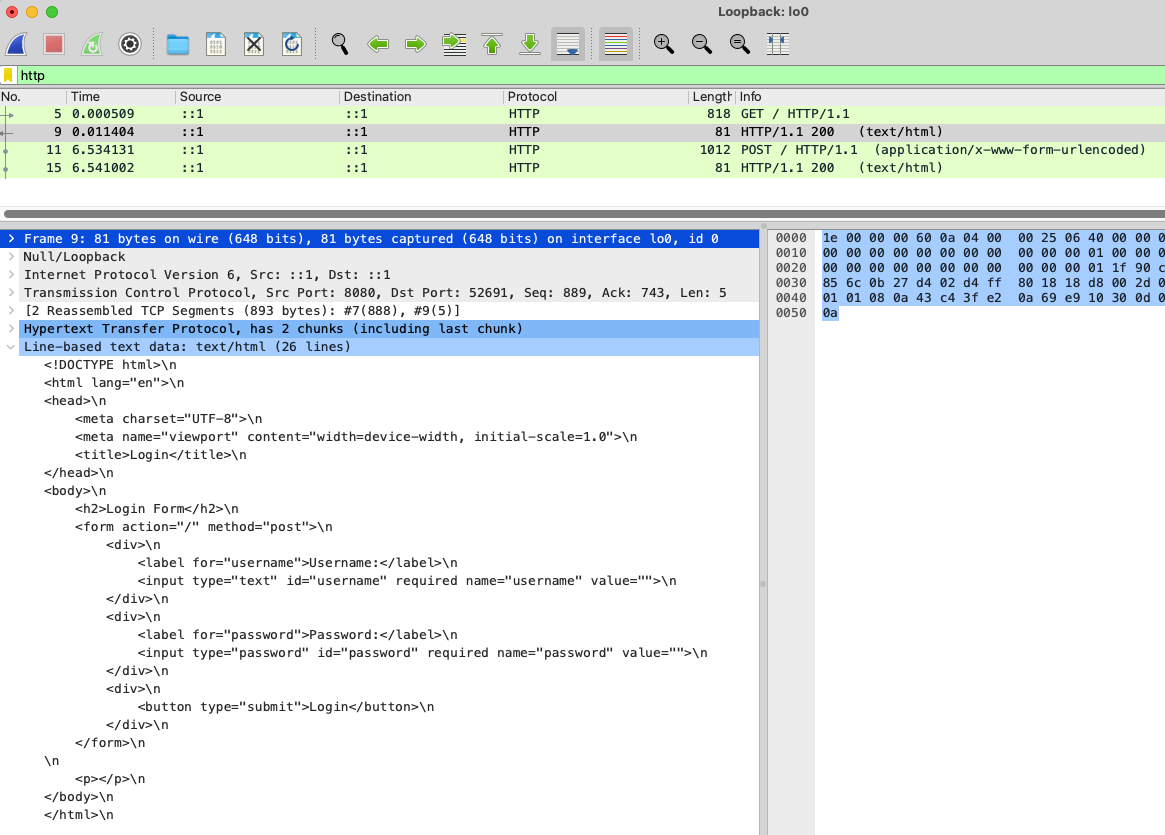


Figure 37 Expand the “Line based text data”.

1. You can retrieve a more readable version of the packet by right-clicking the line in the log > Follow > HTTP Stream, as shown in Figures 38 and 39.

A screenshot of a computer

Description automatically generated

Figure 38 Show all Packets Associated with the Login Form's GET Event.

A screenshot of a computer

Description automatically generated

Figure 39 Following the GET request displays the contents of the web page, cookie information and other details about the request.



* Take a screenshot of the running application at this point.
* Paste the image into a Microsoft Word document.
* Put a caption below the image explaining what is being demonstrated.

1. Select the POST request and expand the contents. You should be able to see the form items for username and password by opening the items in the bottom window, as shown in Figure 40.

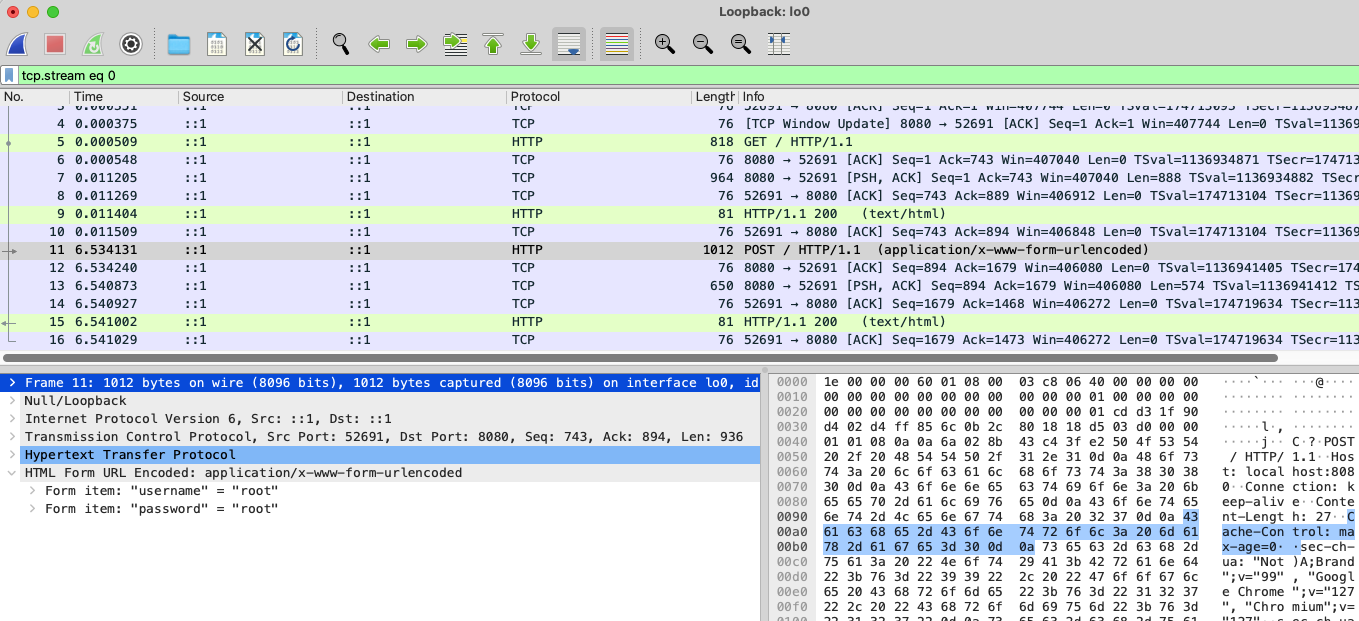


Figure 40 The POST request is the packet generated by the login form. It contains the data from the username and password data entry fields.

1. You can also display the entire packet in a more readable format by right-clicking > Follow > HTTP stream, as seen in Figure 41.

A screenshot of a computer

Description automatically generated

Figure 41 Following the stream of data from the POST request.

1. You can see the http traffic, which includes the cookie information about the form contents, as seen in Figure 42.

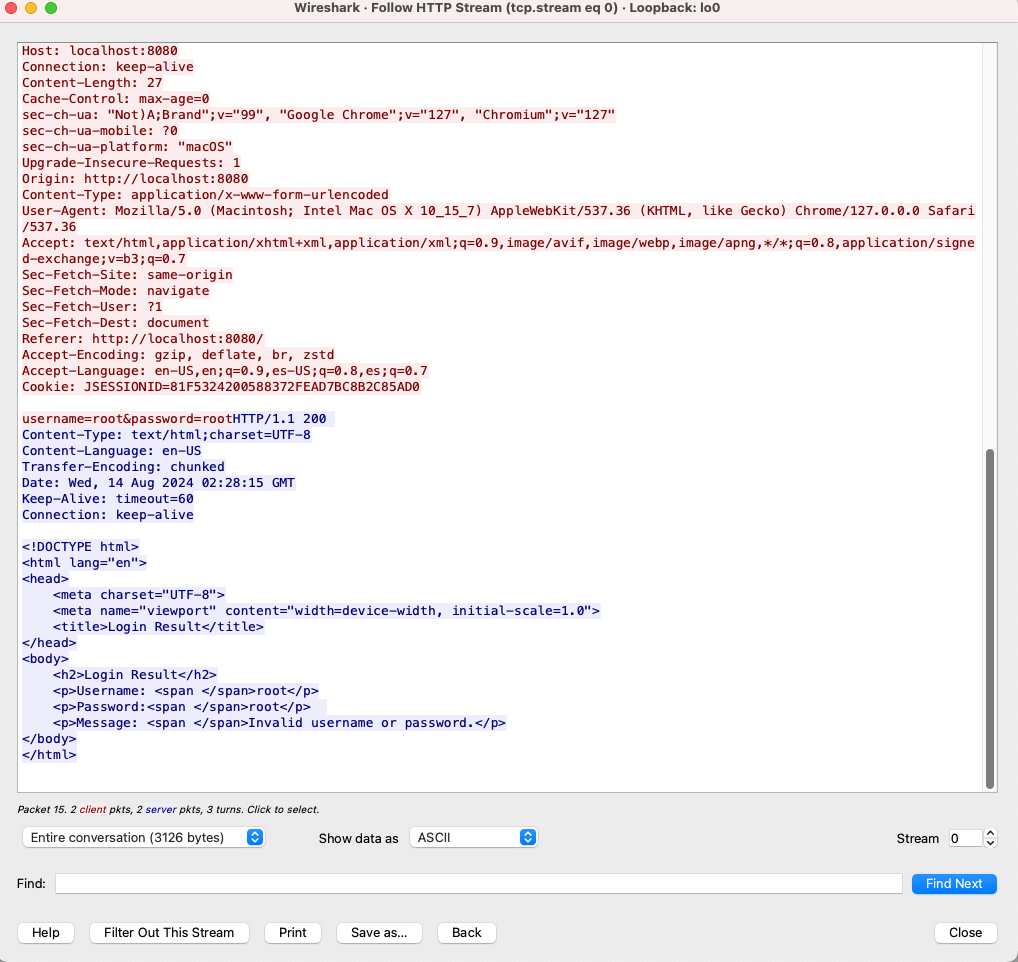


Figure 42 Following the stream from the POST request reveals the form data including the username and password values.



* Take a screenshot of the running application at this point.
* Paste the image into a Microsoft Word document.
* Put a caption below the image explaining what is being demonstrated.

### How You Would Use Wireshark on a Network (Not Just Localhost)

The demonstration we performed in this activity was conducted on a localhost website. In the real world, you would only need to change a few things to do the same task including the following:

1. **Wi-Fi NIC**. When starting Wireshark, select the Wi-Fi network card instead of Loopback. The traffic between your computer and one on the network must travel through the Wi-Fi card.
2. **Monitor Mode**. To capture packets from other computers in the area, set your network adapter to "Monitor Mode." Any computer in a public café could potentially be monitoring your network traffic. Usually, the network card ignores any packets that are not indented for itself. If your network card supports monitor mode, then you can capture packets that travel between the Wi-Fi access point and your neighbor’s computers. Network cards in laptops frequently do not have a monitor mode. A USB Wi-Fi access card is usually required.
3. **Encryption.** Most websites today send all data in an encrypted format. Capturing packets from an https session are scrambled.

## 

## About Internet Protocol Exchanges

Every time a service needs to communicate with a client over digital networks, a protocol must be designed to make the transaction work smoothly. In human life, we have developed protocols of communication that help make the process standardized.

*“Hello?”*

*“Hi, this is \_\_\_\_\_\_\_ from the \_\_\_\_\_\_\_. How are you today?”*

*“I’m fine, thank you. What can I do for you?”*

*(Payload of the conversation)*

*“OK. Thank you calling. That helps a lot.”*

*“Glad to help. Good bye”*

In digital communications the clients and servers work in a similar manner. Each transaction usually begins with a "hello" packet. There are many "Acknowledge" packets, which are essentially "OK" messages. Finally, a "Goodbye" packet ends the transaction.

Wireshark enables you to see many conversations that are taking place simultaneously on a network in a variety of protocols.

## Other Common Protocols

Here is a list of common protocols. Complete the table by researching the name and a one sentence description of each protocol.

|  |  |  |
| --- | --- | --- |
| Initials | Name | What It Is Used For |
| HTTP | Hypertext Transfer Protocol | Send hypertext pages for world wide web applications. |
| TCP |  |  |
| SNMP |  |  |
| FTP |  |  |
| SMTP |  |  |
| IMAP |  |  |
| POP3 |  |  |
| DNS |  |  |
| SSH |  |  |
| UDP |  |  |
| RDP |  |  |
| VoIP |  |  |
| DHCP |  |  |
| LDAP |  |  |
| TLS |  |  |

# How to Fix the Application's Vulnerability

To protect data in transit, we need to add encryption to the login application. The **http** protocol is unencrypted and therefore vulnerable to network sniffers like Wireshark. The **https** protocol was developed to provide secure transport. At first, https was used only for sensitive information such as logins or financial transactions. Unencrypted data is less computationally expensive. Later, https became the default mode to transfer all data.

Here are the tasks we must do to transmit data in encrypted format:

1. Create an SSL / TLS certificate.
2. Configure the application to use the https protocol.
3. Direct old http requests to https.

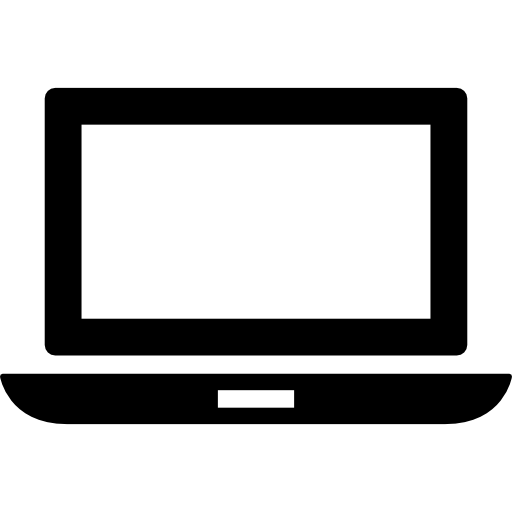
# About Certificates

### What Are SSL Certificates?

SSL (secure sockets layer) certificates are digital certificates that provide a way to encrypt communication between a user's browser and a web server. SSL was replaced by TLS in 2015, but the protocol remains to be called SSL. SSL certificates are like the public and private keys used in the GPG exercise done earlier in this course.

### How SSL Certificates Help Encrypt Data

* When a client connects to a server via HTTPS, an SSL handshake occurs. During this handshake, the client and server use the public and private keys to create a shared encryption key that is used for the duration of the session, as seen in Figure 43.
  + The server presents its SSL certificate to the client. The server's public key is part of the certificate.
  + The client and server agree on an encryption method.
  + A unique session key is created for this specific connection.
  + This session key is used to encrypt and decrypt the data exchanged during the session.



SYN (Synchronize)

ACK

SYN ACK

Client Hello

Client Key Exchange

Server Finished

Server Hello Certificate

Certificate verification

Response 200 OK

POST

GET

Response 200 OK

Establish TCP Connection

SSL Handshake

Encrypted Session

Figure 43 Internet protocols are very "chatty," meaning that two-way communications require many acknowledgements, hello and goodbye messages.

### How Do We Know if an SSL Certificate is Valid?

**Any computer can generate an SSL. In fact, in the next steps we will create our own certificate. However, on a real website you need to choose from a list of "white listed" certificate providers.** A **certificate Aathority (CA)** is an organization or entity responsible for issuing digital certificates. Some well-known CA organizations include **Symantec, Comodo**,and **GoDaddy**, as seen in Figures 44 and 45.These companies are recognized and trusted by Apple, Google, Microsoft, and other leading tech companies. As of August 2024, 133 CA organizations are trusted by Microsoft Windows. The US Federal Government manages the **Federal PKI** for use with all government applications.

A screenshot of a computer

Description automatically generated

Figure 44 GoDaddy will issue a trusted SSL certificate for a fee. Our application is not being used in production, so we will not be paying these prices.

A screenshot of a computer

Description automatically generated

Figure 45 A number of nonprofit organizations, such as "Let's Encrypt," provide free TLS/SSL certificates.

* **Validity Period**: SSL certificates have a validity period, often one or two years. They must be renewed before they expire to continue ensuring secure connections.

### Error Messages from Self-Signed Certificates

Self-signed certificates are SSL/TLS certificates that are generated and signed by the organization itself, rather than a trusted CA. These certificates can be used for testing and internal purposes but are not trusted by browsers by default because they do not come from a recognized CA.

**Common Errors with Self-Signed Certificates**:

* **"Your connection is not private"**: Seen in Figure 46, this error occurs because the browser cannot verify the certificate's authenticity.
* **"Invalid certificate" or "Untrusted certificate"**: This indicates that the certificate is not from a trusted CA.
* **"Self-signed certificate in the certificate chain"**: This error specifically indicates that the certificate is self-signed and not trusted.

A screenshot of a computer

Description automatically generated

Figure 46 Error message from Chrome showing an error from a self-signed TLS certificate.

### Have SSL/TLS Certificates Ever Been Compromised?

Yes, SSL/TLS certificates have been compromised in the past, though such incidents are rare. Some notable incidents include:

* **DigiNotar Hack (2011)**: Attackers compromised the Dutch CA DigiNotar and issued fraudulent certificates, which were used in man-in-the-middle attacks.
* **Comodo Hack (2011)**: Attackers were able to issue fraudulent certificates through a reseller of Comodo, another CA.
* **Symantec Certificate Misissuance (2017)**: Google and Mozilla distrusted Symantec-issued certificates after discovering that Symantec had improperly issued SSL/TLS certificates.

### Generate a Certificate

1. In the terminal, type the following command, seen in Figure 47.

keytool -genkeypair -alias springboot -keyalg RSA -keysize 2048 -storetype PKCS12 -keystore keystore.p12 -validity 3650

A close-up of a computer screen

Description automatically generated

Figure 47 Terminal in VS Code with the keytool command.

## Possible Java Errors

**keytool** is part of the Java Development Kit.

If Java is not installed or the JDK is not part of the path variables, the keytool will not start. You can either navigate to the Java\bin folder and run the keytool from there (as pictured in Figure 48) or you can configure the path variables in Windows and run the keytool command from any folder.

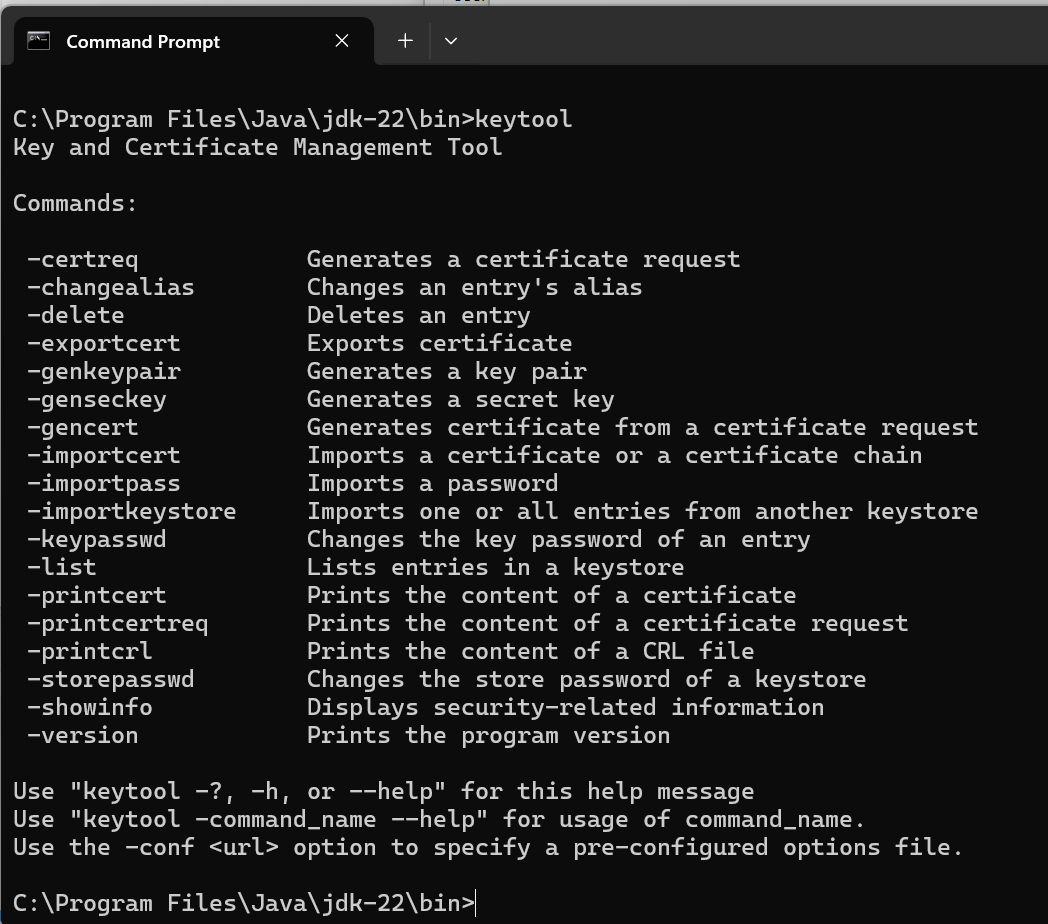


Figure 48 Running keytool from the jdk directory (C:\Program Files\Java\jdk-22\bin) without regard to the path environment variable set.

## Path Variable

Usually, the Java installation process automatically adds **java.exe** to the Windows path. However, some installations may skip the path configuration during setup. Below, in Figures 49 through 51, is an example of a computer with several development tools in the execution search path including Java.

A screenshot of a program

Description automatically generated

Figure 49 Java path is set to a standard location in c:\program files\java\jdk-22\bin

A screenshot of a computer

Description automatically generated

Figure 50 Java path is set to the jdk installed with Eclipse. This folder is much more difficult to find than the standard jdk installation.

A screenshot of a computer

Description automatically generated

Figure 51 The java.exe file is found deep within the Eclipse folder structure.

## About the SSL Create Statement

keytool -genkeypair -alias springboot -keyalg RSA -keysize 2048 -storetype PKCS12 -keystore keystore.p12 -validity 3650

Here's a breakdown of each part of the SSL generator.

* **-genkeypair**: Generates a public-private key pair.
* **-alias springboot**: Specifies the alias for the key pair in the keystore.
* **-keyalg RSA**: Specifies the algorithm to be used for generating the key pair, in this case, RSA.
* **-keysize 2048**: Specifies the size of the key, which is 2048 bits.
* **-storetype PKCS12**: Specifies the type of keystore to be created, PKCS12 is a standard for storing cryptographic information.
* **-keystore keystore.p12**: Specifies the name of the keystore file to be created.
* **-validity 3650**: Specifies the validity period of the certificate in days, which in this case is 10 years.

Figure 52 shows a new file has been added to the application after the keystore command has been executed.

A screenshot of a computer

Description automatically generated

Figure 52 keystore.p12 is displayed in VS Code after the keystore command is executed.

1. Move the keystore.p12 file into the resources folder of the login app, as seen in Figure 53.

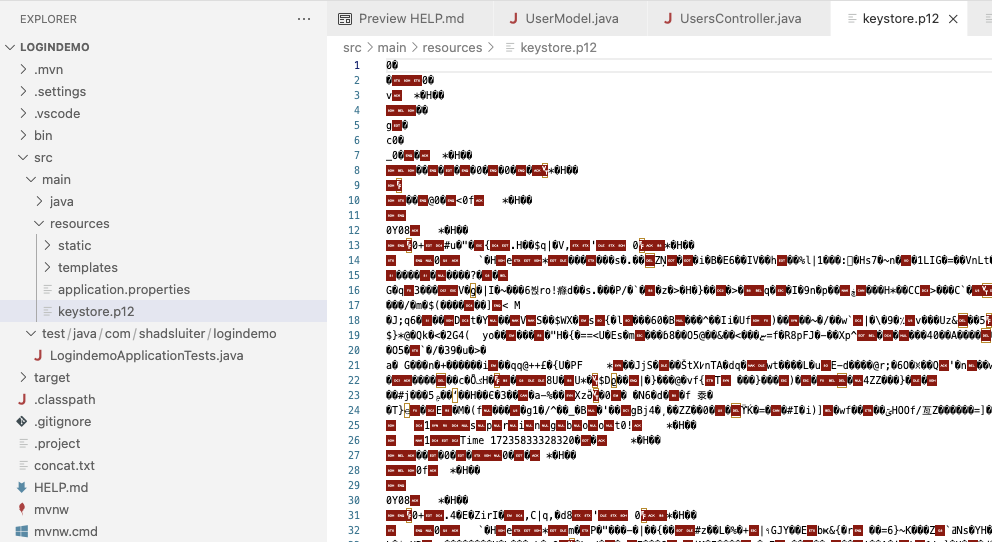


Figure 53 the keystore.p12 file has been placed into the resources folder.

1. Add the following data to the **application.properties** file, as shown in Figure 54.

**server.port=8443**

**server.ssl.key-store=classpath:keystore.p12**

**server.ssl.key-store-password=your\_keystore\_password**

**server.ssl.key-store-type=PKCS12**

**server.ssl.key-alias=springboot**

**A screenshot of a computer

Description automatically generated**

Figure 54 Location of the application.properties file.

## About These Settings

server.port=8443

* By default, Spring Boot applications run on port 8080. However, setting server.port=8443 changes the port to 8443, which is a common default port for HTTPS traffic. If you access the application in a browser, you would go to https://localhost:8443.

server.ssl.key-store=classpath:keystore.p12

* This property specifies the location of the SSL keystore file, which contains the server's private key and the SSL certificate. The classpath: prefix indicates that the keystore.p12 file is located in the classpath of the application. The classpath is the path where application resources and libraries are stored, often found in the src/main/resources directory in a Spring Boot project.
* PKCS12 is a widely used binary format, as seen in Figure 55, for storing cryptographic keys, so p12 is a common file extension.

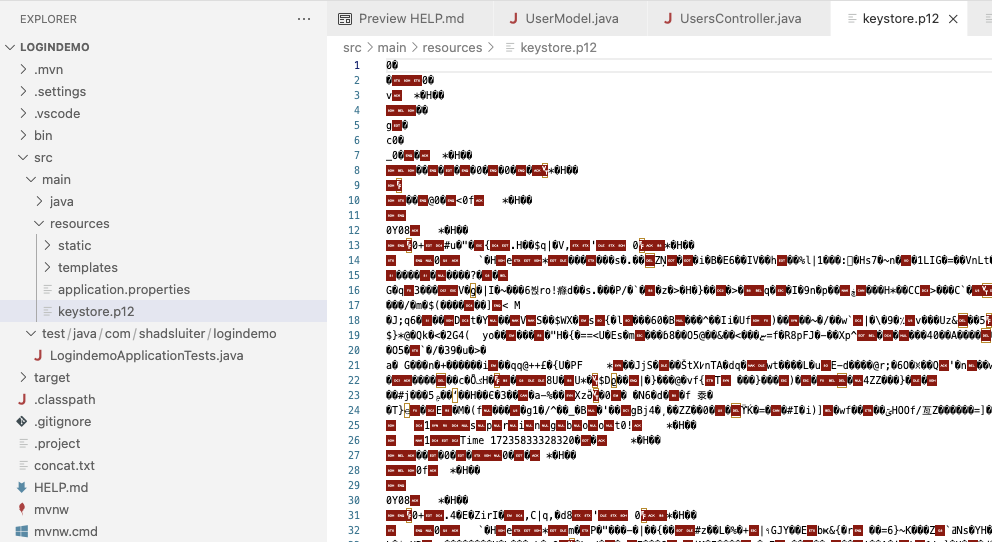


Figure 55 The contents of kestore.p12 will be a binary file, not ASCII, that is unreadable with standard text editors.

server.ssl.key-store-password=your\_keystore\_password

* In the example code, you will see "123456" as the password. Obviously, a long and complex password would be more appropriate. The password should be kept secure and not hard-coded in the configuration file in a production environment. You might want to externalize it using environment variables or a secrets management tool.

server.ssl.key-store-type=PKCS12

* PKCS12 refers to the PKCS #12 standard, which is commonly used for storing cryptographic keys and certificates in a single, portable file. PKCS12 files usually have a .p12 or .pfx extension.

server.ssl.key-alias=springboot

* This property specifies the alias name of the key within the keystore that the server should use for SSL.

1. **Ensure that Keystore is in the Classpath.**

Move the keystore.p12 file to src/main/resources in your project, as shown in Figure 56, so that it is available in the classpath of the application.

A screenshot of a computer

Description automatically generated

Figure 56 Resources folder is the best place to put keystore.p12.

1. **Update URLs in HTML Files.**

Since you're using HTTPS now, update any hardcoded URLs in your HTML files to use https:// instead of http://, if needed. However, typically Spring Boot will automatically redirect HTTP requests to HTTPS if configured correctly.

1. **Run the application.**
2. Navigate to https://localhost:8443/ to access your application, as seen in Figure 57.

A screenshot of a computer

Description automatically generated

Figure 57 The standard 8080 port is no longer active with this application.

A screenshot of a computer

Description automatically generated

Figure 58 Chrome warns me that the site is not to be trusted. The certificate is self-issued.

1. Click "Advanced," shown in Figure 58, and click the link, "Proceed to localhost (unsafe)," as seen in Figure 59.

A screenshot of a computer message

Description automatically generated

Figure 59 "Advanced" allows you to ignore the warning and show the site.

1. The website will display but with a warning in the URL area.
2. Click the "Note Secure" button, seen in Figure 60, to see details about the problem.

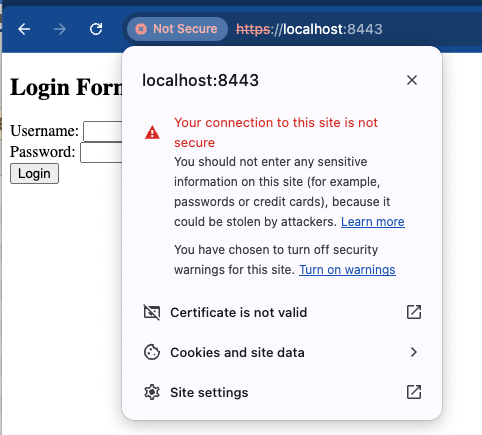


Figure 60 Chrome does not trust this certificate.

1. Click the "Certificate is not valid" link shown in Figure 60 to see more information about this website, as seen in Figure 61.

A screenshot of a computer

Description automatically generated

Figure 61 I am not a trusted certificate authority (CA) even though Grand Canyon University has a great academic reputation.



* Take a screenshot of the running application at this point.
* Paste the image into a Microsoft Word document.
* Put a caption below the image explaining what is being demonstrated.

## Repeat the Wireshark Process with the SSL-Enabled Application

Complete the following steps to demonstrate that even though the login credentials can still be captured by Wireshark, the packets are now unreadable due to SSL encryption.

* 1. Start Wireshark or restart the capture, as shown in Figure 62.

A screenshot of a computer

Description automatically generated

Figure 62 Restart capture option in Wireshark.

* 1. Navigate to <http://localhost:8443>, as seen in Figure 63.

A screenshot of a computer screen

Description automatically generated

Figure 63 Login form is shown on the https version of the application on port 8443.

* 1. Perform a login event with the proper username and password, as shown in Figure 64.

A screenshot of a computer

Description automatically generated

Figure 64 Successful login on the application.

* 1. Stop the Wireshark capture, as shown in Figure 65.

A screenshot of a computer

Description automatically generated

Figure 65 Several events were captured during the login event. No http or https protocols are shown.



* Take a screenshot of the running application at this point.
* Paste the image into a Microsoft Word document.
* Put a caption below the image explaining what is being demonstrated.
  1. The traffic protocol used by the application is no longer listed as **http** (clear text). Nor is there a protocol listed as **https**. Instead, you will see several TLSv1.2 events in the log. Wireshark labels the packets as "TLS" (or "SSL" for older versions) instead of "HTTPS" because it is indicating that the traffic is encrypted with the TLS protocol. Wireshark does not show "HTTPS" as the protocol because HTTPS is not a protocol layer on its own. Https it is just HTTP wrapped inside a secure TLS/SSL "tunnel," as shown in Figure 66.

Client

http packet

TLS packet

(http packet)

http packet

Server

Figure 66 http packets are encrypted and added to the payload of a TLS packet.

## Explanation of the Wireshark data

* The captured packets show a sequence of secure communications.
* It is impossible to determine if these occurred during a login event or some other transaction.
* The encryption provided by TLS ensures that sensitive information like usernames and passwords cannot be easily intercepted or viewed by unauthorized parties.
* Any attempt to view the packets is futile. For example, the description of packet #1 shown below is "application data," a generic term.
* Viewing the contents results in a message that describes the data as "encrypted data."
* Each TLS packets is followed by a TCP Acknowledge packet, as seen in Figures 67 and 68. The Acknowledge packet is simply a confirmation that the previous transaction occurred without communication errors.

A screenshot of a computer

Description automatically generated

Figure 67 TLS packets are labeled as "Encrypted Application Data." No passwords or usernames can be viewed.

A screenshot of a computer

Description automatically generated

Figure 68 Each TLS packet is followed by a TCP "Acknowledge" packet.

# 



* Take a screenshot of the running application at this point.
* Paste the image into a Microsoft Word document.
* Put a caption below the image explaining what is being demonstrated.

# The SSL Certificate as a Government Backdoor

Authoritarian governments such as China and Russia compel many organizations within their countries to use only government-operated CA services. This provides the government with the capability to perform man-in-the-middle surveillance on encrypted web traffic.

Communications with external servers are secure since the government does not have access to the private key of foreign SSL certificates. As a result, many external services are blocked by "The Great Firewall of China" or "Runet." Some international organizations operating within these countries may be permitted to use any CA they choose. Both governments maintain tight control over internet resources and can change policies at any time.

The United States has also participated in extensive backdoor operations related to encrypted network traffic. In addition to the famous PRISM surveillance program, brought to light by Edward Snowden in 2013, in which the government compelled technology companies to provide unencrypted data, the NSA also conducted network tapping operations that were unknown to the target companies.

MUSCULAR

From 2009 to 2013, the NSA and the British counterpart, GCHQ, ran a program called MUSCULAR, which secretly tapped into fiber optic cables connected to major tech companies in Great Britain, including Yahoo and Google. Working like a giant Wireshark packet capture, the NSA collected huge amounts of data.

At the time, tech companies viewed the fiber optic cables between their own data centers as "internal" and therefore secure. **The internal data was not encrypted**.

Early versions of SSL (3.0) and TLS (1.0) had vulnerabilities that could be exploited, which means the NSA had access to even larger amounts of data. MUSCULAR was part of Edward Snowden's leak, as shown in Figure 69.

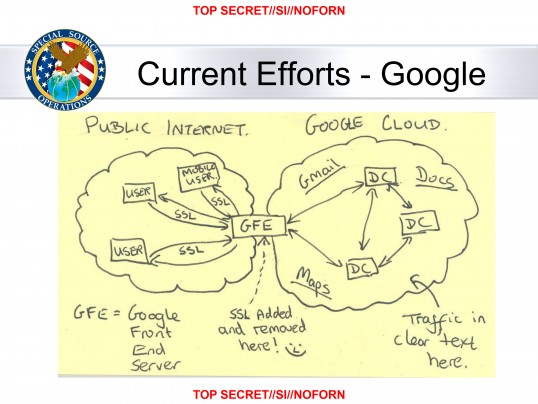


Figure 69 An NSA Power Point slide leaked by Edward Snowden describes the network configuration that existed during the time of MUSCULAR. The hand-drawn smiley face shows the obvious weakness of the design.

## Current SSL Security Practices

Since 2013, all large tech companies encrypt all data, including at-rest and in-transit and internal data, using more advanced encryption standards. However, it can be assumed that the NSA continues to conduct surveillance in many forms.

## End-to-End is More Secure Than SSL/TLS

Signal, IMessage, WhatsApp and other end-to-end communications can be considered secure even if the data is sent through China, or even if it is sent using unencrypted protocols like http. The application itself, performs the encryption process inside the phone before sending out the data. The application does not rely on network encryption.

## What You Learned

In this lesson, you learned about the importance of application security, particularly how data is transmitted across a network. Here's a summary of the key points:

1. **Building and Testing a Secure Login Application**:
   * You built a simple Java Spring Boot application that features a login form and form processor and used it to observe network traffic.
2. **Network Sniffing**
   * Wireshark can be used by both network engineers for diagnostics and hackers for malicious purposes.
   * By using Wireshark, you were able to see how unencrypted HTTP traffic exposes sensitive information like usernames and passwords.
3. **Transitioning to Secure HTTPS Communication**:
   * You learned how to create an SSL/TLS certificate and configure your Spring Boot application to use HTTPS, thereby encrypting the data in transit.
   * After enabling HTTPS, Wireshark showed that the traffic is now using TLS, making it impossible to easily intercept and read sensitive information.
4. **Understanding SSL/TLS and Certificate Authorities**:
   * You learned how SSL/TLS certificates are used during the SSL handshake to establish a secure communication channel.
   * You also explored the role of certificate authorities (CAs) in issuing SSL/TLS certificates and how self-signed certificates are different from those issued by trusted CAs.
5. **Government Control and Surveillance**:
   * The lesson touched upon how some governments, including those of China and Russia, mandate the use of government operated CAs in some situations, potentially enabling surveillance on HTTPS traffic.
   * You also learned about historical examples of government surveillance programs, such as MUSCULAR which leveraged weaknesses in encryption and data handling practices.
6. **Security Best Practices**:
   * The importance of using strong, modern encryption standards was emphasized, particularly considering vulnerabilities in older protocols like SSL 3.0 and TLS 1.0.
   * You learned that while SSL/TLS encryption is crucial, some communication methods, like end-to-end encryption in messaging apps, can offer even higher levels of security.

## Check for Understanding

Although not graded these questions will prepare you for upcoming assessments.

1. Which of the following annotations is used to define a Spring Boot controller that handles HTTP requests?

* A) @Component
* B) @Service
* C) @Controller
* D) @Repository

**Correct Answer:** C) @Controller

1. In a Spring Boot application, which property file is typically used to define configuration properties such as server port and SSL settings?

* A) application.yml
* B) config.xml
* C) application.properties
* D) server.config

**Correct Answer:** C) application.properties

1. What does the @RequestMapping annotation do in a Spring Boot controller?

* A) It specifies the HTTP method to use.
* B) It maps a URL pattern to a specific controller method.
* C) It defines the return type of a method.
* D) It configures the database connection.

**Correct Answer:** B) It maps a URL pattern to a specific controller method.

1. Which command is used to generate a public-private key pair for SSL in a Java application using the keytool utility?

* A) keytool -importcert
* B) keytool -genkeypair
* C) keytool -list
* D) keytool -exportcert

**Correct Answer:** B) keytool -genkeypair

1. In a Spring Boot application, what does the server.ssl.key-store property specify?

* A) The type of SSL/TLS certificate used by the server.
* B) The path to the keystore file containing the server's SSL certificate and private key.
* C) The password used to access the keystore.
* D) The algorithm used for SSL encryption.

**Correct Answer:** B) The path to the keystore file containing the server's SSL certificate and private key.

1. Which of the following is NOT a valid key store type for storing SSL certificates in Java?

* A) JKS
* B) PKCS12
* C) PFX
* D) XML

**Correct Answer:** D) XML

1. What is the primary role of a certificate authority (CA)?

* A) To encrypt data in transit.
* B) To validate and issue digital certificates that authenticate the identity of organizations and domains.
* C) To generate private keys for clients.
* D) To monitor and log network traffic.

**Correct Answer:** B) To validate and issue digital certificates that authenticate the identity of organizations and domains.

1. Which of the following is a common default port for HTTPS traffic in a Spring Boot application?

* A) 8080
* B) 443
* C) 8443
* D) 3306

**Correct Answer:** C) 8443

1. In the SSL/TLS handshake process, what does the server send to the client after receiving the "ClientHello" message?

* A) The session key.
* B) The server's SSL certificate.
* C) The client's public key.
* D) The encrypted data.

**Correct Answer:** B) The server's SSL certificate.

1. Which of the following is a potential risk of using self-signed certificates in a production environment?

* A) Increased computational overhead.
* B) Vulnerability to SQL injection attacks.
* C) The certificate might not be trusted by clients and web browsers.
* D) The certificate will expire too quickly.

**Correct Answer:** C) The certificate might not be trusted by clients and web browsers.

# Deliverables:

1. Submit a Microsoft Word document with screenshots of the application being run. Show each screen of the output and put a caption under each picture explaining what is being demonstrated.
2. In the same document, in one paragraph, write a summary of the key concepts that were demonstrated in this lesson.
3. In the same Word document, include completed the Protocols Table.

|  |  |  |
| --- | --- | --- |
| Initials | Name | What It Is Used For |
| HTTP | Hypertext Transfer Protocol | Send hypertext pages for world wide web applications. |
| TCP |  |  |
| SNMP |  |  |
| FTP |  |  |
| SMTP |  |  |
| IMAP |  |  |
| POP3 |  |  |
| DNS |  |  |
| SSH |  |  |
| UDP |  |  |
| RDP |  |  |
| VoIP |  |  |
| DHCP |  |  |
| LDAP |  |  |
| TLS |  |  |

1. Convert the Word document to a PDF.
2. A screenshot of a computer

   Description automatically generatedSubmit a ZIP file of the project file. In order to save space, you can delete the target folder of the project. These folders contain the compiled version of the application and are automatically regenerated each time the build or run commands are executed.
3. Attach the PDF document separately from the zip file. Multiple files can be uploaded with an assignment.