DEVELOPING CERTIFICATE-BASED PROJECTS FOR WEB SECURITY CLASSES*

Shamima Rahman *Tuan Anh Nguyen* *T. Andrew Yang*
Univ. of Houston – Clear Lake
2700 Bay Area Blvd., Houston, TX 77058
rahmans3984@uhcl.edu
nguyent2591@uhcl.edu
yang@uhcl.edu
(281) 283-3835B

ABSTRACT

Increasing number of applications are using the Internet to exchange data, varying from online chatting to credit card numbers and other sensitive information. Accompanying the widespread use of inter-networks is the ubiquitous problem of malicious attacks at the applications and the underlying networks. Data transmitted without proper protection are subject to unauthorized access and tampering. To fortify an application against attacks, it is important to integrate proper security measures. In this paper we present web security projects utilizing certificate-based mechanisms to secure web applications. The projects involve imitating attacks and protecting resources from those attacks. The projects involve the use of security technologies such as Secure Socket Layer (*SSL*), Digital certificates, and *HTTPS* (Secure HyperText Transport Protocol) for securing communication channels. By integrating the projects into web development courses, instructors may provide practical exercises that help students to acquire real-life knowledge of how these attacks are performed and how the control measures work.

KEYWORDS: Web Security, SSL, Digital certificate, HTTPS, Sniffing, Network attacks

---

1. INTRODUCTION

Since its birth, the Internet has experienced tremendous growth in connecting millions of computers all over the world, and has made it possible for users to share resources across the Internet. Web-based applications, such as e-commerce, e-banking, virtual organizations, etc., rely on the Internet connectivity for their success. As connectivity and sharing increase, security and privacy issues are becoming increasingly critical for web-based applications. Those applications are subject to attacks at various resources, including the web servers, the communication links, the authentication and authorization mechanisms, the data connectivity between the application and the backend databases, etc. Common examples of the attacks include Denial of Service (DoS), eavesdropping, impersonating, etc. As a result, security has become one of the most challenging issues facing any web-based application.

Our aim is to devise a set of projects for web security classes so that students can learn the issues related to web security by implementing the projects, which demonstrate how certificate-based technologies may be used in securing web-based applications. Certificate-based technologies involve the use of digital certificates in computer protocols, such as SSL/TLS, HTTPS, etc. In the rest of this section, digital certificates and related background concepts are discussed. In Section 2, we present the lab setup used for implementing the projects. Section 3 contains the general framework that we adopted from [11] for ensuring consistent project designs. The projects are presented in section 4.

Digital Certificates: In Public Key Infrastructure (PKI) [5], each entity possesses a pair of public and private keys, where the public key is known to others in the system, and the private key must be securely guarded by its owner. Whatever encrypted with the public key can only be decrypted with the corresponding private key, and vice versa. A digital certificate provides the binding between a public key and its owner. An entity's digital certificate is issued to its owner by a trusted Certifying Authority (CA). The CA generates a digital signature for the certificate by encrypting the entity's public key and other identification information with the CA's own private key. The signature allows a user to verify a given certificate to determine, for example, whether the public key contained in the certificate really belongs to that entity. Normally a trusted CA's certificate is distributed to all the members of a PKI system [4]. Therefore, a CA's identity can be securely verified by all the members of a system. The format of a digital certificate is defined by standards such as X.509 [5].

Secure Socket Layer (*SSL*) / Transport Layer Security (*TLS*): SSL [7] provides authentication, data encryption, and data integrity to TCP/IP traffic in a PKI system. SSL achieves authentication by exchanging digital certificates verified by trusted CAs, and provides confidentiality through session-key encryption and data integrity through message authentication codes (MAC). SSL uses digital certificates to authenticate users and systems. TLS [2] is based on SSL and considered the successor of SSL. Many existing applications have embedded SSL support. HTTPS [9] (HyperText Transport Protocol Secure, aka HTTP over SSL), for example, encrypts and decrypts messages and web pages using SSL/TLS.

2. EXPERIMENTAL LAB SETUP

As the projects involve hands-on experiments emulating attacks and counter measures, carrying out the experiments in an academic lab environment could put the campus network at risk. Therefore, academic institutions are typically reluctant to allow such projects to be carried out in the campus network. We are fortunate to have a specially designed Distributed Computer Security Lab (DCSL) [11] for developing and testing computer security projects. The teaching network is insulated from other parts of DCSL and the campus network via a firewall. The workstations in the teaching network are equipped with swappable disk units, on which students may install whatever OS or tools they need to use. Furthermore, attacks performed in the teaching network are contained in that network. (Note[1])

3. FRAMEWORK FOR DESIGNING THE PROJECTS

We have developed a framework for consistently representing the projects [11]. The framework consists of the following components: (a) Learning Objectives of the project; (b) Tools utilized to implement the project; (c) Requirements that must be met when implementing the project; (d) Problem classification: A project may be a study project and/or a programming project. (e) Methods of implementing the project in the security lab: This explains the necessary network infrastructure and privileges students may need in order to implement the project in the Lab. (f) Level of difficulty: beginner, intermediate, or advanced; (g) Grading criteria and methods: This describes the grading criteria and methods for the instructor/grader to evaluate the project.

4. WEB SECURITY PROJECTS

In this section, we present five certificate-based projects, which were developed in an incremental manner, meaning each one is built upon the previous project(s) with increased complexity. The 1st project deals with setting up a simple web-based application, which is to be used as the base of the other projects. The 2nd project involves adding *passive eavesdropping* to the 1st project. The 3rd and the 4th projects are related to securing data transmissions between the web server and the web browser, using SSL-enabled technologies. The 5th project explores possible vulnerabilities associated with SSL.

Project 1: Developing a simple three-tiered application

a) Learning Objective: In a multi-tiered e-commerce application, the communication between the entities involved must be secured in order to protect the transmitted data. This project involves developing a simple three-tiered application. The later projects will be based on this application.

b) Tools utilized:

[1] Information about the DCSL networks is available at http://www.dcsl-uhcl.net/public/experiments.html.

(i) **Apache Tomcat** [1] is a freely available Servlet/JSP container. This application could be used to host the web application (JSP/html pages). **JDK** [10] is required for the installation of Tomcat.

(ii) **MySQL** DBMS [6] is a freely available open source database for non-commercial use. MySQL could be used to create the necessary database, tables, etc. for the three-tier web application.

c) Requirements:

Students are required to download and configure Apache Tomcat and MySQL first, and then develop a 3-tier web application. The front tier (a web browser) provides the web clients proper user interface to the web application (the middle tier), which processes the clients' requests and, if necessary, forwards the requests to the DBMS at the back end (the back tier).

d) Problem classification: This experiment is classified as a programming assignment.

e) Methods of implementing the project in the security lab: Students can work as a team of two. Each student is assigned a swappable hard disk. Students download the necessary applications from their corresponding web sites, and configure them as mentioned above.

f) Level of difficulty: This experiment is classified as an experiment for beginners.

g) Grading criteria and methods:

Graders can perform simple queries like insert, update, select, etc., using the web interface of the project and check whether all the tiers are functioning properly.

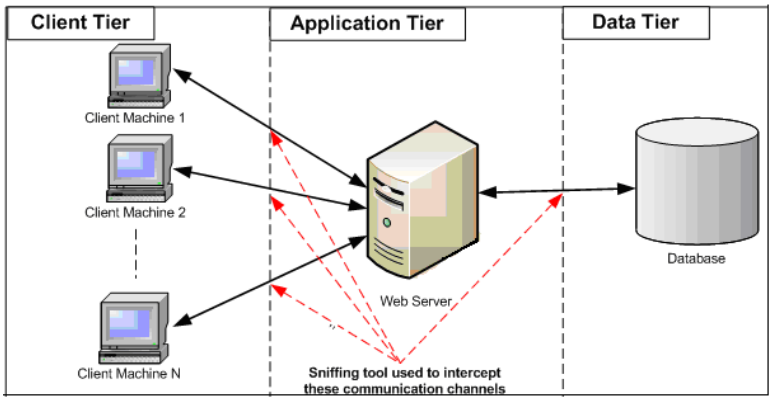

Figure 1: Insecure communication channels in a three-tiered web application

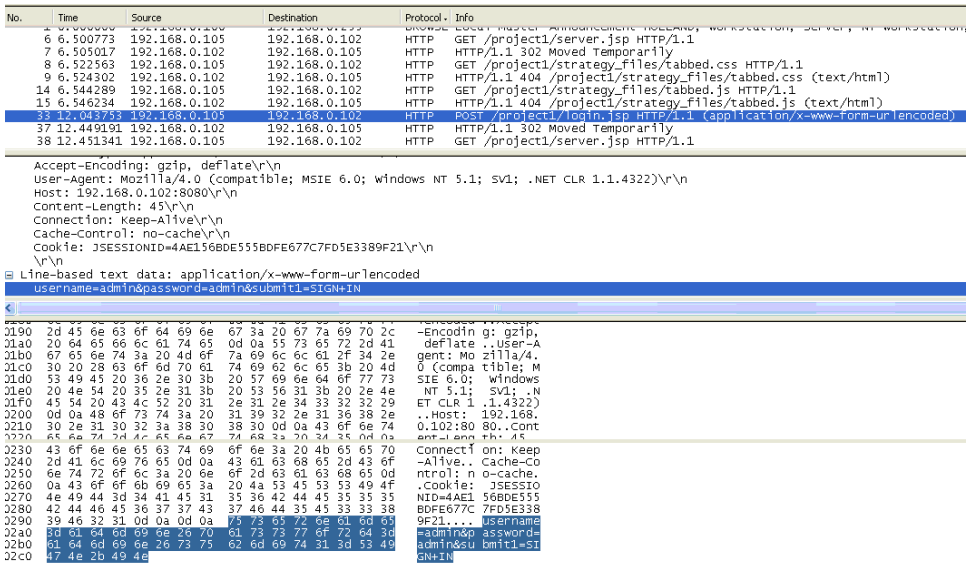

Figure 2: *HTTP* Traffic captured using Ethereal

Project 2: Identifying vulnerability points of the three-tiered application, by using a sniffing tool

a) Learning Objective: In this project, students are to identify the vulnerability points of the three-tiered application developed in project 1. As no security measures were adopted in project 1, the data transmitted between the browser and the web server, and those between the web server and the DBMS, can be easily intercepted. Figure 1 points out the insecure communication channels in a typical web application, which can be intercepted using a sniffing tool. In this project, students will use a packet sniffing tool (e.g., Ethereal [3]) to intercept the data exchanged in between the communicating entities. Figure 2, for example, shows that HTTP protocol is identified by Ethereal, and a set of usernames and passwords are captured from a HTTP traffic.

b) Tools utilized:

(i) **Apache Tomcat** [1]; (ii) **MySQL DBMS** [6]; (iii) **Ethereal** [3] is a freely available packet sniffing tool, which can be used for sniffing packets and monitoring network traffic.

c) Requirements:

Students are required to set up Ethereal to monitor the data traffic. Ethereal can be configured to capture the data passing through a network interface. Students can run Ethereal for capturing data against an interface associated with communication on the client computer, the web server, or the DBMS machine. Plainly exchanged data, such as username, password, etc., can be intercepted.

d) Problem classification: This experiment mostly involves configuring the sniffing tool to capture data traffic, and is classified as a programming assignment.

e) Methods of implementing the project in the security lab: This is an individual project. Students can download and configure Ethereal, and sniff the traffic in

between the client-browser and the web server, and between the web server and the database server.

f) Level of difficulty: The difficulty level for this experiment is classified as for beginners.

g) Grading criteria and methods:

Students submit a report containing details of how they have set up and used Ethereal. They may also submit snapshots of running the tool and intercepting data (e.g., Figure 2).

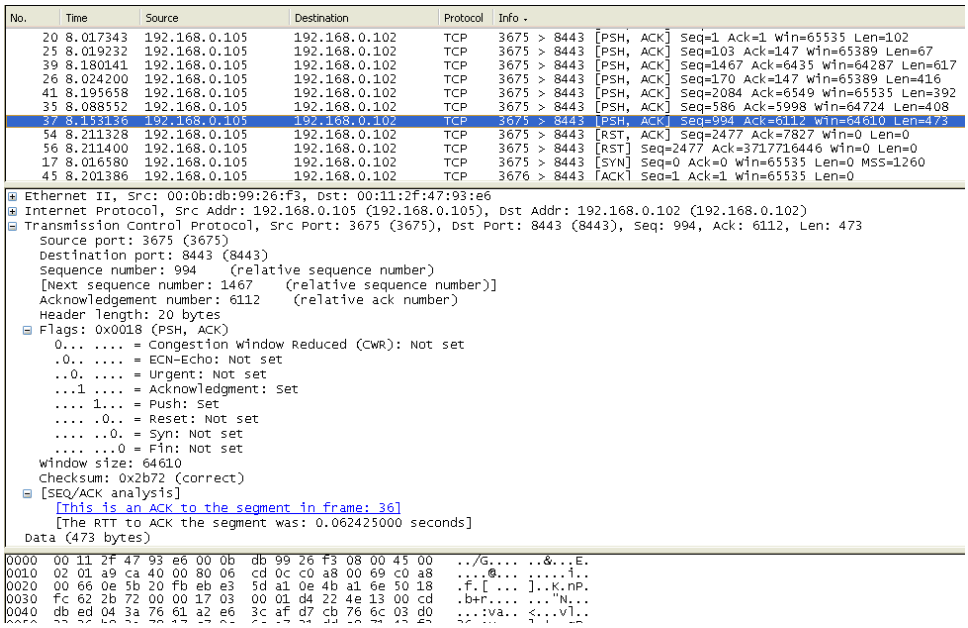

Figure 3: *HTTPS* Traffic captured using Ethereal

Project 3: Configuring SSL for a three-tiered application

a) Learning Objective: In this project, students need to enable SSL between the browser and the web server, and between the web server and the MySQL database server. Figure 3, for example, illustrates that the HTTPS protocol is identified as TCP, and no meaningful data are revealed by the sniffing tool.

b) Tools utilized:

(i) **Apache Tomcat** [1]; (ii) **MySQL DBMS** [6]; (iii) **OpenSSL** [8] is an open source tool with extensive cryptography library implementing SSL and TLS protocol. This tool will be used to create the certificate for MySQL and to support the SSL connections to MySQL. (iv) **JDK** [10] provides a tool, **keytool**, for managing keystores and certificates.

c) Requirements:

(i) To configure SSL between the browser and the web server, a certificate needs to be generated for Tomcat using keytool. Students may use the *keytool* to generate their self-signed certificates.

The configuration file 'server.xml' for Tomcat also has to be modified. A new connector (usually port 8443) has to be added to the server.xml file. Details are available from the Tomcat documentation *ssl-howto.html*.

(ii) To configure SSL between the web server and the DBMS, first generate and install certificates for MySQL using OpenSSL. The Tomcat certificate should be transferred to the DBMS machine, as Tomcat will act as the client in this SSL connection. The MySQL certificate needs to be added to Tomcat's truststore. In addition, the *mycnf.ini* file has to be modified to reflect the creation and location of the certificates. (Note[2])

(iii) After completing all the configurations, students can run the applications and monitor the data traffic using Ethereal to see that the traffic is being encrypted and can not sniffed by Ethereal.

d) Problem classification: This project is classified as a programming and study experiment.

e) Methods of implementing the project in the security lab: Students use the web application developed in project 1 as the base. They can download necessary software and use them as mentioned above.

f) Level of difficulty: This project is classified as an experiment of intermediate difficulty.

g) Grading criteria and methods:

To Test the SSL connection between the browser and the web server, the grader may access the application using the URL starting with https. To test the SSL between the web server and the DBMS, the grader can try to connect to MySQL with simple user accounts and SSL-enabled user accounts. Students may also be asked to capture and submit snapshots of monitoring the encrypted traffic through these communication channels.

Figure 4 illustrates the communication channels being secured and encrypted by implementing SSL.

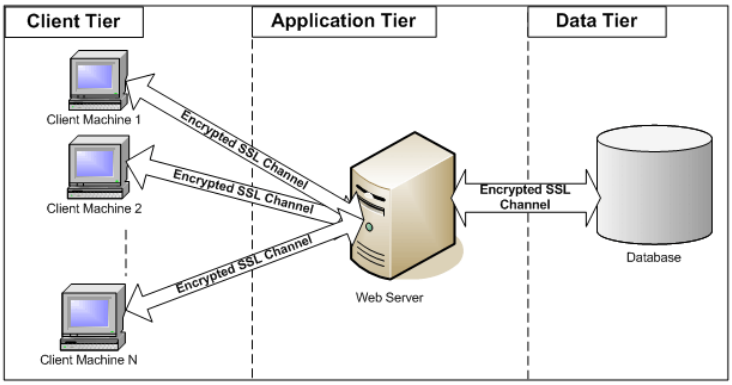

Figure 4: Communication channels being secured with SSL

[2] The tutorial regarding this configuration can be found at http://dev.mysql.com/doc/refman/5.0/en/secure-connections.html and http://dev.mysql.com/doc/refman/5.0/en/cj-using-ssl.html.

Project 4: Securing the communication channels of the three-tiered application by programmatically configuring SSL

a) Learning Objective: In Project 3, we explained how to enable a SSL communication by configuring the related applications (Tomcat, MySQL, etc.). This project involves programmatically implementing SSL connections between them. The main goal of this project is to learn and use the APIs involved for such communication.

b) Tools utilized:

(i) Java has some classes available to implement SSL-enabled connection (such as the *javax.net.ssl* and the *java.security* packages). Examples of such classes are *SSLContext*, *SSLSocketFactory*, *TrustManagerFactory*, *SSLServerSocket*, etc. Students also need *keytool* to generate the certificates for this project. The focus of this project is to use the Java classes to generate applications capable of establishing such connections. (Note[3]) (ii) Students should use the java APIs to develop two socket programs capable of creating a SSL tunnel in between and exchanging data through the tunnel.

c) Problem classification: This project is classified as a programming and study assignment.

d) Methods of implementing the project in the security lab: Students can work as a team of two. Each team is assigned two swappable hard disks for completing the project.

e) Level of difficulty: This project is of intermediate difficulty.

f) Grading criteria and methods:

The grader can use a sniffing or network traffic monitoring tool to check the successful implementation of the SSL-enabled connection. In addition, students submit their reports describing which and how the APIs have been used. They also include snapshots of the programs' execution.

Project 5: Using *ssldump* to decrypt SSL-encrypted communication

a) Learning Objective: This project helps students to figure out vulnerabilities of a SSL-enabled communication. SSL uses public key cryptography to exchange the session key between the web server and the browser. The web server and the browser use this symmetric key to encrypt data before transmitting. Here if a man-in-the-middle is able to acquire the private key of the server's SSL certificate, then he or she can easily acquire the session key from the intercepted communication. In this project, given the server's SSL certificate's private key, students are asked to use the tool *ssldump* [10] to intercept and decrypt the data passing though the communication channel.

b) Tools utilized:

[3] Details about the APIs could be found at http://java.sun.com/j2se/1.4.2/docs/api/index.html.

ssldump is a SSLv3/TLS network protocol analyzer, and decodes the traffic and displays them in a textual format. The installer for *ssldump* is available at http://www.rtfm.com/ssldump/.

c) Problem classification: This project is classified as a programming and study assignment.

d) Methods of implementing the project in the security lab: Students can work alone for this project and use their hard disks to install the necessary tool.

e) Level of difficulty: This is classified as an advanced project.

f) Grading criteria and methods: The grader provides students the private key, and asks them to decrypt a certain communication. The grade depends on how successful the students decrypt the communication.

5. SUMMARY

In the paper, we present five certificate-based projects in the arena of web security. The projects are organized according to a standard template. Some of the projects involve intercepting data transmitted across a web-based application, while the others deal with using certificate-based control measures, such as SSL and HTTPS, to secure data transmissions. The projects presented in the paper will help educators to teach computer security and web development courses. Some of the projects were used in a Web Security course with great success.

ACKNOWLEDGEMENT

The authors are partially supported by the Institute for Space Systems Operations (ISSO), and the National Science Foundation (DUE 0311592).

REFERENCES

1. The Apache Software Foundation. *Apache Tomcat* http://tomcat.apache.org/, 2005.
2. Dierks, T. and C. Allen. The TLS Protocol, Version 1.0 (*RFC 2246*). Jan. 1999.
3. Ethereal, http://www.ethereal.com/distribution/win32/, 2005.
4. Garms, Jess, and Daniel Somerfield. *Professional Java Security*, Wrox Press Ltd, 2001.
5. Housley, R., W. Ford, etc. *Internet X.509 Public Key Infrastructure Certificate and CRL Profile (RFC 2459)*. Jan. 1999.
6. MySQL http://www.mysql.com/, 2005.
7. Netscape. *SSL 3.0 Specification*, http://wp.netscape.com/eng/ssl3/
8. The OpenSSL Project. http://www.openssl.org/, 2005.
9. Rescorla, E. *HTTP over TLS* (*RFC 2818)*. May 2000. ftp://ftp.rfc-editor.org/in-notes/rfc2818.txt

10. Rescorla, Eric (RTFM, Inc). *ssldump*, http://www.rtfm.com/ssldump/, 2005.

11 Sadasivam, Karthik, Banuprasad Samudrala, and T. Andrew Yang. Design of network security projects using Honeypots, *Journal of Computing Sciences in Colleges, 20* (4), 2005.