## **DEVELOPING CERTIFICATE-BASED PROJECTS FOR WEB**

# SECURITY CLASSES\*

Shamima Rahman

Tuan Anh Nguyen 7

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

<sup>\*</sup> Copyright © 2006 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

#### **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 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].

<u>Secure Socket Layer (SSL) / Transport Layer Security (TLS)</u>: 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<sup>1</sup>)

### 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) <u>Tools utilized</u> to implement the project; (c) <u>Requirements</u> that must be met when implementing the project; (d) <u>Problem classification</u>: A project may be a study project and/or a programming project. (e) <u>Methods of implementing the project in the security lab</u>: This explains the necessary network infrastructure and privileges students may need in order to implement the project in the Lab. (f) <u>Level of difficulty</u>: beginner, intermediate, or advanced; (g) <u>Grading criteria and methods</u>: 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 1<sup>st</sup> project deals with setting up a simple web-based application, which is to be used as the base of the other projects. The 2<sup>nd</sup> project involves adding *passive eavesdropping* to the 1<sup>st</sup> project. The 3<sup>rd</sup> and the 4<sup>th</sup> projects are related to securing data transmissions between the web server and the web browser, using SSLenabled technologies. The 5<sup>th</sup> project explores possible vulnerabilities associated with SSL.

### Project 1: Developing a simple three-tiered application

- a) <u>Learning Objective</u>: 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) <u>Tools utilized</u>:

<sup>&</sup>lt;sup>1</sup> 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 noncommercial use. MySQL could be used to create the necessary database, tables, etc. for the three-tier web application.

c) <u>Requirements</u>:

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) <u>Problem classification</u>: This experiment is classified as a programming assignment.
- e) <u>Methods of implementing the project in the security lab:</u> 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) <u>Level of difficulty</u>: 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.

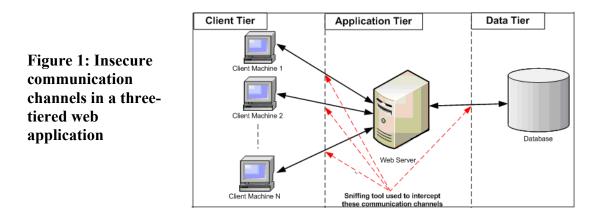


Figure 2: Accept-Encoding: grip, deflate/r/n User-Agent: More 2110/4/0.02 HTTP HTTP HTTP/1.1 Get /project/server.jsp.HTTP/1.1   Figure 2: Accept-Encoding: grip, deflate/r/n User-Agent: More 2101/20.108.0.102 HTTP HTTP HTTP/1.1 Get /project/server.jsp.HTTP/1.1   Figure 2: Market 102.108.0.102 HTTP HTTP HTTP HTTP HTTP/1.1   Market 2: Marke
8 6.522563 192.168.0.105 192.168.0.102 HTTP GET /project1/strategy_files/tabbed.css HTTP/1.1   9 6.524302 192.168.0.105 192.168.0.105 HTTP HTTP /I.1 404 /project1/strategy_files/tabbed.css (text/html)   14 6.544289 192.168.0.105 192.168.0.105 HTTP HTTP /I.1 404 /project1/strategy_files/tabbed.is   13 12.043753 192.168.0.105 192.168.0.105 HTTP HTTP /I.1 404 /project1/strategy_files/tabbed.js   13 12.043753 192.168.0.105 192.168.0.102 HTTP HTTP/1.1 404 /project1/strategy_files/tabbed.js HTTP/1.1   8 12.043753 192.168.0.105 192.168.0.102 HTTP HTTP/1.1 404 /project1/strategy_files/tabbed.js HTTP/1.1   8 12.0431341 192.168.0.105 192.168.0.102 HTTP HTTP/1.1 302 Moved Temporarily   3 12.043753 192.168.0.105 192.168.0.102 HTTP HTTP/1.1 302 Moved Temporarily   3 12.043191 192.168.0.105 192.168.0.102 HTTP HTTP/1.1 302 Moved Temporarily   3 12.043191 192.168.0.105 192.168.0.102 HTTP GET /project1/server.jsp HTTP/1.1   Accept_Encoding: grip, deflate/r\n user_ampe_admines/seconde.org user_ampe_admines/seconde.org   Content_ength 45/r04 files/r04/r01 content_ength   Content_ength 45/r04 files/r04/r07 files/r04/r01
9 6.524302 192.168.0.102 192.168.0.102 HTTP, HTTP/1.1/404 /project/strategy_files/tabbed; css (text/html)   14 6.54428 192.168.0.102 192.168.0.102 HTTP, GET /project/strategy_files/tabbed; strategy_files/tabbed; strategy_
14 6.544289 192.168.0.105 192.168.0.102 HTTP GET /projecti/strategy_files/fabbed.js HTTP/1.1   13 12.043753 192.168.0.105 192.168.0.102 HTTP HTTP/1.1 404 pojecti/strategy_files/tabbed.js HTTP/1.1   13 12.043753 192.168.0.105 192.168.0.102 HTTP HTTP/1.1 404 pojecti/strategy_files/tabbed.js HTTP/1.1   13 12.043753 192.168.0.105 192.168.0.102 HTTP HTTP/1.1 Accept.etcoding: grip.deflate/r\n   38 12.451341 192.168.0.105 192.168.0.102 HTTP GET /projecti/server.jsp HTTP/1.1   Accept.etcoding: grip.deflate/r\n user_adget: mojecti/size/server.jsp HTTP/1.1 Accept.etcoding: grip.deflate/r\n   User_adget: mojecti/size/server.jsp HTTP/1.1 Accept.etcoding: grip.deflate/r\n User_adget: mojecti/server.jsp HTTP/1.1   Contert.length 45/r\n Compatible; MSIE 6.0; Windows NT 5.1; SVI; .NET CLR 1.1.4322)/r\n Host: 192.168.0.102: 192.168.0.102 HTTP   Contert.length 45/r\n Compatible; MSIE 6.0; Windows NT 5.1; SVI; .NET CLR 1.1.4322)/r\n Host: 192.168.0.102 HTTP   Contert.length 195.2168.0.102 196.2168.0.102 196.2168.0.102 196.2168.0.102
33 12:043735:192:168:0:102 HTTP POST / project1/Jobin.5 (pr.HTP/L1 (application/x-www-form=unlencoded)   37:12:449101:92:168:0:102 HTTP POST / project1/Jobin.5 (pr.HTP/L1.1 (application/x-www-form=unlencoded)   38:12:451341:192:168:0:105 192:168:0:102 HTTP GET / project1/Jobin.5 (pr.HTP/L1.1 (application/x-www-form=unlencoded)   38:12:451341:192:168:0:105 192:168:0:102 HTTP GET / project1/server.jsp HTTP/L1.1   Accept=Encoding: gzip, deflate\r\n uservalue Uservalue Uservalue Uservalue   Traffic connection: keep-Altve\r\n connection: keep-Altve\r\n connection: keep-Altve\r\n   captured using Line-based text data: application/x-www-form-unlencoded uservalue HTTP 466:07707   195:23:23:67:24:26:63:67:64:26:25:58:07:66:772:7FD5E3389F21\r\n r\n http://n   v/n Uservalue 195:23:23:67:72:67:72:55:3389F21\r\n   v/n Line-based text data: application/x-www-form-unlencoded uservalue   using 195:23:23:67:72:67:72:57:73:57:72:67:72:55:3389F21\r\n   195:23:24:66:63:67:64:77:73:73:72:72:67:73:73:72:72:67:73:73:72:72:67:73:73:72:72:67:73:73:72:72:74:74:74:74:74:74:74:74:74:74:74:74:74:
37 12.449191 192.168.0.102 192.168.0.105 HTTP HTTP/II.102 Moved Temporarlly   38 12.451841 192.168.0.105 192.168.0.105 HTTP GET /project1/server.jsp HTTP/I.1   Accept-Encoding: gzip, deflate\r\n user-Agent: MozIlla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; Sv1; .NET CLR 1.1.4322)\r\n   HTTP HTTP Content-Length: 45\r\n   Content-Length: 45\r\n Content-Length: 45\r\n   Content-Length: 45\r\n Content-Length: 45\r\n   Cache-Control: no-cache\r\n Content-Length: 45\r\n   Cashe-Control: no-cache\r\n Content-Length: 45\r\n   Cashe-Control: no-cache\r\n Content-Length: 45\r\n   Cashe-Control: no-cache\r\n Content-Length: 45\r\n   V10 Cathe-Control: no-cache\r\n   V11 Cathe-Control: no-cache\r\n   Content-Length: sep-Alive\r\n Content-Length: sep-Alive\r\n   Cathe-Control: no-cache\r\n Content-Length: sep-Alive\r\n   Cashe-Control: no-cache\r\n Content-Length: sep-Alive\r\n   Cashe-Control: no-cache\r\n Content-Length: sep-Alive\r\n   Cashe-Control: no-cache\r\n Content-Length: sep-Alive\r\n   User/Aander Line-Off add sep-Alive\r\n   Line-Osted text data: application/x-www-form-urlenco
38 12.41341 192.168.0.105 192.168.0.102 HTP GET /project1/server.jsp HTTP/1.1   Accept_Encoding: gzip, deflate\r\n user_Agent: Moz11la/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SVI; .NET CLR 1.1.4322)\r\n   HTTP Connection: keep-Allve\r\n compatible; MSIE 6.0; Windows NT 5.1; SVI; .NET CLR 1.1.4322)\r\n   Host: 192.168.0.102 iso80/r\n compatible; MSIE 6.0; Windows NT 5.1; SVI; .NET CLR 1.1.4322)\r\n   Host: 192.168.0.102 iso80/r\n connection: keep-Allve\r\n   Connection: keep-Allve\r\n connection: keep-Allve\r\n   Cache-Control: no-cache/r\n context-Length   Context-Length context-Length   Username=admin& damin& d
Initial State user-Agent: M021113/4:0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322)\r\n   HTTP Host: 192.168.0.102:8080/r\n   Content-Length: 45\r\n Content-Length: 45\r\n   Cather-Control: 10-Cather-Control: 10-Cather-Cont
HTTP Host: 192.168.0.102:8080/r\n   Content-Length: 45\r\n Connection: keep-Allye\r\n   Captured Connection: keep-Allye\r\n   using Line-based text data: application/x=www-form-unlencoded   Using Username-admin& 25 x 66 x 67 x 20 x 27 x 67 x 20 x 2
HTTP content-length: 45/vh   Connection: keep-Alive/vh content-length: 45/vh   Traffic content-length: 45/vh   Cache-Control: no-cache/vh cache-control: no-cache/vh   contextion: keep-Alive/vh cache-control: no-cache/vh   using 1100 20 44 56 66 66 66 77 36 67 72 69 70 2cEncodfin gi: gzlp.   ziao 20 64 65 66 66 c6 17 46 69 6e 67 73 65 72 22 d4 1 deflate: user-A   ziao 20 64 65 66 c6 17 46 96 6c 65 33 20 40 df 77 49 69 6c 66 53 32 20 40 cf cm cache-control: no-cache/vh   ziao 20 20 86 66 df 66 df 77 73 35 20 20 31 32 20 23 35 30 20 57 69 66 64 df 77 73 51 60 cm cache-control: no-cache/vh   ziao 31 49 45 20 36 2e 30 3b 20 55 76 96 66 46 77 73 51 60 cm sci. y windows   ziao 20 44 55 20 30 2 22 33 2b 20 35 2b 20 31 2b 20
Traffic Connection: Rep-Aityet/M   Captured Connection: captured   using Line-based text data: application/x-www-form-urlencoded   using Cot 45 66 63 67 64 69 66 67 23 69 70 2c - Encodin s:: g215, 140 0 44 56 66 61 74 69 66 67 73 66 70 2c - Encodin s:: g215, 140 0 44 56 66 66 174 69 66 67 24 69 66 67 23 69 70 2c - Encodin s:: g215, 140 0 57 65 66 74 32 0 46 67 73 66 70 2c - Encodin s:: g215, 140 0 50 46 56 66 66 174 69 66 67 66 10 46 57 26 69 66 64 77 73 65 72 64 11 0 46 flate: user-A   Line-based 100 20 46 56 66 66 74 69 66 67 23 69 70 2c - Encodin s:: g215, 140 0 50 46 56 76 66 17 73 65 72 64 71 0 46 flate: user-A   Line 36 66 66 67 46 77 76 96 70 66 66 77 73 51 73 65 70 2c - Encodin s:: g215, 140 0 50 24 86 30 66 27 0 20 28 66 31 20 0 45 72 0 51 20 40 0 (compa tible; m) 140 0 31 49 45 20 35 26 31 3b 20 25 76 96 66 46 77 73 51 6.0; windows   Judo 51 34 94 52 0 35 2e 31 3b 20 53 76 91 3b 20 25 76 51 3b 20 26 40 minute: solution   Judo 51 34 94 52 0 35 2e 31 3b 20 53 76 96 66 47 77 73 51 6.0; windows   Judo 20 48 64 20 35 2e 31 3b 20 53 76 96 66 47 77 73 51 6.0; windows   Judo 20 48 54 20 35 2e 31 3b 20 53 76 96 66 47 77 73 51 6.0; windows   Judo 20 48 54 20 35 2e 31 3b 20 53 76 96 66 47 77 73 51 5.0; windows
Traffic   Cache-Control: ip-cache\/\n     Coptured   Coche: JSESSIONID=44B156B0E555BDFE677C7FD5E3389F21\/\n     captured   Line-based text data: application/x-www-form-urlencoded     using   190 2d 45 6e 63 6f 64 69 6e 67 3a 20 67 7a 69 70 2c   -Encodin g: gz1p, Ja0 20 64 65 66 c6 17 4 69 0c c6 13 73 65 72 2d 41     Ethereal   Jb0 67 65 6e 74 3a 20 4d 6f 7a 69 6c c6 c6 12 74 30 9c c6 c5 3b 20 4d 0 (compa tible; m Ja0 20 2d 45 6e 73 66 6d 7 74 69 6c c6 c6 12 74 2d 9c c6 c5 3b 20 4d 0 (compa tible; m Ja0 51 2d 45 76 56 76 73 2d 9c 76 9c c6 c6 12 73 12 2d gent: Mo zilla/4.     Using   Jb0 67 65 6e 74 3a 20 4d 6f 7a 69 6c c6 c6 12 74 10 74 96 6c c6 c6 5b 3b 20 4d 0 (compa tible; m Ja0 51 34 94 52 00 35 2e 31 3b 20 55 76 9c c6 46 77 73 5IE 6.0; windows Jb0 02 02 86 35 2d 31 2b 20 55 76 9b c6 4d 67 77 73 5IE 6.0; windows Jb0 02 04 45 54 00 35 2e 31 3b 20 55 76 9b c6 4d 67 77 75 5IE 6.0; windows
captured using   Line-based text data: application/x-www-form-urlencoded username=admin&password=admin&submit1=SIGN=IN     Using   High 2 dd 5 de 63 df 64 69 de 67 3a 20 67 7a 69 70 2c   -Encodin g: gz1p, deflate. User-A     Habe 2 dd 5 de 63 df 64 69 de 67 3a 20 67 7a 69 70 2c   -Encodin g: gz1p, deflate. User-A     Habe 3 dd 5 de 63 df 64 69 de 67 3a 20 67 7a 69 70 2c   -Encodin g: gz1p, deflate. User-A     Habe 3 dd 5 de 63 df 64 7a 69 6c 6c 61 27 46 9 c6 6c 61 27 46 97 a 69 c6 6c 61 27 a 69 c6 c6 61 27 a 69 c6 61 27 a 69 c6 c6
captured using Line-based text data: application/x-www-form-un-lencoded username-admin&password
captured   username=admin&password=admin&submit1=SIGN+IN     using   M90   2d 45 & 66 & 36 f & 64 & 69 & 66 & 73 & 20 & 67 & 7a & 69 & 70 & 2c & -Encoding g: g21p, Ma0 & 20 & 45 & 66 & 61 & 74 & 69 & 6c & 6c & 12 & 45 & 00 & 40 & 51 & 73 & 65 & 72 & 2d & 41 & deflate User-A     Ethereal   M100 & 2d 45 & 6e & 61 & 74 & 69 & 6c & 6c & 12 & 45 & 00 & 40 & 51 & 73 & 65 & 72 & 2d & 41 & deflate User-A     M100 & 70 & 56 & 67 & 43 & 20 & 4d & 6f & 7a & 69 & 6c & 6c & 12 & 74 & 59 & 6c & 6c & 51 & 34 & 2e & gent: M0 & 2i11a/4.     M100 & 53 & 49 & 45 & 20 & 32 & 63 & 38 & 20 & 57 & 69 & 66 & 4d & 67 & 77 & 73 & 51 & 6.0; windows     M100 & 53 & 49 & 45 & 20 & 32 & 63 & 38 & 20 & 57 & 69 & 66 & 4d & 67 & 77 & 73 & 51 & 6.0; windows     M100 & 70 & 45 & 42 & 30 & 20 & 33 & 20 & 53 & 76 & 95 & 66 & 4d & 67 & 77 & 73 & 51 & 6.0; windows
using   150   2d 45   6e   63   64   69   67   7a   69   70   2c   -Encodin g: gzip,     1100   2d 45   6e   63   66   64   69   67   7a   69   70   2c   -Encodin g: gzip,     1100   20   64   65   66   67   7a   69   70   2c   -Encodin g: gzip,     1100   67   65   66   67   7a   69   cc   cc   1deflate, user-A     1100   20   28   63   66   67   65   52   2d 41   deflate, user-A     1100   20   28   66   66   53   20   47   69   6c   66   53   20   47   49   6c   66   65   35   20   47   49   6c   66   53   20   47   49   6c   6c   65   35   20   47   77
Ethereal   Diao   20   64   65   66   61   74   65   04   03   55   73   65   72   24   1   defTare    user-A     Dibo   67   65   66   74   63   20   44   defTare    user-A     Dibo   67   65   66   67   74   69   66   66   61   27   42   cent: Mo   2111a/4.     Dico   30   20   28   63   67   64   62   66   64   0   Cempat   Theilare    User-A   0   0   Cemp
<b>Ethereal</b> $\begin{array}{c} 11a 0 & 20 & 64 & 65 & 66 & 6c & 61 & 74 & 65 & 0d & 0a & 55 & 73 & 65 & 72 & 2d & 41 & deflate \overline{1}, u\overline{ser} \rightarrow A \\ 11b 0 & 67 & 65 & 6e & 74 & 3a & 20 & 4d & 6f & 7a & 69 & 6c & 6c & 61 & 2f & 34 & 2e \\ 11b 0 & 30 & 20 & 28 & 63 & 6f & 6d & 70 & 61 & 74 & 69 & 62 & 6c & 65 & 3b & 20 & 4d & 0 & (compa tible; M \\ 11d 0 & 53 & 49 & 45 & 20 & 36 & 2e & 33 & 3b & 20 & 57 & 69 & 6e & 64 & 6f & 77 & 73 & st E & 0.9 & windows \\ 11d 0 & 20 & 4e & 54 & 20 & 35 & 2e & 33 & 3b & 20 & 53 & 76 & 93 & 6e & 64 & 6f & 77 & 73 & st E & 0.9 & windows \\ 11d 0 & 20 & 4e & 54 & 20 & 35 & 2e & 33 & 3b & 20 & 53 & 66 & 13 & b & 20 & 2e & 4e & windows \\ 11d 0 & 20 & 4e & 54 & 20 & 35 & 2e & 33 & 3b & 20 & 53 & 66 & 13 & b & 20 & 2e & 4e & windows \\ 11d 0 & 20 & 4e & 54 & 20 & 35 & 2e & 33 & 3b & 20 & 53 & 66 & 13 & b & 20 & 2e & 4e & windows \\ 11d 0 & 20 & 4e & 54 & 20 & 35 & 2e & 33 & 3b & 20 & 53 & 66 & 13 & b & 20 & 2e & 4e & windows \\ 11d 0 & 20 & 4e & 54 & 20 & 35 & 2e & 31 & 3b & 20 & 53 & 66 & 13 & b & 20 & 2e & 4e & windows \\ 11d 0 & 20 & 4e & 54 & 20 & 35 & 2e & 31 & 3b & 20 & 53 & 66 & 13 & b & 20 & 2e & 4e & windows & wi$
11.10 53 49 45 20 36 2∈ 30 35 20 57 69 6∈ 64 67 77 73 51€ 6.0; windows 11.60 20 4∈ 54 20 35 2∈ 31 35 20 53 56 13 3b 20 2∈ 4€ NT 5.1; Sv1; N
01.d0 53 49 45 20 36 2e 30 3b 20 57 69 6e 64 67 77 73 51€ 6.0; windows 01.e0 20 4e 54 20 35 2e 31 3b 20 53 56 31 3b 20 2e 4e n⊺ 5.1; Sv1; N
0200 0d 0a 48 6f 73 74 3a 20 31 39 32 2e 31 36 38 2eHost: 192.168.
)210 30 Ze 31 30 32 3a 38 30 38 30 0d 0a 43 6f 6e 74 0.102:80 80Cont
2220 65 6a 74 2d 42 65 6a 67 74 68 $3a$ 20 34 35 0d 0a entiment the 45
2230 43 6f 6e 6e 65 63 74 69 6f 6e 3a 20 4b 65 65 70 Connectíon: Keep 2240 24 1a 6c 69 76 65 0d 0a 43 61 63 68 65 2d 43 6f - Alive Cache-co
0250 6e 74 72 6f 6c 3a 20 6e 6f 2d 63 61 63 68 65 0d  ntrol: n o-cache.
2260 0a 43 6f 6f 6b 69 65 3a 20 4a 53 45 53 53 49 4f .cookie: jsessio
2270 4e 49 44 3d 34 41 45 31 35 36 42 44 45 35 35 35 5 ND⊙+4AEL 56BDE555 2280 42 44 46 45 36 37 37 43 37 46 44 35 45 33 33 38 BDFe677C 7FD5E338
2290 39 46 32 31 0d 0a 0d 0a 75 73 65 72 6e 61 6d 65 9F21 Username
22a0 3d 61 64 6d 69 6e 26 70 61 73 73 77 6f 72 64 3d
22b0 61 64 64 69 6e 26 73 75 62 6d 69 74 31 3d 53 49 admin&sù bmitl=SI 32C0 47 4e 2b 49 4e

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

- a) <u>Learning Objective</u>: 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) <u>Tools utilized</u>:

(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) <u>Requirements:</u>

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) <u>Problem classification:</u> This experiment mostly involves configuring the sniffing tool to capture data traffic, and is classified as a programming assignment.
- e) <u>Methods of implementing the project in the security lab</u>: 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) <u>Level of difficulty:</u> 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).

	No. Time	Source	Destination	Protocol	ol Info -
	39 8.180141 26 8.024200 41 8.195658 35 8.088552 37 8.153136	192.168.0.105 192.168.0.105 192.168.0.105 192.168.0.105 192.168.0.105 192.168.0.105	192.168.0.102 192.168.0.102 192.168.0.102 192.168.0.102 192.168.0.102 192.168.0.102 192.168.0.102	TCP TCP TCP TCP TCP TCP	3675 > 8443 [PsH, ACK] seq=1 ACK=1 win=65335 Len=102 3675 > 8443 [PsH, ACK] seq=103 Ack=147 win=65389 Len=67 3675 > 8443 [PSH, ACK] seq=1467 Ack=6435 win=64287 Len=617 3675 > 8443 [PSH, ACK] seq=2084 Ack=6349 win=65335 Len=322 3675 > 8443 [PSH, ACK] seq=2084 Ack=6349 win=64734 Len=408 3675 > 8443 [PSH, ACK] seq=2886 Ack=5998 win=64724 Len=408 3675 > 8443 [PSH, ACK] seq=386 Ack=5998 win=64724 Len=408
	56 8.211400 17 8.016580	192.168.0.105 192.168.0.105 192.168.0.105 192.168.0.105 192.168.0.105	192.168.0.102 192.168.0.102 192.168.0.102 192.168.0.102 192.168.0.102	TCP TCP TCP TCP	3675 > 8443 [RST, ACK] seq=2477 Ack=7827 win=0 Len=0 3675 > 8443 [RST] seq=2477 Ack=3717716446 win=0 Len=0 3675 > 8443 [SvN] seq=0 Ack=0 win=65535 Len=0 MSS=1260 3676 > 8443 [ACK] seq=1 Ack=1 win=65535 Len=0
Figure 3: <i>HTTPS</i> Traffic captured using Ethereal					
	[The RTT to ACK the segment was: 0.062425000 seconds] Data (473 bytes)				
	0010 02 01 a9 c 0020 00 66 0e 5 0030 fc 62 2b 7	a 40 00 80 06 cd 0 b 20 fb eb e3 5d a 2 00 00 17 03 00 0	9 26 f3 08 00 45 00 c c0 a8 00 69 c0 a8 1 0e 4b a1 6e 50 18 1 d4 22 4e 13 00 cd f d7 cb 76 6c 03 d0	.f.[ .b+r	/G

### Project 3: Configuring SSL for a three-tiered application

- a) <u>Learning Objective:</u> 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) <u>Tools utilized</u>:

(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.

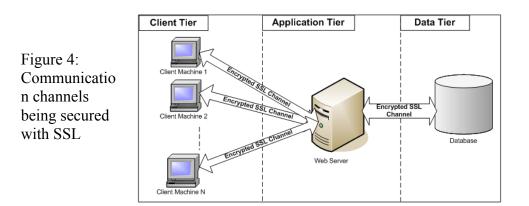
- c) <u>Requirements:</u>
  - (i) To configure SSL between the browser and the web server, a certificate needs to be generated for Tomcat using keytool. Students may use the *keytool* to generate their self-signed certificates.

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<sup>2</sup>)
- (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) <u>Problem classification</u>: This project is classified as a programming and study experiment.
- e) <u>Methods of implementing the project in the security lab</u>: Students use the web application developed in project 1 as the base. They can download necessary software and use them as mentioned above.
- f) <u>Level of difficulty</u>: This project is classified as an experiment of intermediate difficulty.
- g) <u>Grading criteria and methods</u>:

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.



<sup>&</sup>lt;sup>2</sup> 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) <u>Learning Objective</u>: 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) <u>Tools utilized:</u>

(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<sup>3</sup>) (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) <u>Problem classification</u>: This project is classified as a programming and study assignment.
- d) <u>Methods of implementing the project in the security lab</u>: Students can work as a team of two. Each team is assigned two swappable hard disks for completing the project.
- e) <u>Level of difficulty:</u> This project is of intermediate difficulty.
- f) <u>Grading criteria and methods:</u>

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) <u>Learning Objective</u>: 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 manin-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) <u>Tools utilized:</u>

<sup>&</sup>lt;sup>3</sup> 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) <u>Problem classification</u>: This project is classified as a programming and study assignment.
- d) <u>Methods of implementing the project in the security lab</u>: Students can work alone for this project and use their hard disks to install the necessary tool.
- e) <u>Level of difficulty</u>: This is classified as an advanced project.
- f) <u>Grading criteria and methods:</u> 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/innotes/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.