SECURE ONLINE AUCTION WEBSITE

Subject: Network and Computer Security
Campus: Alameda
Group: A28

73216 Nuno Guilherme de Sousa Lobato da Graça

73422 Kévin David Esteves Santos

DISENROLLED

75657 Paulo Jorge Louseiro Gouveia

Abstract

Auction sites are a useful tool to promote the exchange of goods directly exploiting the demand and request needs of dynamic markets. The goal of this work is to create an online auction store, providing sellers and buyers a reliable website with all the mechanisms for the secure transactions of products and goods.
The site is to be secure against threats like XSS, SQL injection, impersonation and other attacks.

Problem

From the topics available our group chose to tackle the development of a secure auction website. Such websites often deal with real money and goods, and as such, are subject to various attacks aiming to steal and/or destroy the money, the goods or the information being shared. Our objective is to defend our website against these attacks by protecting the system infrastructures, as well as protecting the users against possible vulnerabilities and exploitations.

Because it is impossible to protect everything in a system, a model with the assumptions made about its environment are required. Ours are the following:

Social engineering	the user is aware of problems such as phishing and takes the necessary precautions to prevent information theft
Updated browser security	the user is using the latest iteration of his/her browser of choice
Secure database	the database is in a secure location and is itself secure
Inside jobs	people to whom was granted access to the server or database will not use their power to conduct attacks on the system
Network problems	any problems with the communication between clients and the server are not ours to deal with, such problem are the responsibility of respective network links and nodes

Requirements

Given the identified vulnerabilities, the system will have to be protected against several exploits:

Code injection	A malicious user may, use text input fields to inject malicious code into the server. To prevent such exploits input sanitization is necessary.
Packet capture	All data travelling between client and server may be captured. As such, all sensitive information must be encrypted to ensure that only the client and/or server can read that information.
Elevation of privileges	A simple user cannot gain administrative privileges unless specifically given by another administrator.
User impersonation	A malicious user can pretend to be another user. To prevent this a client authentication system must be implemented to assure that a user account is accessible only to the user.
Server-database connection	As with the client-server connection the traffic may be listened to. This channel shall be encrypted as well. The database credentials are stored outside of the server's web root directory.
DoS attacks	High number of requests to the server causing it to become extremely slow in the processing of requests or even crash. Not preventable but mitigated by the use of specific firewall rules.

Proposed Solution

The first iteration of our project will consist of a single machine running both web and database servers with minimal security (simple input sanitization and hashing of stored passwords). With incremental upgrades, the goal is for our final implementation to have separate machines for web and database servers, each with protection against the most common attacks.

The SQL injection and XSS vulnerabilities are easily mitigated if not preventable by sanitization of the input and output of our application. For simplicity, the certificates used will be self-signed X.509 certificates to avoid having to implement or pay an existing Certification Authority. This will enable the use of *SSL* for creating an encrypted channel between client and server. Further security is to be provided by the configuration of *iptables* to prevent attacks such as *SYN flooding*, *LAND*, etc.

There will be no storing of sensitive billing information. After winning a bid, the user will be asked to provide the billing information for immediate process of the purchase.

For the ease of testing and because this is not a real application, all servers will be running in virtual machines. This way, attacks can safely be performed without the risk of ruining a real machine.

The project will be divided into three phases: basic, intermediate and advanced. Each of this phases will respectively increment functionality and the level of security guaranteed upon the previous. The specifications of each development stage are as follows:

Basic *November 17th 2016*	– Basic web interface – Webserver running *PHP* with input sanitization implemented – *MySQL* configured with hashing of passwords
Intermediate *November 24th 2016*	– Different types of accounts – Complete website page tree – Webserver running *SSL* and certificates
Advanced *December 9th 2016*	– Fully functional website with administration functionality – Separate machines for webserver and database – Configuration of *iptables*

Table 1 – *Completion date and specifications of each development stage*

Tools

On the client's side, there will be a web application running industry standard components. The interface will be using *html*, *CSS* and *JavaScript* and the communication with the server will be using *OpenSSL* which allows for self-signed certificates (with *asymmetric keys*) and will work with *https* to assure a confidential channel between browser and server.

As for the servers, they will be running CentOS, Apache HTTP Server and PHP which is a very widely used server configuration, ensuring good documentation and help online. to make sure we have the most recent security updates we will use the latest stable version of each of the software mentioned. The servers' firewalls will consist of the linux kernel firewall tables configured through iptables. For the database we will be using *MySQL* because of the knowledge we already have from previous classes and it still being a very widely accepted industry standard which assures proper testing and help online.

For the virtualization of the server machines we will use Oracle VM VirtualBox which we already know it should work because it's what was used in the laboratory classes.

Work Plan

week	73216	75657
October 27 – November 2	- basic web interface	- basic *MySQL* tables - Apache, *PHP* webserver
November 3 – November 9	- input sanitization - login implementation	- password hash
November 10 – November 16 *<basic>*	- product submission - finalize XSS prevention - *MySQL* tables	- bidding - *MySQL* tables
November 17 – November 23 *<intermediate>*	- *SSL* connection - different account types	- certificates for server - *https* for login - different account types
November 24 – November 30	- product search	- separate webserver and database server
December 1 – December 7 *<advanced>*	- interface touchups - *iptables* configuration	- interface touchups
December 8 – December 10 *<final report>*	- write final report	- write final report

Table 2 – *Functionality to be implemented per week. Weeks are counted from Thursdays to Wednesdays to have the laboratory class in the beginning of the work cycle.*