Project Name

Project:	Project Name
Document Name:	Project Threat model
Date:	4/28/2011
Customer:	Customer
Period of Performance:	4/25/2011 - 5/20/2011
Customer Contact:	Name <e-mail>
Threat Model Author:	Name <e-mail>
Project Manager:	Name <e-mail>

BOSTON | SEATTLE

187 Ballardvale St. Suite A195 ●Wilmington, MA 01887● Ph: +1.978.694.1008

getsecure@securityinnovation.com● www.securityinnovation.com

Contact Information

Security Innovation

Business Contact

<<
Name
Title
e-mail address
Mobile: 123-123-1234
Fax: 123-123-1234
>>

Technical Contact

<<
Name
Title
e-mail address
Mobile: 123-123-1234
Fax: 123-123-1234
>>

<<Customer Company>>

<<
Name
Title
e-mail address
Mobile: 123-123-1234
Fax: 123-123-1234
>>

Table of Contents

Contact Information **1**
Security Innovation 1
Business Contact 1
Technical Contact 1
<<Customer Company>> 1
Table of Contents **2**
Executive Summary **3**
Introduction **4**
Threat Model Creation Methodology **5**
System Decomposition **6**
Assets 6
Roles 6
Components 7
Component Diagram 8
Activity Matrix **9**
Threat Tree Information **11**
Threat Tree Details **12**

Executive Summary

Security Innovation will perform a security audit of this Application as part of a full SDLC Gap Analysis and Review.

Security Innovation reviewed all available documentation, performed exploratory testing and met with Tech Leads to build a complete understanding of the system. During each meeting an understanding each of the following areas was discussed:

- Features and use cases of the component
- Users of the component
- What data is being consumed and produced by this component
- What data must be protected or is considered sensitive
- If there is an administrative interface, how that is used and how it is protected
- Any existing security controls, reviews or considerations
- Protocols, Libraries, Frameworks or other external components used
- Whether the component was written by this company, by a 3rd party company, or was an off the shelf solution
- Biggest security concerns as viewed by the Tech Lead

Threat modeling is the first step for successful software security audits. The threat model shows the results of a close security analysis of the design. Security Innovation will use this threat model to gather attack vectors and generate test cases for comprehensive security testing of the application.

The following list summarizes the important points that impact the attack surface of The Application:

- This application uses SSL for all communications
- All interfaces except authentication itself are protected by single factor authentication
- Only clients and vendors are authorized to be users
- The specific data within application is a high value target

The top threats for malicious attack are:

- Bypassing initial authentication
- Elevation of privilege via injection
- Elevation of privilege via forceful browsing

Introduction

Threat modeling is a necessary step to create actionable security test plans and to properly understand the security footprint of the system. Security Innovation has created a Threat Model to analyze and gather all possible avenues of attack. These attack vectors will be used to generate test cases in the security test plan along with the conditions and steps required to execute each of them.

Threat Model Creation Methodology

The Security Innovation threat modeling methodology comes from years of experience threat modeling to find the most impactful and actionable threats in a system. It is designed to quickly assess each role, asset, component and activity to understand the most common and highest priority threats to the system.

Threat modeling consists of the following steps:

1. Understand architecture and security requirements
2. Identify assets, roles, and system components
3. Build an activity matrix and define the related rules
4. Identify threats that put assets at risk
5. Assign related components to each threat
6. Identify conditions under which a threat may be realized

Once the threat model is complete we use it to:

1. Guide Design Reviews which can highlight early application flaws that can be costly to fix later
2. Highlight high impact areas for a Code Review to help create Code Review Objectives documents
3. Create a Test Plan that can be used to guide black box penetration testing
4. Choose appropriate mitigations and responses to any realized threats

System Decomposition

In this section Security Innovation will enumerate each component of the system that influences the Threat Model.

These features include the following:

- **Assets** – Any high value information the attacker may target
- **Roles** – Each different level of privilege on the system a malicious user may acquire
- **Components** – The physical pieces of the system that may hold assets, validate roles, or connect other components

Assets

- **Authentication Credentials** – When using components that require authentication or role validation, credentials must first be sent to the server
- **Authentication Tokens –** Tokens are sent with each request to maintain a session between a client's browser and the server
- **User Profile information and settings –** User's name, login information, or other data associated with specific accounts, including PII for users on the system that could be used for social engineering attacks
- **Permission Information –** Functional Security Group settings
- **Internal Financial Information –** Sales figures, internal purchasing figures, other information related to the financial health of the organization operating the software, and information related to prices, deals, and forecasting internal to the organization
- **Web Content Integrity –** The User Interface on the webpage in HTML or JavaScript
- **Web server resources**– Processes and bandwidth that may be compromised through command injection.
- **Users' resources –** Processes and bandwidth that may be compromised through command injection.
- **Application IP** – Source code, configuration files, and other internal information that should not be exposed

Roles

- **Anonymous** – An unregistered user on the system (should never have any permissions)
- **User –** A user on the system (effective permissions are controlled within the system by Functional and Data group security
- **Vendor –** A user on the system restricted to read only access on only data specifically assigned to the vendor and no other access

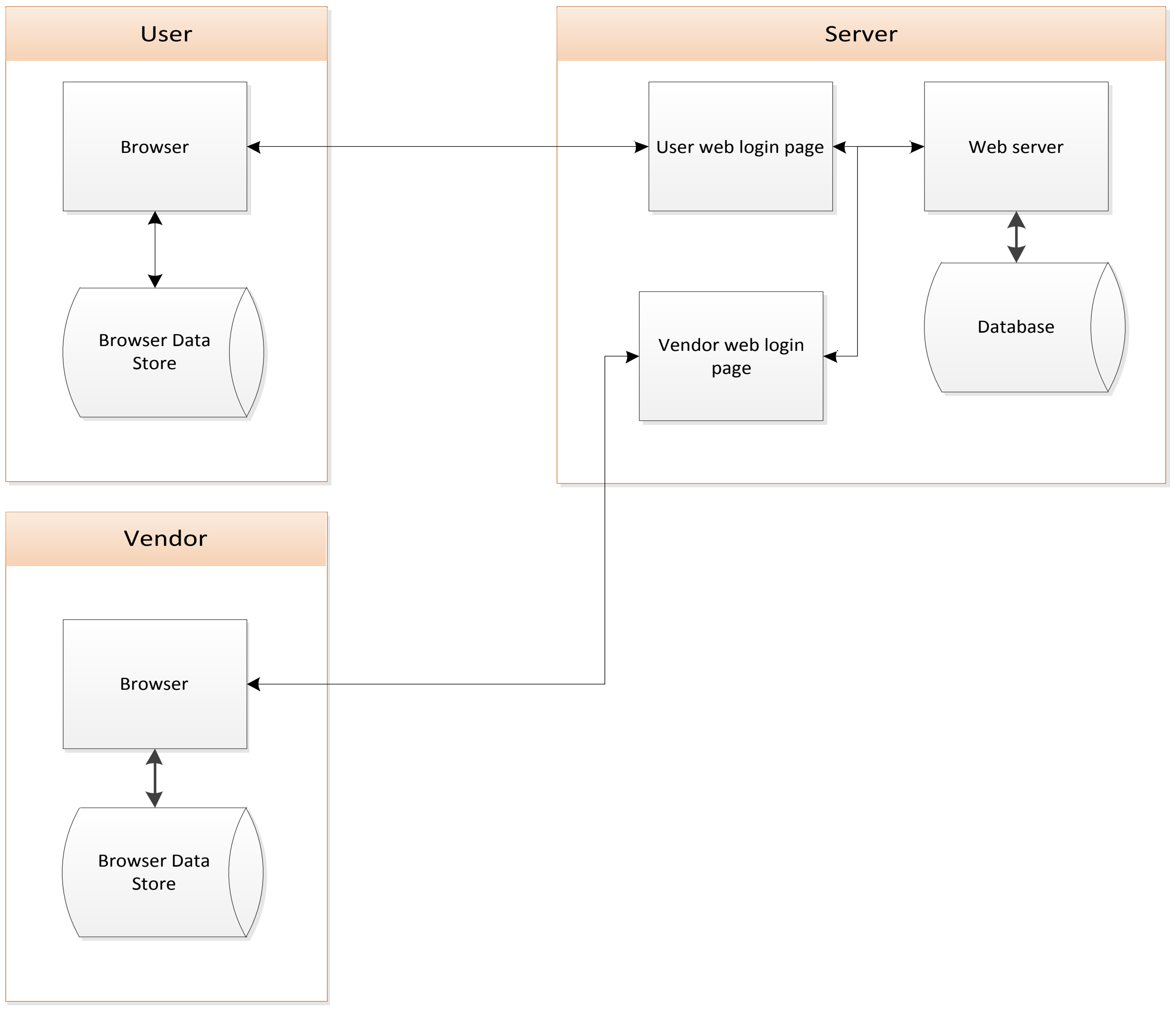
Components

Each component in the following table may hold zero or more assets. The component may store, process or transmit data on the system.

Component Name	Type	Description
User web login	web site	The primary interface to the application used by employees of the client who installed the instance to login.
Vendor web interface	web site	A restricted interface that allows read only access to only specific information.
Web server	web site	The web server shared by all users of the system.
Database	database	The backend data store for all data on the system.
User's browser	browser	The user's interface to the web interface.
User's browser's data store	browser data store	The cache and cookie storage mechanism of the user's browser
Vendor's browser	browser	The vendor's interface to the web interface.
Vendor's browser's data store	browser data store	The cache and cookie storage mechanism of the vendor's browser

Component Diagram

The following component diagram illustrates the relationship between each component in the system. Components highlighted in green or blue are noted above, other components are supporting components and fall outside of the scope of this assessment.



Activity Matrix

The following Activity Matrix shows the interactions between each asset and role in the Application.

Action
Create
Read
Update
Delete

Frequency
Always
Sometimes
Never
Not Applicable

		Role		
Asset	**Action**	**Anonymous**	**Vendor**	**User**
Authentication Credentials	**Create**	Never	Never	Sometimes[2]
	Read	Never	Never	Sometimes[2]
	Update	Never	Never	Sometimes[2]
	Delete	Never	Never	Sometimes[2]
Authentication Token	**Create**	Never	Always	Always
	Read	Never	Sometimes[1]	Sometimes[1]
	Update	Never	Never	Never
	Delete	Never	Sometimes[1]	Sometimes[1]
User Profile information and settings	**Create**	Never	Never	Sometimes[2]
	Read	Never	Sometimes[1]	Sometimes[2]
	Update	Never	Never	Sometimes[2]
	Delete	Never	Never	Sometimes[2]
Permission information	**Create**	Never	Never	Sometimes[2]
	Read	Never	Never	Sometimes[2]
	Update	Never	Never	Sometimes[2]
	Delete	Never	Never	Sometimes[2]
Internal Financial Information	**Create**	Never	Never	Sometimes[3]
	Read	Never	Never	Sometimes[3]
	Update	Never	Never	Sometimes[3]
	Delete	Never	Never	Sometimes[3]
Web server resources	**Create**	Never	Never	Never
	Read	Never	Never	Never
	Update	Never	Never	Never
	Delete	Never	Never	Never
Web Content Integrity	**Create**	Never	Never	Never
	Read	Never	Never	Never
	Update	Never	Never	Never
	Delete	Never	Never	Never

Users' resources	**Create**	Never	Never	Sometimes[1]

	Read	Never	Never	Sometimes[1]
	Update	Never	Never	Sometimes[1]
	Delete	Never	Never	Sometimes[1]
Application IP	**Create**		NA	
	Read	Never	Never	Never
	Update		NA	
	Delete		NA	

The following rules apply to the items labeled "Sometimes" in the Activity Matrix above:

- Sometimes[1] – This user can perform this task, but only for assets they own.
- Sometimes[2] – This user can perform this task, but only for assets they own or if they have been allocated permissions to perform this action.
- Sometimes[3] – This user can perform this task, but only if they have been allocated permissions to perform this action.

Threat Tree Information

The following section contains the complete list of threat trees developed for the Application. Each threat includes the priority and a description of the potential threat. Beneath each the threat header is the component that is affected, the sub bullets under each asset represent attack scenarios that could make the threat possible.

The following list represents theoretical threats against the system. A test plan will be created using these threat trees. The individual tests against the live system will show or disprove the existence of each threat.

Threat Priority

The priority rating for each threat is based upon the perceived damage impact to the asset.

- **P1:** Significant system compromise via elevation of privilege, disclosure of sensitive assets, or tampering with/repudiation of critical system activities.
- **P2:** Server side or widespread denial of service, disclosure of implementation detail or less sensitive assets, non-critical repudiation/logging issues.
- **P3:** Client side or minor denial of service, alteration of user experience without affecting functionality, minor information disclosures.

Glossary

- **SQLI**: SQL Injection – enables an attacker to inject SQL commands that alter the meaning of a query to view or modify the database.
- **XSS**: Cross-Site Scripting – enables attackers to execute client-side code in users' browsers, to steal session information or perform other client-side attacks.
- **CSRF**: Cross Site Request Forgery – enables an attacker to force a victim to perform actions on the behalf of the attacker
- **DoS**: Denial of Service – enables attackers to prevent other user's access to an application or service.

Threat Tree Details

Legend

Priority: Threat (STRIDE Type)

- Related Component
    - Conditions

P1: Code can be executed directly on the server (Elevation of Privilege)

- Web server
    - Code uploaded directly to be executed by the web server then called by an attacker or executed automatically by the server
    - A component may contain a vulnerability that allows remote or local code inclusion
    - A component written in a low level language may have a vulnerability that allows code to be executed in the context of the web server (buffer overflow, string format vulnerability)

P1: Authentication credentials are sent insecurely (Information Disclosure)

- Authentication system
    - SSL is not used or improperly configured
    - Credentials are cached to insecure locations
    - Credentials are sent as GET parameters that may be cached

P1: Authentication system can be bypassed (Elevation of Privileges)

- Web Server
    - SQL injection or command injection in the login field
    - Predictable authentication tokens allow tokens to be crafted directly bypassing authentication
    - Command or SQL injection on a page that can be directly accessed allows direct access to the database
    - Weak tokens can be brute forced
    - SQL injection in other fields allows for arbitrary database writes

P1: Any User is able to create, update, or delete any user or role (Elevation of Privilege, Tampering, Denial of Service)

- User administration page
    - Direct request against administration pages allows users to be created, updated, or deleted without logging in
- All accessible pages
    - SQLi in any page allows direct access to the backend database

P1: Authentication can be disabled (Denial of Service)

- Authentication system
  - SQL injection or command injection allows authentication to be broken
  - Special characters can be send in to the authentication system that will disable the system
  - Existing malformed data can force the system to stop checking credentials

P1: An attacker is able to force another authenticated user to perform an action on their behalf (Elevation of Privileges)

- Web server
  - Application fails to generate and verify a unique token for every request (CSRF)
  - Session tokens can be fixated allowing an attacker to create a token that will be used later on

P1: Internal Product Information, Financial Information, or Location Information can be created updated, read, or delete by an Anonymous user, User unauthenticated to access the information, or Vendor (Information Disclosure, Elevation of Privilege).

- Web Server
  - Pages dealing with internal product or financial information may be accessed by direct request
- Database
  - SQLi may allow access to information that would otherwise be inaccessible to the user

P2: Anyone can change the structure of web pages in a temporary or permanent way (Elevation of Privileges, Tampering, and Denial of Service)

- Web server
  - Reflected XSS allows an attacker to temporarily change the structure of a page for use against another target
- Database
  - Data can be stored in the database without verification that can permanently change the structure of a web page

P2: Web server configuration issues may allow the web server to be used as a proxy to attack internal network components (Elevation of Privilege)

- Web server
  - Configuration issues could allow traffic to be proxied to the internal network

P3: Application IP (source code, internal documents, etc.) may be accessible via external web interface (Information Disclosure)

- Web server
  - Internal documents may be accessible via direct request
  - Configuration or other vulnerabilities may allow source code or internal documents to be retrieved

P3: Specially crafted data could be placed into the database that could be used to create reports that exploit client applications (Elevation of Privilege)

- Reporting system (web server)
  - Specially crafted data could exploit flaws in reporting software which could be used to inject XSS into to PDF reports, or create reports that could exploit user's document viewers