A Prototypical Security Architecture for Securing Web Applications using Secure Sockets Layer (SSL)

GRADUATE PROJECT REPORT

Submitted to the Faculty of
the Department of Computing Sciences
Texas A&M University-Corpus Christi
Corpus Christi, Texas

in Fulfillment of the Requirements for the Degree of
Master of Science in Computer Science

by

Konsam Urmila Devi
Fall 2008

Committee Members

Dr. Mario Garcia
Committee Chairperson

Dr. David Thomas
Committee Member

Dr. Long-zhuang Li
Committee Member
A Prototypical Security Architecture for Securing Web Applications using Secure Sockets Layer (SSL)

GRADUATE PROJECT REPORT

Submitted to the Faculty of
the Department of Computing Sciences
Texas A&M University-Corpus Christi
Corpus Christi, Texas

in Fulfillment of the Requirements for the Degree of
Master of Science in Computer Science

by

Konsam Urmila Devi
Fall 2008

Committee Members

Dr. Mario Garcia
Committee Chairperson

Dr. David Thomas
Committee Member

Dr. Long-zhuang Li
Committee Member
ABSTRACT

Security architecture is a blueprint of how to place resources optimally in an IT environment to support businesses, with security as a critical component. Corporations face challenges in doing this as data and applications in a corporate network need to be accessed by several entities within and outside the network. As organizational units in an enterprise create new Web applications, it is important that they take care of all security loopholes at the Web application level.

This project aims at first designing an enterprise wide Security Architecture that will serve as a guideline or a reference to translate new business needs into secure application platforms, that comply with the enterprise wide Security Architecture. Texas A&M University, Corpus Christi (TAMU-CC), is the model organization for which the Enterprise wide Security Architecture is designed. The designed Security Architecture and enterprise wide security standards are then used to design a target security architecture for a Web application in TAMU-CC. The target architecture is designed to comply with the enterprise wide security standards and architecture.

The Secure Sockets Layer (SSL) protocol is used to provide secure connections between applications over the network/Internet. This project concentrates on implementing SSL on Web server as well as Weblogic application server. This research also focuses on generating private keys using various open source tools, obtaining self-signed or digitally signed certificates from the Certificate Authority (CA) and applying them to the above Web and application servers.
# TABLE OF CONTENTS

ABSTRACT ........................................................................................................................ ii
TABLE OF CONTENTS ................................................................................................... iii
LIST OF FIGURES ............................................................................................................ v
LIST OF TABLES ............................................................................................................. vi

1. INTRODUCTION AND BACKGROUND ................................................................ 2
   1.1 Introduction to Security Architecture ............................................................... 2
   1.2 History of SSL .................................................................................................. 2
   1.3 Basic Cryptography .......................................................................................... 4
       1.3.1 Encryption ............................................................................................. 4
       1.3.2 Secret Key Encryption .......................................................................... 4
       1.3.3 Public Key Encryption .......................................................................... 5
   1.4 Secure Communication using SSL ................................................................... 5
   1.5 Advantages of using SSL .................................................................................. 7
   1.6 Disadvantages of using SSL ............................................................................. 7

2. A PROTOTYPICAL SECURITY ARCHITECTURE FOR SECURING WEB
   APPLICATIONS USING SECURE SOCKETS LAYERS ............................................... 9

3. SYSTEM DESIGN .................................................................................................... 12
   3.1 Designing a Security Architecture .................................................................. 12
       3.1.1 Security Architecture Overview ......................................................... 12
       3.1.2 Security and Architectural Principles ................................................. 14
       3.1.3 Security Architectural Design for TAMU-CC .................................... 17
       3.1.4 Target application architectural design ............................................... 31
       3.1.5 Design Phase ....................................................................................... 31
       3.1.6 Implementation Phase ......................................................................... 33
       3.1.7 Securing application platform with SSL ............................................. 36
   3.2 Building Web servers ...................................................................................... 39
       3.2.1 Apache ................................................................................................ 39
       3.2.2 iPlanet ................................................................................................. 40
   3.3 Building Application server ............................................................................ 41
LIST OF FIGURES

Figure 1.1 SSL Operating above the TCP ................................................................. 3
Figure 1.2 Secret Key Encryption [Chou 2002] ....................................................... 4
Figure 1.3 Public Key Encryption [Chou 2002] ......................................................... 5
Figure 2.1 High Level Overview of Project Results .................................................. 11
Figure 3.1 High Level Architectural Diagram ......................................................... 22
Figure 3.2 TAMU-CC SSO Logon Page ................................................................. 23
Figure 3.3 SSO Logon ......................................................................................... 25
Figure 3.4 Application Architecture ................................................................. 32
Figure 3.5 SSO Architecture ....................................................................... 33
Figure 4.1 Administrative Server Log ............................................................... 42
Figure 4.2 Administrative Console ................................................................. 43
Figure 4.3 Managed Server Log ................................................................. 43
Figure 4.4 Administrative Console for Servers .............................................. 44
LIST OF TABLES

Table 3.1 Externally Controlled Domain ................................................................. 18
Table 3.2 Uncontrolled Domain ............................................................................. 19
Table 3.3 Controlled Domain ................................................................................ 19
Table 3.4 Restricted Domain .................................................................................. 20
Table 3.5 Secured Domain ...................................................................................... 21
Table 3.6 Comparison of Entrust GetAccess and RSA Access Manager ............... 26
Table 3.7 Server Matrix .......................................................................................... 34
Table 3.8 Application Matrix .................................................................................. 34
Table 3.9 Software Matrix ...................................................................................... 34
Table 3.10 Application and Group ID Matrix .......................................................... 35
Table 3.11 Firewall Ports Matrix ............................................................................. 35
Table 3.12 Load Balancer Matrix .......................................................................... 36
Table 3.13 Application Certificate Matrix ............................................................... 39
1. INTRODUCTION AND BACKGROUND

1.1 Introduction to Security Architecture

The objective of security is to guard data and networks against attacks. Corporations face challenges in doing this as data and applications in a corporate network need to be accessed by several entities within and outside the network. There are several technologies in the market that help in securing corporate networks. Implementing and using these technologies effectively to secure a corporate network involves understanding of potential risks in the network, analyzing the consequences of security breaches at various levels in the corporate network and incorporating security architecture at every level of the corporate network based on the analysis.

Apart from securing a corporate network as a whole, it is important to secure every Web based application in the organization. This involves using good authentication and authorization mechanisms to allow users appropriate access to corporate resources, using monitoring and auditing techniques to detect vulnerabilities and also using appropriate incident resolution techniques to resolve any unforeseen security compromises.

1.2 History of SSL

As enterprise businesses are increasing tremendously, online security becomes more crucial. Hence, it is important that organizations implement protocols which are capable of protecting sensitive data, maintaining data integrity, authenticate senders and protect Web sites from attacks and unauthorized access.
For Virtual Private Networks (VPN), the protocol should be able to provide strict access control and data protection. Whereas, for e-business, the protocol should be able to provide easy access but provide authentication and data security. To provide this need, Secure Sockets Layer (SSL) has been widely implemented. [Chou 2002] The Secure Sockets Layer (SSL) protocol is used to provide secure connections between applications over the network/Internet by authenticating the applications’ identities and encrypting the exchanged data.

SSL was first developed by Netscape in order to provide secure connections between the clients and servers over the Internet. [Webopedia 2007] In order to provide unified solution even for non-HTTP applications, Netscape developed SSL to sit on top of Transmission Control Protocol (TCP) as shown in Figure 1.1, hence providing TCP-like interface for the upper layer applications. [Chou 2002]

![Figure 1.1 SSL Operating above the TCP](image-url)
1.3 Basic Cryptography

Cryptography is the practice and study of hidden information. It deals with protecting information by data encoding and transformation techniques. Following are some of the cryptography details which are useful in understanding the working of SSL.

1.3.1 Encryption

Encryption is the process of transforming information (plaintext) to make it unreadable to anyone except those possessing special knowledge, usually referred to as a key. The result of the process is encrypted information (referred to as ciphertext). The reverse process is called decryption. [Wikipedia 2007]

1.3.2 Secret Key Encryption

![Figure 1.2 Secret Key Encryption](Chou 2002)

In Secret Key Encryption, both the sender and receiver use the same key to encrypt and decrypt data [Wikipedia 2007] as shown in Figure 1.2. Some of the secret key algorithms are DES (Data Encryption Standard), Triple DES, AES (Advanced Encryption Standard) and RC4. One problem with secret key encryption is distributing keys in a secure and reliable way. [Chou 2002] If the key distribution is not secure, anyone can intercept and decrypt the message.
1.3.3 Public Key Encryption

In Public Key Encryption, two keys, private and public keys are used. Normally, the sender uses the public key to encrypt the data and the receiver uses its private key to decrypt it as shown in Figure 1.3. In some cases, the sender can use the private key to encrypt the data and the receiver can only decrypt using the associated public key. [Weaver 2006] This is used in message authentication like in digital signature. It is not possible to decrypt the message with only the public key knowledge. Public key encryption does not have the key distribution problem however; they require more computing power than secret key algorithm. [Chou 2002]

In order to utilize the strengths of both the algorithm, a slightly variant public key agreement algorithm is often used by security protocols. SSL uses the RSA public key algorithm, which is used to distribute secret-key parameters.

1.4 Secure Communication using SSL

SSL communication takes place in two phases, handshake and data transfer. [Chou 2002] During the handshake phase, the public key encryption algorithm is used to determine the secret key parameters. During the data transfer phase, both the server and the client use the secret key to encrypt and decrypt the data.
The client initiates an SSL connection by first sending a message, which contains a list of secret-key algorithms called cipher-specs. The server responds with a similar message selecting the preferred cipher specs and then it sends a certificate containing the server’s ID, its public key and other parameters. The certificate is a set of data that validates the server’s identity. Some of the major certificate authorities are Entrust, Geotrust, Thawte and Verisign. The Certificate Authority (CA) generates the certificate and verifies its authenticity. [Wikipedia 2007] The server sends its public key to the CA in a secure channel. The CA then creates the certificate, which contains its own ID, the server’s ID, the server’s public key and other information. The CA creates a certificate fingerprint and encrypts the fingerprint with its private key to create a certificate signature.

In order to validate a server’s certificate, a client uses the CA’s public key to decipher the signature and then read the fingerprint. The client independently computes the certificate’s fingerprint and if the two fingerprints do not match, then the certificate has been tampered. [Chou 2002] After the client authenticate the server, both the server and the client use a public key algorithm to determine secret key information. When both the server and the client are ready to begin using the secret key, the handshake phase is completed.

During the data transfer phase, the outgoing messages from both the server and the client are broken into fragments and Message Authentication Codes (MACs), which are fingerprints computed from message contents, are appended to them. While transmitting, the data fragment, MAC and the record header are combined and encrypted with a secret key to produce a SSL packet. And while receiving, the packet is decrypted;
the MAC address is computed and compared to the received MAC. [Chou 2002] This is how secure communication is achieved using SSL.

1.5 Advantages of using SSL

Since the data is encrypted during the transmission in SSL communication, the data transfer is protected from Internet traffic eavesdroppers from reading or changing the data during the transit. This is especially useful in shared environments, such as a college campus or a large office which uses standard 10 base T Ethernet hubs or thin Ethernet, where all machines can see all traffic. Using secure data transfer also helps increase business since people feel safe to submit personal and credit card details over the secure sites. The SSL communication also allows the Web browser to authenticate the Web server. It can also be noted that the SSL is the mostly widely used protocol for VPN security since it uses the same encryption protocols as many e-commerce sites and Web-enabled applications, which makes it more compatible with the networks, the remote workers connect through. The SSL is also simple to install and it makes use of firewall ports that have already been opened to secure the Internet traffic, enabling the users to connect into a network securely via a standard Web browser without the need to install special software. This alleviates connectivity problems with firewalls and negates the need to install software on the remote clients. [Oien, 2007]

1.6 Disadvantages of using SSL

The SSL communication requires SSL-compatible Web browser to access data from the server. It requires additional setup and maintenance for managing certificates. It
also requires additional system resources to perform SSL handshake and to encrypt and decrypt the data. [VM:Webgateway 2007]

In SSL VPN, the security of any client connecting to the network must be heavily scrutinized since it offers a much greater choice of client platform. And in order to avoid systems being left connected to the network by the users, sessions will be terminated after a period of inactivity. Also, to ensure the correct user is still using the connection, periodic authentication during a session is implemented which may be annoying for the users.
A PROTOTYPICAL SECURITY ARCHITECTURE FOR SECURING WEB APPLICATIONS USING SECURE SOCKETS LAYERS

The goal of this project was to design an enterprise wide secure architecture for an organization, which can be used as a model by different organizational units to decide the physical location of their IT components and to design secure platforms. Texas A&M University, Corpus Christi is the organization for which a secure architecture was designed in the course of this project. An Enterprise Security Architectural diagram was designed based on

- Enterprise Architectural Standards and
- Enterprise Security Standards

The standards were formulated for TAMU-CC. The resultant Security Architecture was a diagram detailing the different networks, essential firewalls and authentication and authorization methods. This is a snapshot of the enterprise network that every department in TAMU-CC should be aware of before designing any departmental web applications that will be integrated with the enterprise network.

A target Web application architecture was then designed for the department of Computing and Mathematical Sciences (CAMS) at TAMU-CC. A secure platform was designed for a Java Web application. Making a Web application available on the Internet involves various components like the load balancers, Web servers, application servers and database servers. If the transactions on a Web site need to be secured, the application needs to be secured at every level of the architecture.

Designing a target architectural diagram and implementing it, involved identifying the security steps that need to be taken by the application platform architects to secure the application. The step by step process of requesting IDs for separation of
duties, requesting firewall holes etc. were first identified. Then a target secure platform was designed for the Web application. In order to make sure that end users have secure transactions end to end, methods to determine what points of the Web and Application servers need to be secured using SSL were outlined. The result was a target architectural diagram and a checklist of things to be done in order to secure applications at various tiers.

Real time Web servers and application servers were built in order to implement the application level architecture. A Web application was hosted on the application server and was accesses through the Web server which acted as a proxy server. The Web applications hosted on these servers were protected by implementing SSL security.

Securing applications involves generating a private key, generating a certificate signing request, self-signing the certificate in order to generate a private key, determining methods of storing the SSL certificates like files, keystores etc. and finally installing the certificates on the Web server and application server. Apache Web server and Weblogic application server were built as part of this project. The flowchart in Figure 2.1 gives an overview of what was achieved in this project.
Understand TAMU-CC User base and Access Zones

Design Enterprise Security Architectural diagram.

Recommend a Single-sign product for authentication, based on comparing two market leaders.

Formulate Architectural Policies.

Design Target Security Architectural diagram for a sample Java application.

Formulate a checklist of server level, application level and network level items that need to be accomplished to secure Web applications.

Implement part of the application architecture on a stand alone Linux system.

Figure 2.1 High Level Overview of Project Results
3. SYSTEM DESIGN

3.1 Designing a Security Architecture

“A chain is only as strong as its weakest link” and is very true when one talks of how secure a network is. It is important to acknowledge that there is no single security architecture that can be applied across all organizations. The “Security Architecture” designed will serve as a guideline or a reference that will help translate business requirements into an architecture that will meet the overall security requirements of an enterprise. The Security Architecture is a generalized solution and is not specific to any software or application. The reference architecture is then used as a guideline to arrive at a target architecture that is realized using specific Web server and application server products. The enterprise security architecture designed in this project is described using Texas A&M University, Corpus Christi (TAMU-CC) as the target organization, but can be used as a model for any enterprise.

3.1.1 Security Architecture Overview

In order to design a model security architecture for TAMU-CC, the architect first needs to understand the following:

- Who are the users that access the resources of TAMU-CC?
- What access methods or channels do the users of TAMU-CC resources use?
- Where do the users access the TAMU-CC resources from?
- What are some of the common software/applications or services that TAMU-CC needs to use or can use to realize a comprehensive security architecture which can then be used by every department to secure their applications?
3.1.1.1 Users

Users are entities that need to interact with TAMU-CC’s network or systems and consume the resources of TAMU-CC or provide services to TAMU-CC. This includes the following:

Employees People who work directly for TAMU-CC through its various departments.

Customers People who consume TAMU-CC resources like the current and prospective students, professors etc.

Third Party An external entity that provide services to TAMU-CC or whose resources TAMU-CC consumes. For example, IEEE resources that TAMU-CC consumes, interlibrary loan facilities that TAMU-CC uses.

3.1.1.2 Access Methods

Access Methods are different channels used by the users to access TAMU-CC resources. This includes the following:

Web Browser A thin client that communicates with systems using various protocols like HTTP, HTTPS and FTP.

Software Applications A software tool can access various resources using specific business logic embedded in the software.

Email/Fax/wireless Other methods used to communicate from or with the TAMU-CC systems or network.
3.1.1.3 Access Zones

Access zones are different networks that the users use to log into the TAMU-CC network. Users could login through one of the following:

**Internet** Accessible by all.

**Intranet** TAMU-CC internal network and can be accessed only by authorized users.

**Extranet** Network that other TAMU-CC partners networks login from.

**Wireless network** Users connect to the TAMU-CC network over a wireless connection.

3.1.2 Security and Architectural Principles

Enterprise Security Architecture needs to be designed in alignment with the security principles. In order to design a secure architecture, it is important that security policies are decided in sync with the enterprise architectural policies. [NIH 2006]

3.1.2.1 Enterprise Architectural Policies

Enterprise Architecture is considered an important key in any organization in order to reduce costs and risks and enhance profits and efficiency. The enterprise architecture planning process is very tedious and can very easily lead to a lot of ambiguity and even failure of the process. [Akkasi 2008] Starting off with a set of principles or best practices will help speedup the process.
The following are the enterprise architectural policies that have been arrived at in this project:

1. **Strive for simplicity** An enterprise is made up of several Organizational Units. Consider the various departments in TAMU-CC like the CAMS, S&T, Business Office, Library etc. It is important that each of the departments understand that there is an organizational wide architecture and follows it as it.

2. **Reuse** Architectural design strategy should emphasize the importance of the reuse of existing software [Dabous 2006] and systems, which will in turn lead to cost effectiveness and easy maintenance.

3. **Architectural Views** Enterprise architecture can be divided into different views of architecture like the business, data, application, application architecture and technology architecture [Carver 1990]. The technology architecture is a combination of the business architecture, the data architecture and application architecture [Bruce 2005]. The data architecture helps the organization develop an information architecture, which the entire organization can use instead of each department developing its own database model. The business architecture complies with the business requirements of the organization and the application architecture helps application architects with a reference to work with, in order to develop the application infrastructure where the applications will be hosted.

4. **Adopt Open Solutions** It is very easy that a group of technology architects often try to implement new and innovative solutions. But it is often very difficult to keep the organizational units well informed about the architectural design so they
could use it as a framework and secrets degrade over time. Open solutions are easy to understand and keep up with.

3.1.2.2 Architectural Policies

The following are the architectural policies that have been arrived at in this project:

1. Enforce Separation of duties

   Separation of duties is a concept of having multiple people to perform a task. [Wikipedia 2008] For example, in the area of application support, no one team should perform more than one type of duty. Application server support team should not support database operations and vice versa. The primary objective of this is preventing errors and fraudulent activities.

2. Defense in Depth

   C.L Smith in his paper discusses the concept of “Defense in Depth” as a series of barriers that surround the asset to be protected. Using the “Defense in Depth” strategy delays access to a resource. [Smith 1999] Given the fact that attackers could be insiders or outsiders, organizations need to put in checks at multiple points. This ensures that hackers, who get access to systems located at one layer of the enterprise, cannot launch attacks on systems that are located at other restricted layers.

3. Provide the least privileges necessary

   This will make sure that the performer does not accidentally or intentionally perform activities that they are expected to do.
4. Deny before Permit

Denying access to a resource before permitting makes sure that the correct authentication and authorization principles are in place [Bruce 2005].

5. Secure Network Enclaves

[Brunette 2005] in their paper "Toward systemically secure IT architectures" talk about secure network enclaves, which are building blocks that are used to compartmentalize and control access to communities of users and services at a network level. These secure enclaves can be used to group together networks of people, e.g., people residing in the CAMS building and library need to belong to different networks.

6. Thin Client Architecture

Thin client Architecture is one where a central server is used to mediate all communication between the end user and resources. [Haigh 1995] This architecture goes a long way in securing information, as all significant information is stored on the central server that is adequately protected. [Haigh 1995]


The authentication and authorization methods should be common to the entire organization. This will make user and role management very easy.

3.1.3 Security Architectural Design for TAMU-CC

The Enterprise Architectural policies and Security Architectural policies arrived at in the previous sections were used to design a target architectural design for TAMU-CC.
3.1.3.1 Domains

Based on the “Defense in Depth” security policy, the TAMU-CC network can be first divided into 5 domains as follows:

Table 3.1 Externally Controlled Domain

<table>
<thead>
<tr>
<th>Externally Controlled Domain</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>An environment outside of TAMU-CC’s direct control that accesses TAMU-CC data or resources. This domain is responsible for implementing its own security policies, model, and architecture. Where required by law or TAMU-CC policy, TAMU-CC must verify that the domain meets a minimum standard of security and is maintained to that standard by contract.</td>
</tr>
<tr>
<td>Perimeter Controlled By:</td>
<td>External Entity.</td>
</tr>
<tr>
<td>Inter-domain Data Transfers:</td>
<td>• TAMU-CC Confidential and Confidential/Proprietary data must be encrypted.  • Security credentials must be encrypted.</td>
</tr>
<tr>
<td>Intra-domain Data Transfers:</td>
<td>• TAMU-CC Confidential and Confidential/Proprietary data must be encrypted if being sent to or received from a publicly accessible computer.  • Security credentials must be encrypted.</td>
</tr>
<tr>
<td>Intra-domain Data Storage:</td>
<td>• Confidential and Confidential/Proprietary data must be encrypted if stored on a publicly accessible computer.  • Security credentials must be encrypted.</td>
</tr>
<tr>
<td>Access Rules:</td>
<td>• Access to this domain must adhere to TAMU-CC standards and be limited by the External Entity at the domain boundary to protect TAMU-CC data.  • Administration of any node within this domain is generally controlled by the External Entity.</td>
</tr>
<tr>
<td>Examples:</td>
<td>A third party supplier that has a business relationship with TAMU-CC.</td>
</tr>
</tbody>
</table>
Table 3.2 Uncontrolled Domain

<table>
<thead>
<tr>
<th>Definition:</th>
<th>An environment with unknown or inadequate security policies, models, and architecture by TAMU-CC. standards. The domain is outside of TAMU-CC’s control or direction and must be considered hostile and insecure.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perimeter Controlled By:</td>
<td>Various Entities.</td>
</tr>
</tbody>
</table>
| Inter-domain Data Transfers: | • TAMU-CC Confidential and Confidential/Proprietary data must be encrypted.  
• Security credentials must be encrypted. |
| Intra-domain Data Transfers: | • TAMU-CC Confidential and Confidential/Proprietary data must not be sent to or received from a computer in this domain.  
• Security credentials must be encrypted. |
| Intra-domain Data Storage: | • Confidential and Confidential/Proprietary data must not be stored in this domain.  
• Security credentials must not be stored in this domain. |
| Access Rules: | • Access is limited at domain boundaries by other domains.  
• Various entities control portions of the internal access rules.  
• Administration of any node within this domain is generally not under the control of TAMU-CC. |
| Examples: | Internet, internet café, cardholders’ computers |

Table 3.3 Controlled Domain

<table>
<thead>
<tr>
<th>Definition:</th>
<th>An environment that can be the initial layer of external user access to TAMU-CC’s resources from un-trusted external domains while also providing a layer of protection and isolation. The domain should consider all external domains to potentially be hostile.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perimeter Controlled By:</td>
<td>TAMU-CC.</td>
</tr>
</tbody>
</table>
| Inter-domain Data Transfers: | • TAMU-CC Confidential and Confidential/Proprietary data transferred from this domain to Externally Controlled and Uncontrolled domains must be encrypted.  
• TAMU-CC Confidential and Confidential/Proprietary data should be encrypted if being transferred to or from a Restricted domain. |
• Security credentials must be encrypted.

Intra-domain Data Transfers:
• TAMU-CC Confidential and Confidential/Proprietary data must be encrypted if being sent to or received from a publicly accessible computer.
• Security credentials must be encrypted.

Intra-domain Data Storage:
• Confidential and Confidential/Proprietary data must be encrypted if stored on a publicly accessible computer.
• Security credentials must be encrypted.

Access Rules:
• Direct inbound access is allowed from Uncontrolled Domains or Externally Controlled Domains but not both.
• Limited inbound and outbound access is allowed from a Restricted Domain.
• Access is controlled at the domain boundaries.
• Administration of any node within this domain must be from the Restricted Domain and employ encryption.

Examples: Demilitarized Zone (DMZ).

<table>
<thead>
<tr>
<th>Table 3.4 Restricted Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Restricted Domain</strong></td>
</tr>
<tr>
<td><strong>Definition:</strong></td>
</tr>
<tr>
<td><strong>Perimeter Controlled By:</strong></td>
</tr>
<tr>
<td><strong>Inter-domain Data Transfers:</strong></td>
</tr>
<tr>
<td><strong>Intra-domain Data Transfers:</strong></td>
</tr>
<tr>
<td><strong>Intra-domain Data Storage:</strong></td>
</tr>
<tr>
<td><strong>Access Rules:</strong></td>
</tr>
</tbody>
</table>
Table 3.5 Secured Domain

<table>
<thead>
<tr>
<th>Secured Domain</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Definition:</strong></td>
<td>A TAMU-CC environment that is isolated from the restricted domain, and may contain systems and data that exhibits a high degree of trustworthiness (integrity, confidentiality) and is protected accordingly. A Secured Domain only has connectivity to/from the Restricted Domain.</td>
</tr>
<tr>
<td><strong>Perimeter Controlled By:</strong></td>
<td>TAMU-CC.</td>
</tr>
</tbody>
</table>
| **Inter-domain Data Transfers:** | • TAMU-CC Confidential/Proprietary data should be encrypted.  
• Security Credentials must be encrypted. |
| **Intra-domain Data Transfers:** | • TAMU-CC Confidential/Proprietary data should be encrypted.  
• Security Credentials must be encrypted. |
| **Intra-domain Data Storage:** | • Confidential/Proprietary data should be encrypted.  
• Security Credentials must be encrypted. |
| **Access Rules:** | • Direct inbound access is allowed from Restricted Domains and other Secured Domains, but is limited at the domain boundary.  
• Outbound access from Secured Domains may be restricted, depending upon the business needs.  
• Administration of any node within this domain must be from within the Secured Domain or from the Restricted Domain or other comparably Secured Domains. |
| **Examples:** | eEngineering, ETI, isolated development areas. |
The high level architectural diagram of the placement of the previously explained domains is shown in Figure 3.1.

Figure 3.1 High Level Architectural Diagram
3.1.3.2 Single Sign On

TAMU-CC is already using the concept of Single Sign-On (SSO) as seen in Figure 3.2.

The concept of the Enterprise Architectural policy “software reuse” and the Enterprise Security policy “Enterprise wide Authentication and Authorization”, led to including SSO as part of the enterprise wide architecture in this project. Almost every big organization uses the SSO concept for identity and role management. [Zhao 2004] Some of the most popular SSO products in the market are Netegrity Site Minder, Entrust GetAccess and RSA AccessManager.
SSO allows users to log in once and gain access to all the applications that they are authorized to access without having to login again. [CAS 2008] For example, if a TAMU-CC’s student has access to the interlibrary loan as well as SAIL, if these two applications are configured to use SSO, the student will only be asked for credentials once and will be able to access both the applications that he is authorized to access.

3.1.3.2.1 Working of SSO

The general concept of the working of a SSO product is the same. Every SSO product has an agent that can be installed on the Web and application servers whose resources need to be protected. The SSO servers will have an entry for the client servers that are being protected. Whenever a protected resource on a server is being accessed, the SSO agent on the client machine checks the headers to see if there is an authenticated cookie. If this cookie is missing, the SSO agent sends the browser or end user to the SSO servers, where the user is presented with a sign-in page. Once the user enters his credentials, the SSO server checks the User ID and Password against its own database and authenticates the user. Once the user is authenticated, the user roles and policies are checked against the SSO database to determine the resources that the user is allowed to access. The user is then sent back to the original resource that he requested, which means, to the client server that the user initially accessed. As the user has an authenticated cookie in its headers, the SSO agent on the client allows the user to access the requested resource.
Figure 3.3 SSO Logon

3.1.3.2.2 SSO Products

Entrust getAccess and RSA Access Manager are the two SSO products that were considered for this project. A detailed study of the two SSO products was conducted and the following features and differences were drawn between the two products as shown in Table 3.5. While both the SSO products have number of features and differences, once those features that would help make a decision as to which SSO would work best for TAMU-CC are being considered in this project. The differences between the two products were drawn as a result of installation of the basic products and studying the pros and cons as well as from the theoretical details given by the vendor in their respective sites.
Table 3.6 Comparison of Entrust GetAccess and RSA Access Manager

<table>
<thead>
<tr>
<th>Components of the Main Product</th>
<th>Entrust GetAccess</th>
<th>RSA Access Manager</th>
</tr>
</thead>
<tbody>
<tr>
<td>Third Party Web site Authentication</td>
<td>Federated Identity Management (FIM) is a Part of the base product.</td>
<td>FIM is a separate Product.</td>
</tr>
<tr>
<td>Authentication Mode</td>
<td>Permit before Deny.</td>
<td>Deny before Permit.</td>
</tr>
<tr>
<td>Administration GUI</td>
<td>Part of the Base Product.</td>
<td>Separate Java API that needs to be installed on an application server.</td>
</tr>
<tr>
<td>GUI Complexity</td>
<td>GUI available to administer servers.</td>
<td>GUI available only to add client entries on the server, but not to administer the SSO servers.</td>
</tr>
<tr>
<td>Architecture</td>
<td>More complex than RSA.</td>
<td>The architecture is simpler to understand.</td>
</tr>
<tr>
<td>Server side client entries</td>
<td>Entries made automatically when client side agent is installed.</td>
<td>Need to make client entries manually on the server side.</td>
</tr>
</tbody>
</table>

3.1.3.3 Recommendation

After the above research and analysis of the two major SSO products as shown in Table 3.6, it was decided that RSA would be a better product for TAMU-CC’s SSO for the reasons in Table 3.5 which are explained next:
3.1.3.3.1 Simpler Architecture

RSA has a simpler and understandable architecture as the base product is separate from the following:

- FIM which is used for third party authentication.
- Administration application that is installed on a separate Application server.

This separation also allows organizations to buy only the base product if they do not need FIM which reduces the overall cost.

3.1.3.3.2 Deny before Permit

In the case of RSA once the agent is installed on the client side and the client entry is added on the server side using the GUI, everything under the root of that server is protected by default. If certain pages do not need SSO, they need to be explicitly unprotected which follows the Deny before Permit rule and hence it is more secure. In the case of Entrust, even though the client server entries are added on the server side, the resources are not protected and every resource has to be explicitly protected which follows the Permit before Deny rule. [Entrust 2008] It is possible that certain resource could be unintentionally unprotected.

3.1.3.3.3 Separation of Administrative Application

The separation of the Administrative application helps protect the SSO’s Administrative application itself through a different technology other than SSO.
3.1.3.4 Placing the SSO servers in the TAMU-CC domain

The SSO servers were then integrated in the TAMU-CC domain, after the following considerations:

3.1.3.4.1 Load balancer

Load balancers are used to distribute load among two or more servers according to the load balancing algorithm chosen. A load balancer is a direct point of contact for the Internet users. This requires that they are placed in an isolated domain from which hackers cannot compromise other enterprise systems. Hence, the SSO load balancer that distributes requests to https://sso.tamus.edu and https://TAMU-CCFIM.edu was placed in the DMZ domain.

3.1.3.4.2 SSO and FIM Web servers

Web servers act as proxy servers in front of actual servers so that the servers with confidential information like application logic and data are not directly exposed to the external world. [Dotti 1999] The Web servers receive requests from the Load Balancers and proxy the requests to the application logic or the middle tier. The Web servers act as a protective barrier to the application servers. Hence, the SSO and FIM Web servers were also placed in the DMZ zone to contain any attacks on the rest of the TAMU-CC network.
3.1.3.4.3 SSO and FIM Application Servers

The SSO and FIM application servers are placed in the restricted domain and will only receive requests from the proxy servers. Hence, only those ports that need to receive requests from the Web servers will be opened to the Web servers. All the other application server ports are closed to all domains.

3.1.3.4.4 Administrative Application

This application talks to the SSO servers and needs to be accessible only to the TAMU-CC administrators. Hence, this application server is placed directly in the restricted domain. The port where the administrative application is served from is SSL and password protected so that unauthorized TAMU-CC internal users do not access the application. This ensures separation of duties even within the TAMU-CC network.

3.1.3.4.5 LDAP and Databases

Information is the most important and the most confidential in any enterprise. Hence, Databases are placed in the secured domain and can be reached only from the restricted domain.

Based on the discussion in this section, the different SSO components were placed in the TAMU-CC domains. It is shown in Figure 3.1. This architectural diagram also represents the high level secure architectural diagram for TAMU-CC.
3.1.3.5 Secure Application Platform Design Standards

Every TAMU-CC department will have access to the overall Security Architectural diagram in Figure 3.1 which will be a guideline to decide the physical location of their servers. Table 3.1 is a security standards guideline that was developed in this project. Different departments within TAMU-CC can refer to the Architectural diagram and the security standards for application architecture in order to create new domain. The following are the design guidelines that every department can follow:

1. Obtain new servers in the appropriate domains. Front end proxy servers are essential and need to be placed in the DMZ. The backend application servers need to be placed in the restricted domain and all databases need to be in the secured domain which can be accessed only from the restricted domain.

2. Place requests to the firewall engineers to open appropriate ports only to necessary domains. By default all ports are closed to other domains.

3. On UNIX based systems, create new application IDs with least privileges to each ID, so that the applications can perform the minimal tasks they are required to.

4. If there is a need to access any ports from application servers directly, rather than protected end user Load Balancers Virtual IPs, make sure they are accessed only via a https URL.

5. If it is a secure application that needs authentication and authorization, use the company wide SSO authentication and authorization.

6. If additional authorization is needed, the application can authorize the authenticated user against its own database.
7. Make sure the Web servers, application servers and Database servers are protected at the local level.

3.1.4 Target application architectural design

The Security architectural design and the secure application platform design standards were used to design target application architecture. A secure application platform was designed for a new secure Web application for the CAMS department.

The application considered for this architecture was a java based application.

3.1.5 Design Phase

In the design phase, the following steps are performed:

- The application is a java based Web application that needs application servers in order to run.
- In order to comply with the enterprise level security architecture, the application servers need to be protected by front end Web servers that proxy requests to the application servers.
- The application servers in turn connect to the Database servers.
- The application needs authentication for which the enterprise wide SSO authentication and authorization is used.
- For additional authorization, the users are in turn authorized against the application Database. This can be achieved by checking if the User ID is present in the application database. Password authentication against the application database is not needed as the user is already authenticated and this is only a second level of authorization.
Using the information obtained while using the enterprise wide architectural diagram, the application architecture was designed as shown in Figure 3.4.

Once the application architecture was designed, the enterprise wide security policies were used to place the servers in appropriate domains. It is shown in Figure 3.1.

**Figure 3.4 Application Architecture**
3.1.6 Implementation Phase

All the tasks necessary to implement the architecture and secure it are explained in this section. The URL, Web server names, App server names and firewall ports were assumed to explain the implementation phase.
3.1.6.1 Obtaining the necessary servers

The servers in Table 3.7 need to be created in the appropriate domains to realize the architecture.

Table 3.7 Server Matrix

<table>
<thead>
<tr>
<th>URL</th>
<th>Purpose</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAMS WebServer1,</td>
<td>DMZ.</td>
<td>Web servers for Internet</td>
</tr>
<tr>
<td>CAMS WebServer2.</td>
<td>DMZ.</td>
<td>users.</td>
</tr>
<tr>
<td>CAMS WebServer3,</td>
<td>Restricted Domain (TIN).</td>
<td>Web servers for TAMU-CC</td>
</tr>
<tr>
<td>CAMS AppServer1,</td>
<td>Restricted Domain (TIN).</td>
<td>Application servers</td>
</tr>
<tr>
<td>CAMS AppServer2,</td>
<td>Restricted Domain (TIN).</td>
<td>behind the Web servers.</td>
</tr>
<tr>
<td>CAMS DB.</td>
<td>Secure Domain.</td>
<td>BackEnd Database server.</td>
</tr>
</tbody>
</table>

3.1.6.2 Obtaining the URLs

The URLs in Table 3.8 need to be created for the application.

Table 3.8 Application Matrix

<table>
<thead>
<tr>
<th>URL</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAMU-CC.externalapplication.com</td>
<td>End user URL.</td>
</tr>
<tr>
<td>TAMU-CC.internalapplication.com</td>
<td>URL that internal users will be redirected to when they access TAMU-CC.externallapplication.com.</td>
</tr>
</tbody>
</table>

3.1.6.3 Obtaining the necessary software

The software in Table 3.9 were considered for this project.

Table 3.9 Software Matrix

<table>
<thead>
<tr>
<th>Server</th>
<th>Software</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web server</td>
<td>Apache/iPlanet.</td>
</tr>
<tr>
<td>Application server</td>
<td>Weblogic/Websphere/Tomcat.</td>
</tr>
</tbody>
</table>
3.1.6.4 Obtaining necessary application IDs and Groups

The different application IDs and groups in Table 3.10 were created for the separation of duties.

Table 3.10 Application and Group ID Matrix

<table>
<thead>
<tr>
<th>Server Name</th>
<th>Application ID</th>
<th>Group</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAMS WebServer1, CAMS WebServer2, CAMS WebServer3, CAMS WebServer4.</td>
<td>Wsuser</td>
<td>gwsuser</td>
<td>Web server belongs to this ID.</td>
</tr>
<tr>
<td></td>
<td>Axmsso</td>
<td>gaxmsso</td>
<td>SSO agent on Web server belongs to this ID.</td>
</tr>
<tr>
<td>CAMS AppServer1, CAMS AppServer2, CAMS AppServer3.</td>
<td>Beasys</td>
<td>gbeasys</td>
<td>Weblogic server belongs to this ID.</td>
</tr>
<tr>
<td></td>
<td>Appbeasys</td>
<td>gappbeasys</td>
<td>Java application belongs to this ID.</td>
</tr>
<tr>
<td></td>
<td>Wlbeasys</td>
<td>gwlbbeasys</td>
<td>Weblogic domain belongs to this ID.</td>
</tr>
</tbody>
</table>

3.1.6.5 Firewall Port Requests

Firewall ports as shown in Table 3.11 need to be opened for communication between the different tiers.

Table 3.11 Firewall Ports Matrix

<table>
<thead>
<tr>
<th>Source</th>
<th>Destination</th>
<th>Ports</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>CAMS WebServer1, CAMS WebServer2.</td>
<td>CAMS AppServer1, CAMS AppServer2, CAMS AppServer3.</td>
<td>9070, 9072.</td>
<td>Proxy requests from Web servers to Application servers – firewall between DMZ and TIN.</td>
</tr>
<tr>
<td>CAMS AppServer1, CAMS AppServer2, CAMS AppServer3.</td>
<td>CAMS DB1.</td>
<td>1521.</td>
<td>Database connectivity from Application servers to the DB server – firewall between TIN and secured domain.</td>
</tr>
<tr>
<td>CAMS WebServer3, CAMS WebServer4.</td>
<td>CAMS AppServer1, CAMS AppServer2, CAMS AppServer3.</td>
<td>9070, 9072.</td>
<td>Ports already open as both set of servers are in the TIN domain and have no firewall between them.</td>
</tr>
<tr>
<td>CAMS WebServer1, CAMS WebServer2, CAMS WebServer3, CAMS WebServer4.</td>
<td>SSO Servers.</td>
<td>SSO Web server ports.</td>
<td>External Web servers and SSO Web servers are both in DMZ. No firewall between them. Firewall need to be opened between Internal Web servers and SSO Web servers.</td>
</tr>
</tbody>
</table>
3.1.6.6 Load Balancer Requests

Load balancer’s pool members shown in Table 3.12 need to be configured to distribute load between the Web servers.

Table 3.12 Load Balancer Matrix

<table>
<thead>
<tr>
<th>URL</th>
<th>Pool Members</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAMU-CC.externalapplication.com when accessed in TIN</td>
<td>TAMU-CC.internalapplication.com: 443.</td>
<td>External application when accessed from TIN is redirected to the internal load balancer.</td>
</tr>
</tbody>
</table>

The steps discussed in the implementation phase follow the enterprise security policies. The policies of “Defense in Depth”, “Protected Enclave”, “One point authentication” and “Separation of Duties” were followed in order to arrive at the steps for the groundwork which forms the basis of the platform building.

3.1.7 Securing application platform with SSL

An SSL certificate secures or encrypts secure information during a transaction. In the architecture that was designed in section 3.1.4, there are several points in the platform where data is not encrypted. These points of vulnerabilities were identified in this section.
3.1.7.1  End User URL Protection

When the end users accesses www.TAMU-CC.externalapplication.com or www.TAMU-CC.internalapplication.com, it is important that the site is secure. In order for the users to see an https URL on their browsers, these end user URLs need to be protected using SSL certificates. The platform architecture reveals that the users’ first point of contact in the organization is the Load Balancer. The load balancer in turn talks to the Web servers. In order for the URL to be SSL protected, SSL certificates need to be installed either at the Web server level or the Load Balancer. Terminating SSL at the Load Balancer is advantageous and cost effective as the organization needs to only obtain and maintain one certificate per user.

But SSL termination can be done at the Load Balancer only if the Load Balancer and the Web servers exist in the same domain. If they are in different domains, there is a risk of allowing clear traffic to flow across the network. In the case where the Load Balancer and Web servers reside in different networks, it is important that the end user SSL encryption is only terminated at the Web server level. This means SSL certificates for the Web site need to be installed on every Web server. While it is difficult to maintain multiple certificates for the same domain, it is important that this is done in order to encrypt all end user transactions.

3.1.7.2  SSL between the Web and Application tiers

The Web servers proxy end user requests to the middle tier, where the application servers carry the business logic. If the Web and Application tiers reside in different networks, it is very important that SSL certificates are installed on the application
servers. In this project, SSL certificates were installed on the Weblogic server managed instances which receive proxy requests from the Web tier.

3.1.7.3 SSL for Application Administration

Administrative applications are only accessed by application or server administrators internal to the organization. End users do not require access to these applications. These ports are usually open directly from the application servers to the administrators. For example, Weblogic console is the administrative application used to manage Weblogic domains. Administrators are located in the restricted domain and hence the port is open to everyone in the domain. The two forms of security used to protect this resource are password authentication and SSL certificates. In case of application servers, the SSL certificates cannot be stored in files as in the Web servers. [Zhang 2001b] The certificates need to be imported to a keystore which will be used by the application server to secure all transactions between the Web tier and the application tier or between the browser and the application servers.

The application servers can be configured with one-way as well as two-way SSL. In the case of one way SSL, only the server is expected to present a certificate to the client, which the client accepts in order to begin a secure transaction. In the case of a two-way SSL, both the client and the server are expected to present certificates to each other and accept the certificates in order to begin a secure transaction. In this case, the client is either the Web server or the Web browser.

3.1.7.4 SSL Certificates for the CAMS Web site

SSL certificates for the applications in Table 3.13 need to be generated and installed on the servers.
Table 3.13 Application Certificate Matrix

<table>
<thead>
<tr>
<th>Certificate Name</th>
<th>Installation Location</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>TAMU-CC.internalapplication.com</td>
<td>Load Balancer/Web server.</td>
<td>Secures the internal Web site.</td>
</tr>
</tbody>
</table>

3.2 Building Web servers

Web servers are run on the 32-Linux operating system. Linux is a widely used operating system for servers since it is portable, more stable than other contemporary operating systems and it is a freely available open source operating system.

A Web server is responsible for accepting HTTP (Hypertext Transfer Protocol) request from the clients (Web browsers) and providing them with the HTTP response along with any optional data contents. The HTTP response usually consists of an HTML document but can also be an image or some other type of document. If there is some error in the client’s request, the server should be able to inform the client about the error. The Web server also usually has the capability of logging detailed information about the client requests and the server responses to log files.

3.2.1 Apache

Apache has been the most popular Web server on the Internet since 1996. [Apache 2007] Originally, it was a Unix-based server but now versions for Windows, OS/2 and other platforms exist. As with most open-source projects, there are numerous add-ons and tailored versions of the server available, which are created using the Apache module API (Application Program Interface). It is primarily used to serve both static and dynamic Web pages on the World Wide Web. Installation and configuration instructions are mentioned in Appendix A.
3.2.2 iPlanet

iPlanet Web server is an extremely powerful multi-process, multi-threaded, secure Web server built on open standards that enables business enterprises to seamlessly integrate with other internal and external systems. [iPlanet Web server 2007] It was jointly developed by Sun Microsystems and Netscape Communications.

iPlanet includes e-commerce services, portal services, communication services, Web and application services, integration services, and user management services. E-commerce services provide enterprises with the ability to deploy secure selling, and bill presentation and payment. Portal services allow enterprises to authenticate employees, suppliers, and customers who wish to access applications, content, and data on a customized portal using a browser and a dial-up Internet connection. Communication services enable Internet Service Providers (ISPs), telephone companies, and enterprises to provide messaging, calendar, and e-mail services using a single phone number. Web and application services provide users access to applications, databases, and Web pages. [SearchSOA 2007]

Secure authentication can occur with a valid certificate so that the client can trust the server and then secure communication of sensitive and confidential data can be taken place without being tampered over the network.

In order to validate the server certificate, it is important to check the valid date range of the certificate, the public keys of the certificate. It is also important to check if the certificate is not in the Certificate Revocation List (CRL). Installation and configuration instructions are explained in Appendix A.
3.3 Building Application server

An application server is a software engine that delivers applications to the client computers or devices. An application server handles most of the business logic and data access of the application. The main advantage of an application server is the ease of application development since they are assembled from the building blocks of the application server itself.

3.3.1 Weblogic Server

Weblogic server is an application server used for developing and deploying multitier distributed enterprise applications. It centralizes application services such as Web server functionality, business components, and access to backend enterprise systems.

Weblogic server operates in the middle tier of a multitier architecture. A multitier architecture determines where the software components that make up a computing system are executed in relation to each other and to the hardware, network, and users. [BEA 2007]

Weblogic server is based on Java 2 Platform, Enterprise Edition (J2EE), the standard platform used to create Java-based multi-tier enterprise applications.

The main features of Weblogic server include connectors that make it possible for any legacy application on any client to interoperate with server applications, Enterprise JavaBean (EJB) components, resource pooling, and connection sharing that make applications very scalable. [BEA 2007] The Weblogic feature which supports SSL for the encryption of data transmissions, as well as authentication and authorization
mechanisms, makes applications and transactions secure. Installation and configuration instructions are mentioned in Appendix A.

4. EVALUATION AND RESULTS

4.1 System Testing

The Web servers and the application servers were started and the log entries of the servers were carefully tested to make sure that the servers did not produce errors and came up successfully. The application server log files are checked to see if the certificates successfully loaded on the server.

The log as shown in Figure 4.1 shows that the Application Server Administrative console came up successfully.

Figure 4.1 Administrative Server Log
The Administrative console was accessible after the server came up successfully as shown in Figure 4.2.

![WebLogic Server Administration Console]

**Figure 4.2 Administrative Console**

The Managed Application server started successfully as shown in Figure 4.3.

![Managed Server Log](http://localhost:9170/console/login/LoginForm.jsp)

**Figure 4.3 Managed Server Log**
The Weblogic console showed all the servers including the Admin and Managed instances Up and Running as shown in Figure 4.4.

![Administrative Console for Servers](image)

**Figure 4.4 Administrative Console for Servers**

### 4.2 Functional / User Acceptance Testing

The application hosted on the Web servers and the application servers were accessed through a browser using the https protocol to make sure that the Web pages came up successfully. The Web pages were also tested to see if the valid certificates on the respective servers came up. The application hosted on the application server was also accessible through the Web server that was setup as proxy server for the application server.
5. FUTURE WORK

Securing the Web application platform was done at the network and server level in this project. The architectural diagram as well as the checklist for securing Web application has scope for future work. The enterprise wide Security Architecture designed can be improved to include more details apart from domain structuring and Single Sign-On Architecture.

The security standards can be improved by including application level security parameters that need to be configured. TAMU-CC uses various applications. A extension of this project could drill down into the details of the major applications that TAMU-CC user community uses. For example, on apache one can use the URL redirection feature securing each of them in order to hide the actual application URL from the Internet users.
6. CONCLUSION

As a result of this research, an architectural diagram of the designed network with the indication of the security features was drawn. This diagram is a detailed network architecture that any organization hosting e-commerce Web sites can use in order to implement a secure Web site where customers can make transactions without the fear of their credit cards or personal information being hacked. Part of the architecture containing the Web tier and application tier was constructed and SSL certificates were applied to both the tiers.

After installing the Apache and iPlanet Web servers, the servers were restarted and the logs entries were checked to see if the servers started successfully without any error messages. The servers were brought up successfully, the application hosted on the Web servers was accessible on a browser. Once the Web page was up, a small lock appeared on the right hand corner of the Web page. Clicking on the lock brought up the certificate that was installed on the Web server and indicated if the certificate was valid.

Similarly, after installing the application servers, the servers were started and the logs were checked to see if the servers came up fine. Accessing the application hosted on the application servers now brought up secure Web pages using HTTPS protocol.

The Web servers were also setup as proxy servers for the application servers to add an extra layer of security for the applications hosted on the application servers and the application will be accessed via the Web servers using a proxy plug-in for the application server. During the oral presentation of the project, it was suggested that this project could be used for teaching the “Information Assurance” course.
BIBLIOGRAPHY AND REFERENCES


I. Procedure to install and configure Apache

1. Download Apache for Linux from http://httpd.apache.org/download.cgi

2. Uncompress and untar the downloaded file by using the following commands:
   -d httpd-2_0_NN.tar.gz
   tar xvf httpd-2_0_NN.tar
   httpd-2_10 folder is created.

3. Go to the httpd-2_10 folder and configure Apache by running the command ./configure

4. Go to the build folder and and run the following commands to install Apache:
   install
   make install
II. Procedure to install and configure Weblogic

1. Download the Weblogic 8.1 binary from the following Oracle Web site:
   http://commerce.bea.com/showallversions.jsp?family=WLS

2. Copy the installation file to the desired location.
   (In this example, the location is /opt/beasys/wlserver81/)

3. Go to /opt/beasys/wlserver81/common/bin and run the config file in console mode.

   java -version .:/config.sh -mode=oracle

   <-------------------------------------------------------------------------- BEA Weblogic Configuration Wizard -------------------------------------------------------------------------->
   Create or Extend a Configuration:
   -1] Create a new Weblogic configuration
   | Start here to create a Weblogic configuration in your projects directory.
   2] Extend an existing Weblogic configuration
   | Start here to extend an existing Weblogic configuration. Use this option to add applications and services, including Database server (JDBC) and Messaging (JMS). This option also enables you to extend functionality by enabling Weblogic Workshop.

   Enter index number to return OR [Exit][Previous][Next]> 1

4. Install in non-express mode.

   Enter index number to select OR [Exit][Previous][Next]> 1

   <-------------------------------------------------------------------------- BEA Weblogic Configuration Wizard -------------------------------------------------------------------------->
   Choose Configuration Option:
   ---------------------------------------------

   *Do you want to run the wizard in express mode?
   ->] Yes
   2] No

   Enter index number to return OR [Exit][Previous][Next]> 2
5. Configure the Admin server.

Enter value for "SSL listen port" OR [Exit][Previous][Next] > 9670

Configure the Administration Server:

Enter administration server configurations. Each WebLogic Server domain must have one Administration Server. The Administration Server hosts the Administration Console which is used to perform administrative tasks.

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name:</td>
<td>CMAX_ADMIN</td>
</tr>
<tr>
<td>Listen address:</td>
<td>localhost</td>
</tr>
<tr>
<td>Listen port:</td>
<td>9670</td>
</tr>
<tr>
<td>SSL listen port:</td>
<td>9670</td>
</tr>
<tr>
<td>SSL enabled:</td>
<td>true</td>
</tr>
</tbody>
</table>

Select Option:
1. Modify "Name"
2. Modify "Listen address"
3. Modify "Listen port"
4. Modify "SSL listen port"
5. Modify "SSL enabled"
6. Discard Changes

Enter option number to select OR [Exit][Previous][Next] >

6. Configure the Managed servers.

Modify "SSL listen port" OR [Exit][Previous][Next] > 8072

Configure Managed Servers:

Add or delete configuration information for managed servers. A typical production environment has one or more managed servers. Each managed server is an instance of WebLogic Server used to host enterprise applications.

<table>
<thead>
<tr>
<th>Name*</th>
<th>Listen address</th>
<th>Listen port</th>
<th>SSL listen port</th>
<th>SSL enabled</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMAX_MS1</td>
<td>localhost</td>
<td>9070</td>
<td>8070</td>
<td>true</td>
</tr>
<tr>
<td>CMAX_MS2</td>
<td>localhost</td>
<td>9072</td>
<td>8072</td>
<td>true</td>
</tr>
</tbody>
</table>

Select Option:
1. Modify "Name"
2. Modify "Listen address"
3. Modify "Listen port"
4. Modify "SSL listen port"
5. Modify "SSL enabled"
6. Done

Enter option number to select OR [Exit][Previous][Next] >
7. Create Cluster.

Modify "Cluster address" OR [Exit][Previous][Next] > localhost:9070, localhost:9072

Configure Clusters:

A cluster contains multiple WebLogic Server instances (servers) that run simultaneously and work together to provide increased scalability and reliability. A cluster appears to be a single WebLogic Server instance to clients.

<table>
<thead>
<tr>
<th>Name*</th>
<th>Multicast address</th>
<th>Multicast port</th>
<th>Cluster address</th>
<th>Frontend host</th>
<th>Frontend HTTP port</th>
</tr>
</thead>
<tbody>
<tr>
<td>CASE_CL</td>
<td>257.0.0.1</td>
<td>7001</td>
<td>localhost:9070, localhost:9072</td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>

Select Option:
1 - Modify "Name"
2 - Modify "Multicast address"
3 - Modify "Multicast port"
4 - Modify "Cluster address"
5 - Modify "Frontend host"
6 - Modify "Frontend HTTP port"
7 - Done

Enter option number to select OR [Exit][Previous][Next] >

8. Assign Managed servers to the Cluster.

Assign managed servers to a cluster in the domain.

Cluster
| CASE_CL [1] |

Enter number exactly as it appears in brackets to toggle selection OR [Exit][Previous][Next] > 1

Assign Servers to Clusters:

Assign managed servers to a cluster in the domain.

*Select WebLogic Servers and assign them to a cluster. CASE_CL
1|CASE_M1
2|CASE_M2

Select Option:
1 - Select
2 - Select All

Enter option number to select OR [Exit][Discard][Accept] >
9. Enter the target domain directory location.

```
Select the target domain directory for this configuration:
-----------------------------------------------

"Target Location" = [/opt/beasys/user_projects/domains]
```

10. Enter the domain name.

```
Edit Domain Information:

<table>
<thead>
<tr>
<th>Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;Name&quot;</td>
<td>mydomain</td>
</tr>
</tbody>
</table>
```

Enter value for "Name" OR [Exit][Previous][Next] > censdomain

11. Domain creation is completed.

```
Creating Domain...
0% 25% 50% 75% 100%
[-------------------------][-------------------------]
[-------------------------][-------------------------]

**** Domain Created Successfully! ****
```
12. To start the Weblogic admin console, go to the installation domain directory and run the startWeblogic.sh &

```
drwxr-xr-x  3 beasys  beasys  4096 Nov 24 01:35 applications
drwxr-xr-x  5 beasys  beasys  4096 Nov 24 01:35 CAMS_ADMIN
-rw-r--r--  1 beasys  beasys  4613 Nov 24 01:35 CAMS_Domain.log
drwxr-xr-x  2 beasys  beasys  4096 Nov 24 01:34 _cfgwiz_donotdelete
-rw-r--r--  1 beasys  beasys  5047 Nov 24 01:35 config.xml
-rw-r--r--  1 beasys  beasys  5047 Nov 24 01:35 config.xml.booted
-rw-r--r--  1 beasys  beasys  1436 Nov 24 01:35 config.xml.original
-rw-r--r--  1 beasys  beasys  2263 Nov 24 01:34 DefaultAuthenticatorInit.ldift
-rw-r--r--  1 beasys  beasys  3380 Nov 24 01:34 DefaultAuthorizerInit.ldift
-rw-r--r--  1 beasys  beasys  1799 Nov 24 01:34 DefaultRoleMapperInit.ldift
-rw-r--r--  1 beasys  beasys  462 Nov 24 01:34 fileRealm.properties
drwxr-xr-x  2 beasys  beasys  4096 Nov 24 01:35 rmfilestore
-rw-r--r--  1 beasys  beasys  38 Nov 24 01:34 SerializedSystemIni.dat
-rw-r--r--  1 beasys  beasys  1167 Nov 24 01:33 setEnv.sh
-rw-r--r--  1 beasys  beasys  627 Nov 24 01:33 startManagedWebLogic_readme.txt
-rw-r--r--  1 beasys  beasys  4438 Nov 24 01:34 startManagedWebLogic.sh
-rw-r--r--  1 beasys  beasys  2874 Nov 24 01:34 stopWebLogic.sh
-rw-r--r--  1 beasys  beasys  1774 Nov 24 01:34 stopWebLogic.sh
```

13. If the admin server starts successfully, the server log has the following entry.

```
13. To access the Admin Console, type the URL:
http://adminurl:port/console
```

![WebLogic Server Administration Console]

**WebLogic Server Administration Console**

Sign in to work with the WebLogic Server domain **camsdomain**

**Username:** weblogic

**Password:**

[Sign In]
14. To start the Weblogic Managed Server(s), go to the installation domain directory and run the startManagedWeblogic.sh
/startManagedWeblogic.sh CAMS_MS http://localhost:9170 &

15. If the managed server 1 starts successfully, the server log has the following entry.

16. If the managed server 2 starts successfully, the server log has the following entry.

17. All the servers are up and running now.

III. Procedure to install and configure IPlanet

1. Download the IPlanet binary from the following link:
http://www.iplanet.com/downloads/download

2. Copy the installation file to the desired location.
(In this example, the location is /opt/netscape/iplanet06sp8)
3. Unzip and extract the downloaded file.

4. Run the setup file.
   
   `./setup`

5. Follow the onscreen instruction and press Enter when the installation is complete.
APPENDIX B

I. SSL Cert Generation and Configuration using Keytool

1. Set Java PATH

(eg) export PATH=/opt/beasys/j2sdk1.4.2_13_bin:$PATH

2. Generate Private Key


Note: Make sure CN=the domain name (eg) TAMU-CC.capitalone.com

3. Generate CSR (Certificate Signing Request)

keytool -certreq -v -alias TAMU-CC -file TAMU-CC_certreq.pem -keypass ****** -storepass ****** -keystore TAMU-CC.jks

4. Obtain the public cert and the CA cert

Entrust

- In case of entrust go to the entrust website http://www.entrust.net/
- Click on Web server.
- Enter the Reference Number, Authorization Code and the cert request and click on Submit Request.
- The PublicKey is displayed.
- Copy the contents to a cert request file on the server.
- Click on display and copy the CA cert to a file on the server.

Verisign

- Go to http://www.verisign.com/ssl/
- Enter all the required details including the cert request
- Make sure you choose the number of licenses needed. Each IP address that you need to install the cert on needs one license.
- Click on Accept.
- Verisign will send you the public cert and the intermediate cert in an email. Place them in the appropriate files on the server.
5. Import CA cert

```bash
keytool -import -v -trustcacerts -alias entrust_cacert -file cacert.pem -keystore TAMU-CC.jks -storepass ******
```

6. Import Public Key

```bash
keytool -import -v -alias TAMU-CC -file TAMU-CC_public.pem -keypass Weblogic -keystore TAMU-CC.jks -storepass ******
```

7. Configure Weblogic SSL

- Click on the managed server where you want to configure SSL and go to the keystores & SSL Tab.
- Fill in the following details and restart the managed server.

![Configure Keystores](image_url)

- **Custom Identity Key Store File Name**: enter the keystore
- **Custom Identity Key Store Type**: jks
- **Custom Identity Key Store Pass Phrase**: ******
- **Confirm Custom Identity Key Store Pass Phrase**: ******

The fully qualified path to the identity keystore.

The type of the keystore. Generally, this is JKS.

The password defined when creating the keystore. This field is optional or required depending on the property in order to write to the keystore. However, some keystores do not require the passphrase to read from a keystore so whether or not you define this property depends on the requirements of the keystore.
8. Configure SSL on Apache

a. SSL was configured by creating the following entries in the httpd.conf file.

```
<VirtualHost _default_:8443>
  SSLEngine On
  SSLCACertificateFile /opt/apache/ssl/ca.crt
  SSLCertificateFile /opt/apache/ssl/demo.crt
  SSLCertificateKeyFile /opt/apache/ssl/demo.key
  SetEnvIf User-Agent ".*MSIE.*" nokeepalive ssl-unclean-shutdown
  DocumentRoot /opt/apache/content
  ServerName localhost:9070
  ErrorLog logs/ssl_error_log
  TransferLog logs/ssl_access_log
  CustomLog logs/ssl_request_log \
    "%t %h %{SSL_PROTOCOL}x %{SSL_CIPHER}x %r" %b
</VirtualHost>
```
b. Proxy plugin was configured by creating the following entries in the httpd.conf file.

```
<IfModule mod_weblogic.c>
   WebLogicCluster localhost:9070,localhost:9072
   KeepAliveEnabled ON
   DynamicServerList OFF
</IfModule>
<Location /tammuapp/>
   SetHandler weblogic-handler
</Location>
```

II. SSL Cert Generation and Configuration using OpenSSL

To generate and configure SSL using OpenSSL, run the following commands:

1. bash-3.00$ openssl genrsa -out demo.key 1024
   Generating RSA private key, 1024 bit long modulus
   ........................+++
   ........................+++
   unable to write 'random state'
e is 65537 (0x10001)

2. bash-3.00$ ls -l
   total 8
   -rw-r--r--  1 user group 887 date demo.key

3. bash-3.00$ openssl req -new -key demo.key -out demo.csr
   You are about to be asked to enter information that will be incorporated into your certificate request.
   What you are about to enter is what is called a Distinguished Name or a DN.
   There are quite a few fields but you can leave some blank
   For some fields there will be a default value,
   If you enter '.', the field will be left blank.
   ----- Country Name (2 letter code) [GB]:US
   State or Province Name (full name) [Berkshire]:TX
   Locality Name (eg, city) [Newbury]:Corpus Christi
   Organization Name (eg, company) [My Company Ltd]:TAMU-CC
   Organizational Unit Name (eg, section) []:CAMS
   Common Name (eg, your name or your server's hostname) []:5467589987
   Email Address []:

   Please enter the following 'extra' attributes to be sent with your certificate request
   A challenge password []:
   An optional company name []:
4. bash-3.00$ ls -ltr
   total 12
   -rw-r--r--  1 user group 887 Jul 20 17:20 demo.key
   -rw-r--r--  1 user group 660 Jul 20 17:41 demo.csr

http://entrusta/cda-cgi/clientcgi?action=start