Secure AD authentication made EASI:

Using a Java based Kerberos module to securely authenticate users against Active Directory.

Barton E. Jokinen
GSEC Version 1.4b, option 2

Abstract:

This paper details how we implemented a solution allowing third party applications running on Windows, UNIX or Linux to authenticate users securely against Active Directory using a Java based module that performs a Kerberos Authentication Session. The paper begins with a brief description of the technologies that were involved in the project. It then discusses the business and technology requirements. The reader is then taken through the testing and implementation process. Finally it discusses the benefits that we obtained from a more secure computing environment and what steps we plan to take forward to enhance the capabilities of this product.

My goal is to provide a possible solution to assist any other organization that faces a situation similar to ours and allow them to successfully solve their authentication issues.

Introduction:

With the introduction of Active Directory (AD) Microsoft provided a powerful tool for managing resources in your windows environment. AD can be used for authentication of users, a storage area for information about users and other objects that can be accessed through LDAP calls.

Any new technology that a company deploys offers many benefits as well as many challenges. The challenges that we faced with the introduction of Active Directory as a central source for authentication were:

- Secure authentication of non-ADSI third party and Unix/Linux based applications against AD
- Fiscal constraints
- Time constraints- limited window for development and deployment

Setting the stage:

One of our major goals for deploying Active Directory was to move more towards a central source for authentication. Like many companies we had several Windows NT 4.0 domains. Our NT 4.0 Domains were primarily based on geography and were consolidated through a parallel migration to one Windows 2000 domain (with an empty root domain).

Once all the users had migrated to the new Windows 2000 domain the Active Directory Team, began receiving queries on the best way to authenticate users to Active Directory from third party applications. Most of these requests came from internal application developers and project managers of applications that traditionally relied on their own internal or external security database for authentication of users to the application.

Some of these applications are capable of using LDAP (v2 and v3) for authentication. Lightweight Directory Access Protocol (v3) as outlined in RFC 2251 supports Simple Authentication and Security Layer (SASL). SASL is defined in RFC 2222 and provides security during the authentication phase. LDAP (v2) does not have this security ability and would pass user credentials in plain text across the network. The LDAP clients provided by the Independent Software Vendors with most of these applications did not use the a SASL method that was recognized by Active Directory and thus reverted to LDAP Version 2 and passed user credentials in clear text.

Clear text passwords being passed through the network was determined to be an unacceptable security risk by the Active Directory Security Representative (myself). An alternate means of authentication was needed if our new Active Directory was to function as the central authentication repository.

We began by investigating the authentication mechanisms supported by default in Active Directory. We did not want to add authentication mechanisms to our AD structure if it was not necessary. The rootDSE in Active Directory specifies what SASL mechanisms are supported and by default includes GSSAPI and GSS-SPNEGO. GSSAPI is used for Kerberos authentication and Kerberos and NTLM authentication use GSS-SPNEGO. Unfortunately most versions of the LDAP (v3) client do not support these SASL mechanisms by default.

Another issue that we discovered early on was that, by default, Active Directory creates an X.500 standard “cn” parameter as “firstname lastname” rather than the user id that is used to login into the domain (the “sAMAccountName” attribute in the Active Directory). This means that even if the LDAP client used by the application could pass the user’s credentials in a secure fashion across the network, the user name would not be recognized by Active Directory as a valid user unless the “cn” parameter was changed to be the sAMAccountName.

Further research indicated that creating a Kerberos module similar to a Pluggable Authentication Module like those used in a UNIX/ Linux environment might provide an inexpensive and quickly deployable way to securely authenticate users against Active Directory.

What exactly is Kerberos and why did it make such a good choice for authentication? Kerberos is a network authentication protocol that came out of MIT’s Athena Project. It provides strong authentication for client server applications by using secret key cryptography. Kerberos is imbedded in all Windows 2000 and XP operating systems and is easily added to Unix and Linux operating systems if they do not already have a client installed. It is not the intent of this paper to explain Kerberos in depth. For further information I highly recommend reading Kerberos: Secure Authentication by Jose L. Marquez on the SANS Reading Room (http://www.sans.org/rr/win2000/kerberos2.php) if you want a very well laid out explanation of the inner workings of Kerberos.

Planning:

A workgroup was formed consisting of representatives from Application Development, Project Management, the Active Directory Team and Enterprise Security Team. My position on this team was unique as I am a member of both the Security and Active Directory Team so I was able to provide input on how Microsoft Windows 2000 uses Kerberos and also articulate the requirements of the Enterprise Security Team. The information below outlines the requirements and initial plans that came out of the meetings of this workgroup.

Requirements for a reusable Security Module:

Application group security requirements:

- Short-term
  - Authenticate against the Active Directory
  - Be able to query the user group memberships
  - Need to support multiple OS – Windows, Linux, Solaris
- Long-term
  - Be able to use the authenticated credentials to access the domain resources
  - Provide single sign-on between applications

Security requirements from Enterprise Security Team:

- Use Windows 2000 Active Directory for authentication and authorization.
- Make use of the MIT Kerberos v5. Protocol for communication to the Active Directory
  - Krb5 is natively supported by W2k
  - Krb5 enables single sign-on with the use of ticket (TGT)
- Use of simple LDAP v2 **not allowed** as it sends the passwords in clear-text over the wire
- Use of LDAP v3 permissible with use of SASL/GSSAPI calls that in turn use the Kerberos underneath.
- Use of LDAP over SSL to be evaluated for overhead on Domain Controllers.

Issues with meeting security requirements:

- Active Directory, by default, creates the X.500 standard "cn" parameter as "firstname lastname" rather than the user id that is used to login into the domain ("sAMAccountName" attribute in the Active Directory). This cn parameter must be changed on user by user basis to match the sAMAccountName in order to make it work for LDAP authentication

- Many applications do not support LDAP v3, even if they support LDAP v3 – do not support SASL/GSSAPI
- Many applications do not support MIT Kerberos authentication or other secure authentication mechanisms to Active Directory
- Usually applications support simple LDAP or at the most LDAP over SSL
- Simple LDAP sends clear text passwords, that can be sniffed over by any freely downloadable network sniffing program

Initial plan of attack:

Using the requirements from all the involved parties we agreed on the following. The items below gave us a starting point.

- Development of a reusable component for security that can talk to the Active Directory using MIT Kerberos v5 protocol securely in order to authenticate and also to secure the LDAP calls to the directory
- Make use of the Java Authentication and Authorization Service (JAAS) / Java API to develop this module
- Acquire connectivity to this module from your application /application server
- Use Kerberos client credentials (TGT) to further communicate LDAP to the Active Directory using SASL/GSSAPI

Figure 1 below illustrates the initial high-level design for the Reusable Security Component. We already had a good idea of the primary applications that would be able to have immediate use from this component.

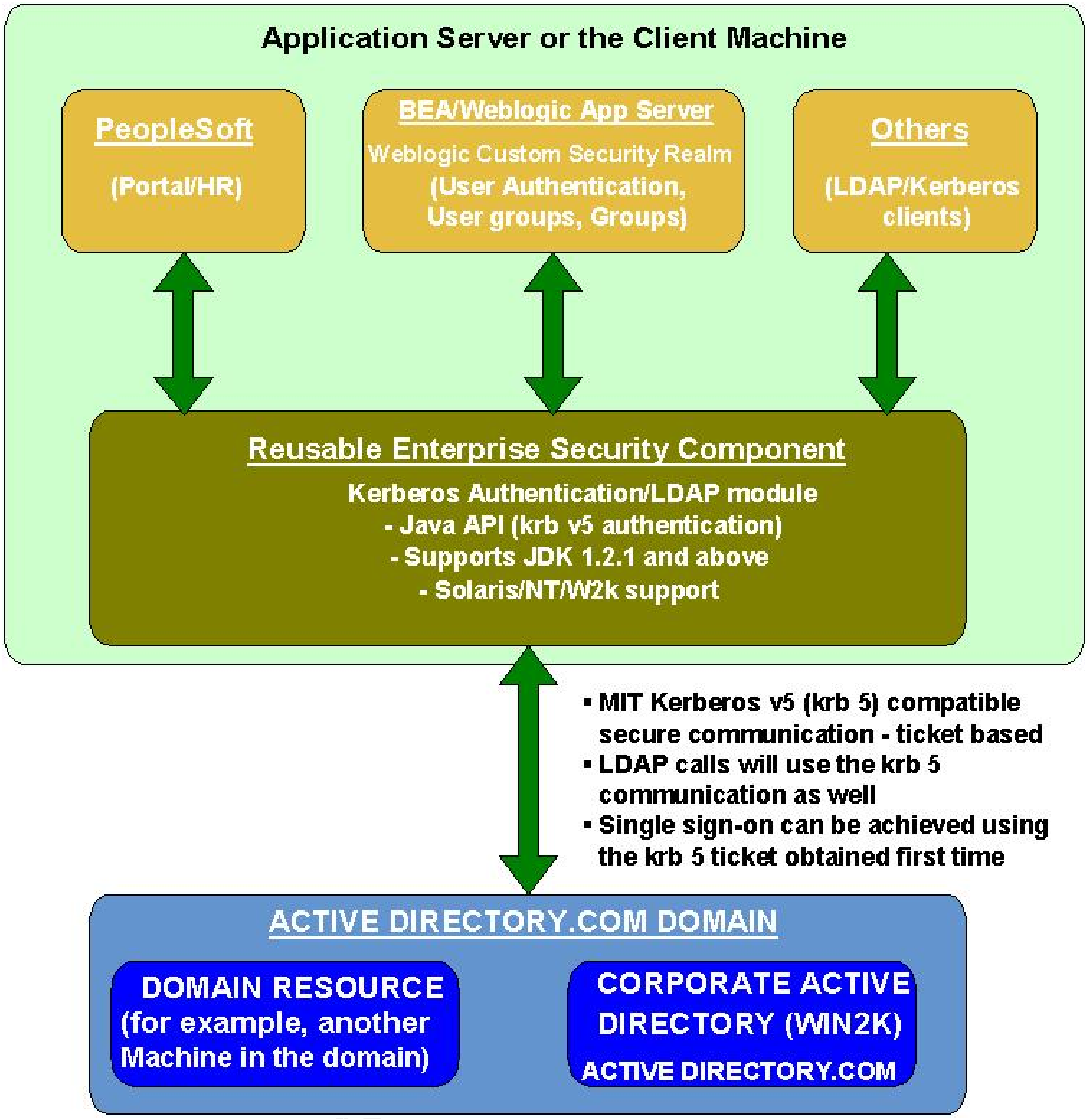
Application Server or the Client Machine
PeopleSoft
(Portal/HR)
BEA/Weblogic App Server
Weblogic Custom Security Realm
(User Authentication,
User groups, Groups)
Others
(LDAP/Kerberos
clients)
Reusable Enterprise Security Component
Kerberos Authentication/LDAP module
- Java API (krb v5 authentication)
- Supports JDK 1.2.1 and above
- Solaris/NT/W2k support
▪ MIT Kerberos v5 (krb 5) compatible secure communication - ticket based
▪ LDAP calls will use the krb 5 communication as well
▪ Single sign-on can be achieved using the krb 5 ticket obtained first time
ACTIVE DIRECTORY.COM DOMAIN
DOMAIN RESOURCE
(for example, another
Machine in the domain)
CORPORATE ACTIVE
DIRECTORY (WIN2K)
ACTIVE DIRECTORY.COM

Figure 1 Overview of the Reusable Security Component Architecture

In Figure 2 the Reusable Enterprise Security Component is broken down in more detail. It shows the low level components involved in the reusable security component. This was our first attempt at the architecture and appeared to meet our requirements while allowing for future enhancements to services supplied by the module.

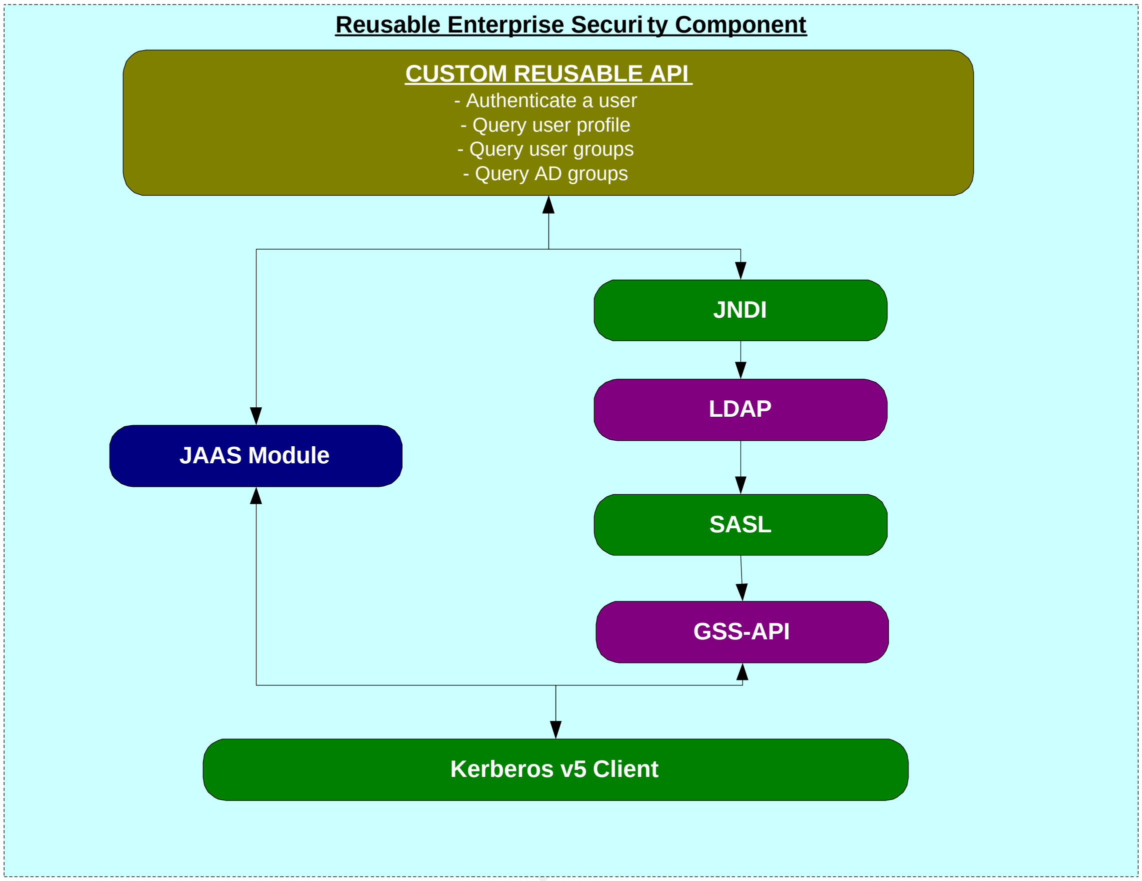

Figure 2 Detailed Reusable Enterprise Security Component Architecture

We then evaluated the pros and cons of our initial plan of attack to see if it met the requirements and goals from our initial meetings. We also viewed the solution from a variety of perspectives to ensure we did not overlook anything.

Pros of initial plan

- Provides a reusable component that can be used by other enterprise applications.
- Able to use this on the Solaris/Win2k/NT – Java API based
- Conforms to the security standards for single sign-on using Kerberos, a rising de facto standard
- Backward compatible up to JDK 1.2.1 for applications running on older versions
- Extends the functionality of applications/server do not currently support corporate security standards in effect allowing the vendors to "catch up".

Cons of initial plan

- Development and maintenance of the module – predictably not much once it is properly tested and working

At this point it was unclear if the Reusable Security Component would be a long-term solution or a quick fix until an “off the shelf” application could be evaluated, purchased and deployed. With this in mind we decided that we would design it so that future functionality could be added with minimal cost and development time.

The project was dubbed Enterprise Application Security Integration or EASI. The next step was the development of some basic java code that could be our working model. This development took place in a laboratory environment until it was deemed ready for a production test.

Implementation:

We settled on our new PeopleSoft Portal application as the test bed for the new EASI application. This high visibility application did not support any secure means to authenticate users against Active Directory and therefore was an ideal candidate to test the capabilities of EASI.

PeopleSoft Portal Details

- Portal runs on WLS 6.1 SP1 (JDK 1.3.1)
    - It is just another J2EE (Servlet based) application that runs on a standard J2EE application server
    - Portal provides the capability to use the external authentication
- PeopleSoft portal maintains its own users and authenticates against its own user database
- PeopleSoft portal does not leverage the active directory for authentication, hence, can not provide single sign-on with user desktop/workstation

Desired End State for EASI/PS Portal Implementation

- Single sign-on with the user desktop on internal network
    - Be able to use the network authentication to active directory if the user is already logged into the domain
- Ability to authenticate securely against AD using username and password
- Ability to pull the user profile from the active directory using secure LDAP v3. compliant calls
    - Ability to pull profile with NT or Windows 2000 user id vs. LDAP compliant cn parameter

Proposed Architecture for EASI/PS Portal Implementation

- Use IIS web server 5.0 as the front-end web server layer for the portal servers
    - Leverage the “Integrated Security” which is based on NTLM and standard Kerberos v5

- Leverage the Enterprise Application Security Integration (EASI) to communicate with the AD to pull the user profile
    - Leverage the standard LDAP v3. compliant communication using GSSAPI/kerberos v5
- Configure PS portal security to use external authentication

Figures 3 and 4 below illustrate both the Physical and Logical representations of the proposed usage for EASI with the PeopleSoft Portal system.

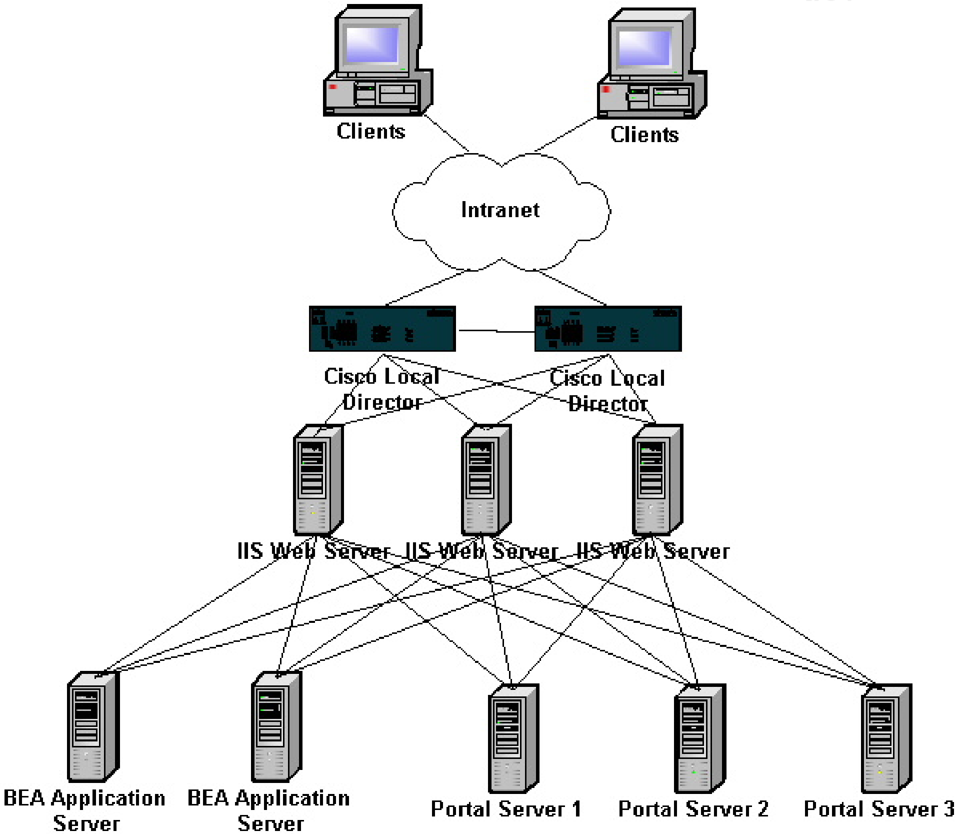

Figure 3 Proposed PeopleSoft EASI Physical Architecture

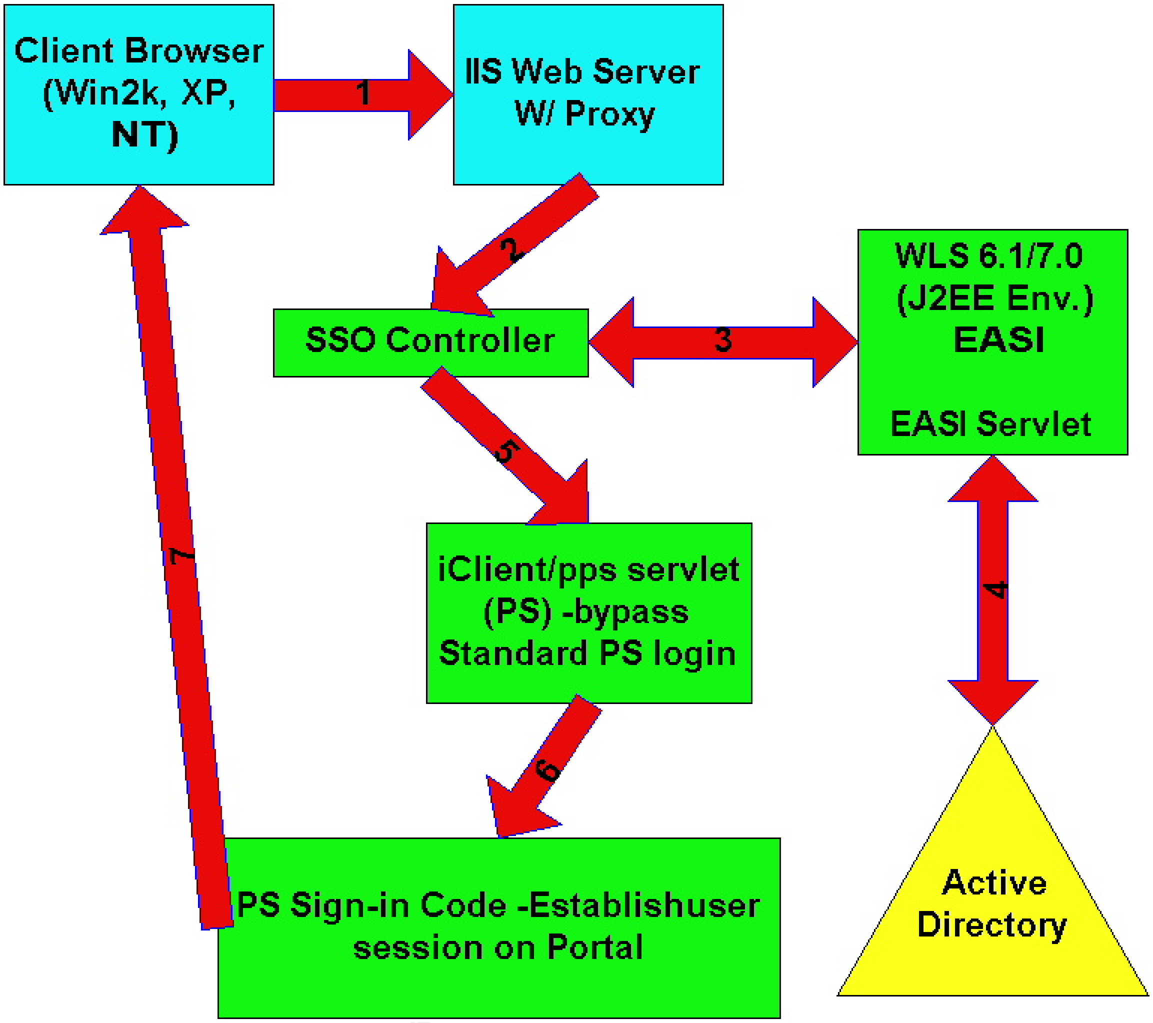
Client Browser (Win2k, XP, NT)
1
IIS Web Server W/ Proxy
2
SSO Controller
3
WLS 6.1/7.0 (J2EE Env.) EASI
EASI Servlet
4
5
iClient/pps servlet (PS) -bypass Standard PS login
6
7
PS Sign-in Code -Establishuser session on Portal
Active Directory

Figure 4 Proposed PeopleSoft EASI Logical Architecture

Required Resources for EASI/PS Portal Implementation

- Timeline
    - Need an integration Proof of Concept before a solid timeline can be established
    - More time will be required in the integration test – communication, access to PS portal environment, etc.
- Resources
    - Web Architecture Team can provide necessary expertise and resource for EASI, SSO controller development, setup of IIS, etc.
    - PeopleSoft team must provide necessary resource and expertise to configure the bypass portal security, PeopleSoft sign-in code, etc.
    - Level of effort for development will be low since skilled resources are doing the development

Support Concept for EASI/PS Portal Implementation

- EASI components

    - Support is already in place via the web architecture team
- SSO Controller
    - Very little or no support required, since very little custom code in a servlet
    - No impact to this implementation expected as a result of the upgrades to the application server, portal application, etc.
- IIS Server
    - Standard support model already in place

Test Plan for EASI/PS Portal Implementation

- Perform an integration test with all the components in place to test the hand-off between EASI and portal
- Make a go/no go decision

We set up a thorough testing plan that was executed in our Active Directory Lab (a scaled down version of our production environment). We brought in several applications that were candidates for the EASI application so that we could test the EASI with them during the final developmental stages. The testing plan consisted of the following:

- Code problems/ Functional: The code was tested as a white box with all conditions and features being tested.
- JAVA Developers Kit (JDK) Compatibility: The code was tested for compatibility for different JDKs. The lowest denominator being JDK 1.2.2.
- Load testing: Load testing was performed on the code to determine if the components could survive the expected load and also to find the breaking point for the application.
- Network Security: A packet analyzer was used to determine if the components were compromising security by passing user IDs or passwords as clear text.

Once the EASI module passed the rigors of the lab we implemented it in our production environment on the new PeopleSoft Portal installation and moved a select number of test users to the system. We ran the test users in the new environment while monitoring the EASI module with debug commands. These debug commands where developed and placed into the EASI module during the lab testing phase and it was decided to leave them in the application for troubleshooting purposes as they did not add excessive overhead.

The data that was collected during the testing and implementation process showed that making EASI a shared centralized module would be advantageous. The first advantage being a reduction in the number of applications that we had to maintain and troubleshoot, the second was the ability to control who has access to the EASI module and what they would be using it for.

After:

By implementing the EASI module in our environment we were able to securely integrate software developed in-house and software purchased off the shelf to securely connect to a central point of authentication (Active Directory) and do so with minimum cost and programming effort. We are now able to control what applications are using Active Directory to authenticate users. Not allowing clear-text passwords to be passed across the network ensures that our computing environment is more secure.

The EASI application is just the first step in the company's effort to transform from using multiple security databases to one central location of user names and passwords. The next step is promoting the EASI application throughout the company and setting it as the standard for third party authentication to Active Directory. We will also begin training the help desk and the software developers in how to troubleshoot and support the EASI application.

Although I cannot publish the proprietary JAVA code for the EASI application my hope is that by reading this paper it should be simple to create similar code while avoiding most of the "land-mines" that we encountered along the way.

The process outlined in this paper allowed our company to take a security vulnerability and turn it into an asset. It also provided the company a way to identify applications with poor security practices and reform them into secure, functional applications in a cost effective manner. This proves that taking the time to make the enterprise secure can also have direct business benefits.

References

1. De Clerq, Jan. “Windows 2000 Authentication” URL: http://ww.windowsitlibrary.com/Content/617/06/1.html

2. Microsoft. “Windows 2000 Kerberos Interoperability” URL: http://java.sun.com/security/jaas/doc/pam.html

3. Cole, Eric. “GSEC, Security Essentials Toolkit”. ISBN:0789727749

4. Samar, Vipin. “Making Login Services Independent of Authentication Technologies” URL: http://java.sun.com/security/jaas/doc/pam.html

5. Skoudis, Ed. “Counter Hack, A Step-by-Step Guide to Computer Attacks and Effective Defenses”. ISBN: 0130332739

6. Marquez, Jose. “Kerberos: Secure Authentication”. URL: http://www.sans.org/rr/win2000/kerberos2.php

7. Allen, Julia. “The CERT Guide to System and Network Security Practices”. ISBN: 020173723X