Global Information Assurance Certification Paper

Web Single Sign-On Meets Business Reality
Tim Mather
February 2002

Summary/Abstract

Single Sign-On (SSO) will not provide SSO across all applications. Even scaling SSO to just Web servers is a challenge. While vendors will tell you how easy Web SSO is, and how quickly it can be deployed, the reality is quite different. This paper discusses some of the real-world operational challenges in getting a Web-only SSO deployed, starting with the impetus for why to deploy SSO; some considerations in vendor selection; operational considerations in a deployment, including challenges with having SSO and load balancing work effectively together, and; some compensating security controls are discussed.

Background and Goals

This paper presents information on one company's single sign-on (SSO) implementation for Web applications that attempts to meet the goals discussed below. While SSO has been a "holy grail" for sometime, and vendors have repeatedly marketed SSO as being able to handle legacy applications as well as Web applications, the reality has fallen short of that. SSO for Web applications is difficult enough; it is difficult trying to image realistically implementing SSO for legacy applications as well. It is also interesting to note that SSO for both Web and legacy applications is again being marketed, recently under the new marketing term of "enterprise application management" or EAM. (For example, see the article "EAM Ain't Easy" in the January 2002 issue of Information Security magazine.)

With the widespread adoption of customer relationship management (CRM) software (e.g., PeopleSoft, SAP, or Siebel) by many enterprises, implementation goals often are:

- distribute customer information across the enterprise, securely with data consistency and a unified view;
- allow the customer to view their own data externally, and, on a limited basis, input or update information;
- provide an easy to use implementation, whereby internal employees and external customers/partners only need to login once to allow access to data for which they are authorized;
- provide an easy to use implementation that securely authenticates internal employees and external customers/partners, and at least enforces minimum security requirements for usernames and passwords, and preferably uses stronger authentication, and;
- provide a centralized account management system, both for internal employees and external customers/partners.

The above are relevant business unit goals, and important for information security. However, there are additional security considerations as well. For example,

- What is the product's ability to handle SSO for applications other than Web-based (i.e., non-HyperText Transport Protocol, HTTP) – since the business will inevitably want to extend the application?
- How is authentication of users handled?
    - Basic authentication (i.e., username + password), or
    - Strong authentication (e.g., X.509v3 certificates)?
    - Is cookie-based SSO acceptable, or does confidentiality necessitate proxy-based SSO?
- How are systems or SSO components authenticated to other systems (e.g., SSO server authentication of directory server)?
- How is authorization handled (e.g., by an SSO component or the server to which access has been granted based on a valid authentication)?
- Is the SSO solution for a single domain (e.g., xyz.com only), or is there a requirement for cross-domain SSO (linking to abc.com from within xyz.com)?[1]
    - If there is a requirement for cross-domain SSO, how is authentication handled?
    - If there is a requirement for cross-domain SSO, how is authorization handled?
- What auditing services are available?
    - From a security-perspective, access activities (authentication, authorization, and policy changes) are important to audit.
    - From a business perspective, user data (when did user enter site, what pages did user view, how long did user visit site for) is important to audit for marketing reasons.
- How is the product securely administered (e.g., Web-based interface using HTTPS)?

Meeting all of the above goals is not only a technical implementation challenge, but also presents security challenges.

Vendor Selection

Our company's CRM software had already been selected by a cross-functional business group team to be Siebel, including its eService module. eService provides a Web portal for external customers/partners to view, and, on a limited basis, input or update information. With that decision made, the number of vendors to consider for Web SSO was then limited to those which are compatible with Siebel. Fortunately, or unfortunately, that list of Web SSO vendors which are compatible with Siebel is currently quite short: Netegrity's SiteMinder product, and Oblix's NetPoint product[2]. At this point, the Information Security group went from being interested observers to active participants in the vendor selection process.

In preparation for this selection process, product and other information (e.g., white papers) was

[1] Currently, cross-domain SSO is a problem. However, a XML-based protocol, SAML, is being developed to address this issue.

[2] See, for example, Oblix press release "Oblix Becomes First Single Sign-On Provider To Receive Siebel Validation;" dated June 5, 2001; URL: http://www.oblix.com/news/releases/2001/060501.html.

obtained from each vendor's Web site. Based on this initial review of product information, each vendor was then asked to provide further, more detailed product information not generally available to the public through the Web. Obtaining this further information involved execution of non-disclosure agreements (NDAs) between prospective customer and vendor. From this information then, there were several security evaluation criteria that the Information Security group required detailed responses from the vendors about their products capabilities.

Much of this product security evaluation focused on the products' use and implementation of encryption, both symmetric and asymmetric. This was a key evaluation criteria, and our experience showed why. (Good reference information on encryption can be found in RSA Security's Cryptography FAQ[3], and by consulting Bruce Schneier's excellent book on cryptography, "Applied Cryptography," which can be purchased on-line at Amazon.com.) SSO is first and foremost marketed as a security solution, with additional benefits (e.g., easier, more efficient account administration) for businesses. However, it is important to ensure that marketing personnel are not too far out ahead of what developers/engineers have actually been able to accomplish. To our chagrin, we found this to be a problem with both vendors.

With Netegrity's SiteMinder product, we were somewhat dismayed at how the vendor accomplishes authentication amongst product components. SiteMinder's policy server, policy store, and Web agents authenticate each other through use of a shared secret. When asked why SiteMinder was not using a more robust authentication method (e.g., SSL using server- and client-side authentication), vendor personnel responded that when the product was developed, SSL was not widely adopted.[4] (Apparently, the product team has not revisited this architectural decision in five years, despite the now widespread use of SSL.) Because Siebel's eService's runs on Microsoft's Internet Information Server (IIS) only, which has numerous security issues[5], the risk of an IIS server compromise then compromising the entire SSO infrastructure was deemed unacceptable. Therefore, Netegrity's product was not selected.

With Oblix's NetPoint product, we were pleased that Oblix had at least taken a more thoughtful approach to system/component authentication, using X.509v3 certificates. Additionally, Oblix meets most of our security requirements, and more favorably than Netegrity's product. So, Oblix's NetPoint product was selected for our SSO implementation.

Initial, Non-Production Deployment

Actually getting that Oblix implementation to function correctly proved to be frustrating and time consuming. While NetPoint supports two production modes of system authentication, "Simple" using self-signed X.509v3 certificates and "Certificate" using third party X.509v3 certificates (e.g., issued by VeriSign), apparently our company was the first to attempt a deployment using "Certificate" mode. We stumbled across several technical challenges. For example, NetPoint

[3] Frequently Asked Questions.

[4] This is a rather weak argument by the vendor, since SSL client-side authentication has been available since SSL 3.0, which was documented in an Internet Draft and incorporated into Netscape Navigator 2.0, both released in March 1996.

[5] See Microsoft's own IIS Security Web page.

uses a proprietary NetPoint Access Protocol over SSL for secure communications between extranet Web servers installed with the Web Gate component, and the more protected NetPoint Access Server. Unfortunately, the application requires a range of TCP[6] ports to be available to check component availability. When using a proxy firewall,[7] it must be specially configured to handle this range of ports (e.g., using TCP All Ports Generic Service Passer), instead of being limited to a single port, which is preferable security-wise.

Additionally, the business unit supporting customer access wants a challenge/response capability in the SSO product, so that if a customer forgets his/her password, then he/she does not need to contact customer support to have his/her password reset. This is a valid business need, improving customer support efficiency. However, another security-related challenge is that the product has no capability to limit input into the user's challenge nor the associated response. To do so requires custom coding, after a programmer has received special instruction on the vendor's Application Programming Interface (API[8]). "Out of the box," there is no capability to limit either "type" of input (e.g., A-Z, a-z, and 0-9), nor is there the capability to limit the "amount" of input (e.g., minimum number of characters required to input and maximum number of characters accepted for input). Instead, the malicious user could attempt a buffer overflow with input specifically intended to gain administrator access to the system.

Deployment Architecture

Initially, we had believed that the deployment architecture would probably look similar to what the vendor had suggested:

[6] Transmission Control Protocol -the TCP part of TCP/IP. TCP and UDP (User Datagram Protocol) are the two transport protocols in TCP/IP. TCP ensures that a message is sent accurately and in its entirety.
[7] See "Building Internet Firewalls" by D. Brent Chapman & Elizabeth D. Zwicky.
[8] "A language and message format used by an application program to communicate with the operating system or some other control program...or communications protocol. APIs are implemented by writing function calls in the program, which provide the linkage to the required subroutine for execution. Thus, an API implies that some program module is available in the computer to perform the operation or that it must be linked into the existing program to perform the tasks. (CMP TechWeb's TechEncyclopedia)

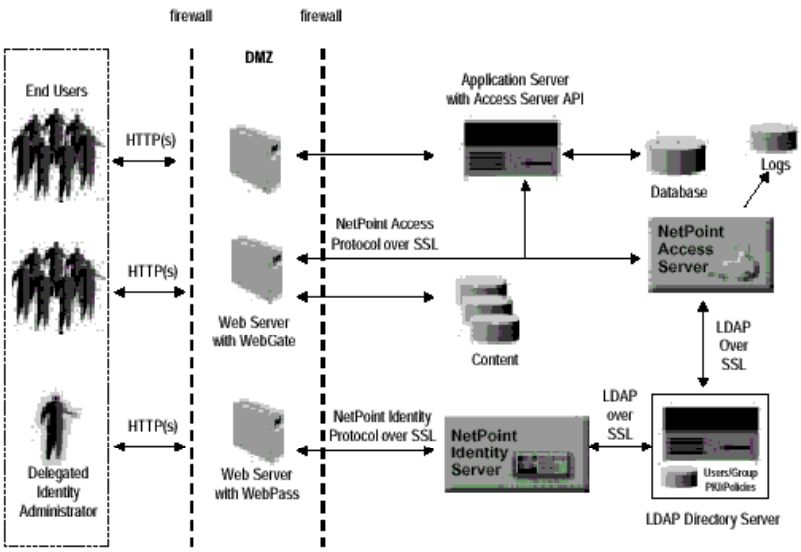

Figure 1: "Oblix NetPoint 5.1 Installation and Setup Guide Final Draft"

However, detailed review of the SSO authentication and authorization process within a partner's (non-employee) access to the Siebel CRM system revealed that the application flow amongst all the components is:

1. Client browser connects to a Web server. Client-supplied credentials are passed to SSO Agent running on the same server.
2. SSO Agent sends the credentials via a proprietary protocol (IP) over an SSL encrypted connection to the SSO Access Server.
3. SSO Access Server decrypts the information and re-encrypts the credentials and performs a secure LDAP (SLAPD) query.
4. LDAP server authenticates and authorizes the supplied credentials and sends the response over a secure LDAP reply (SLAPD) to the SSO Access Server.
5. SSO Access Server sends the authorization and authentication information back to the SSO Agent running on Web server.
6. The SSO Agent (via Web server) sends an encrypted session cookie back to the client's browser.
7. Client's browser now connects to another Web server.
8. Encrypted session cookie is passed to SSO Agent running on Web server.
9. SSO Agent decrypts the client credentials and passes them to SSO Access Server over an SSL encrypted connection.
10. SSO Access Server decrypts the information and re-encrypts the credentials and performs a secure LDAP (SLAPD) query.
11. LDAP server authenticates and authorizes the supplied credentials and sends the response over a secure LDAP reply (SLAPD) to SSO Access server.
12. Client connection is redirected via the CRM Web Extensions Agent running on the Web server to the primary Scheduler.
13. Scheduler determines the best application server to redirect the client session to and

redirects accordingly.

14. In the case that the client credentials are stale or a password change is forced, Steps 1 through 5 are completed. If the password is to be changed, the client passes new credentials (e.g., password) to SSO Agent, which then passes them via an SSL encrypted connection to SSO Identity Server.
15. SSO Identity Server updates the LDAP server over a secure SSL connection. Steps 1 through 6 are completed after the update, which validates a successful password or credential change.
16. In the case of administrative access, a secure (SSL)/(HTTPS) connection is made from the administrator's machine, running the SSO Administration Client to the SSO Administration Server running a Web server.
17. Administrator's credentials are passed to the SSO Agent running on the Administration Server, which then passes them via an SSL encrypted secure connection to the Access Server.
18. Access Server decrypts and re-encrypts the credentials and initiates a secure LDAP query.
19. Once authenticated, the administrator uses the SSO Access Manager tool running on the a Web Server (SSO Administration Server) to modify the LDAP server.
20. Administrator may use the SSO Agent to gain access to the SSO Identity Server via an SSL connection.
21. Administrator (via the Identity Server) may access the LDAP server for administration purposes via SSL.

Visually, this application flow looks like:

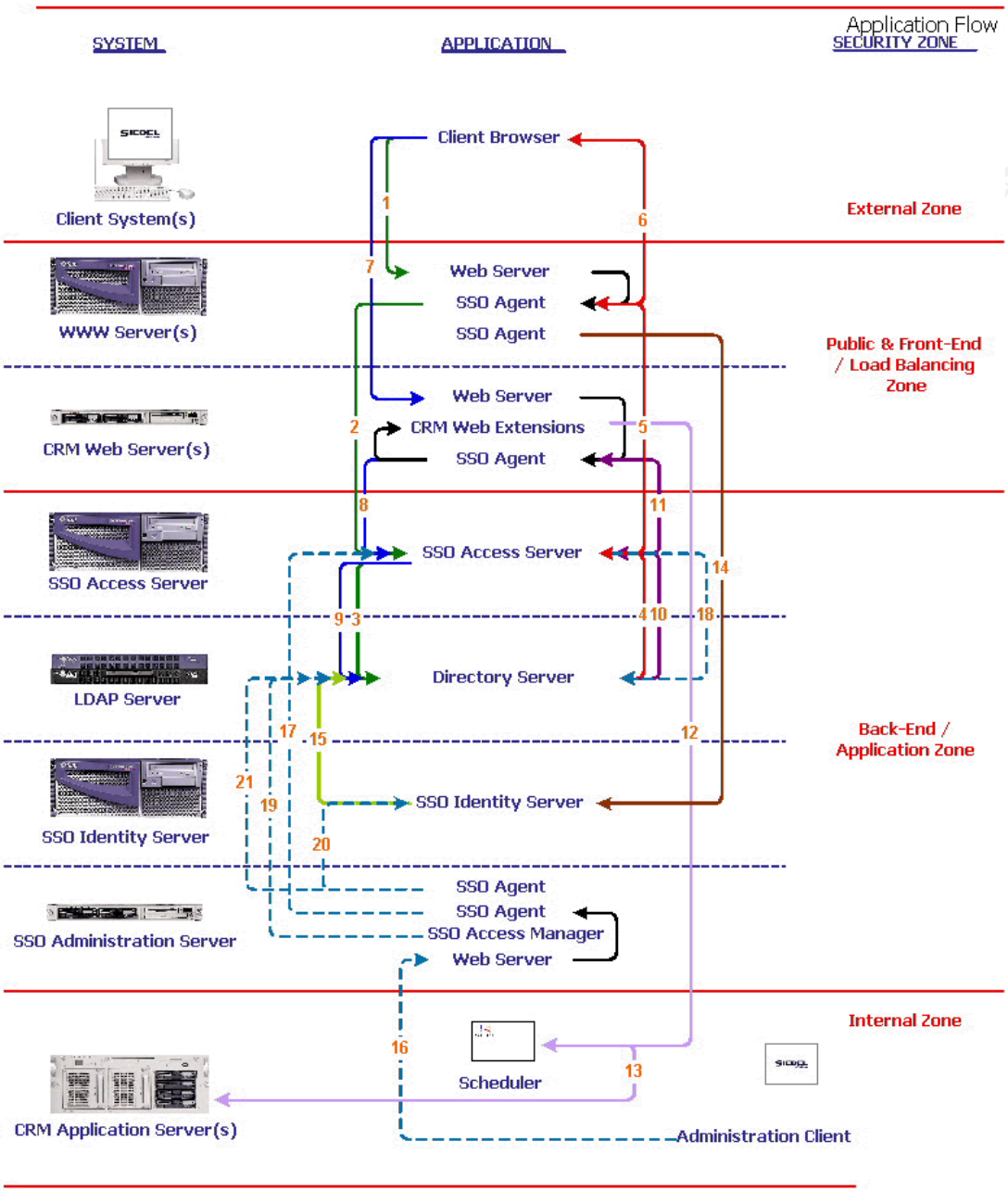

Figure 2: SSO Application Flow. (Diagram designed and maintained by co-worker; "sanitized" by author.)

The important consideration here is that SSO, while improving some aspects of security, is not a total security solution unto itself. In fact, SSO addresses only a limited number of security considerations – and even then, some of the security considerations addressed by SSO might not be robust enough.

Given the above considerations, the deployment architecture we actually implemented looks like:

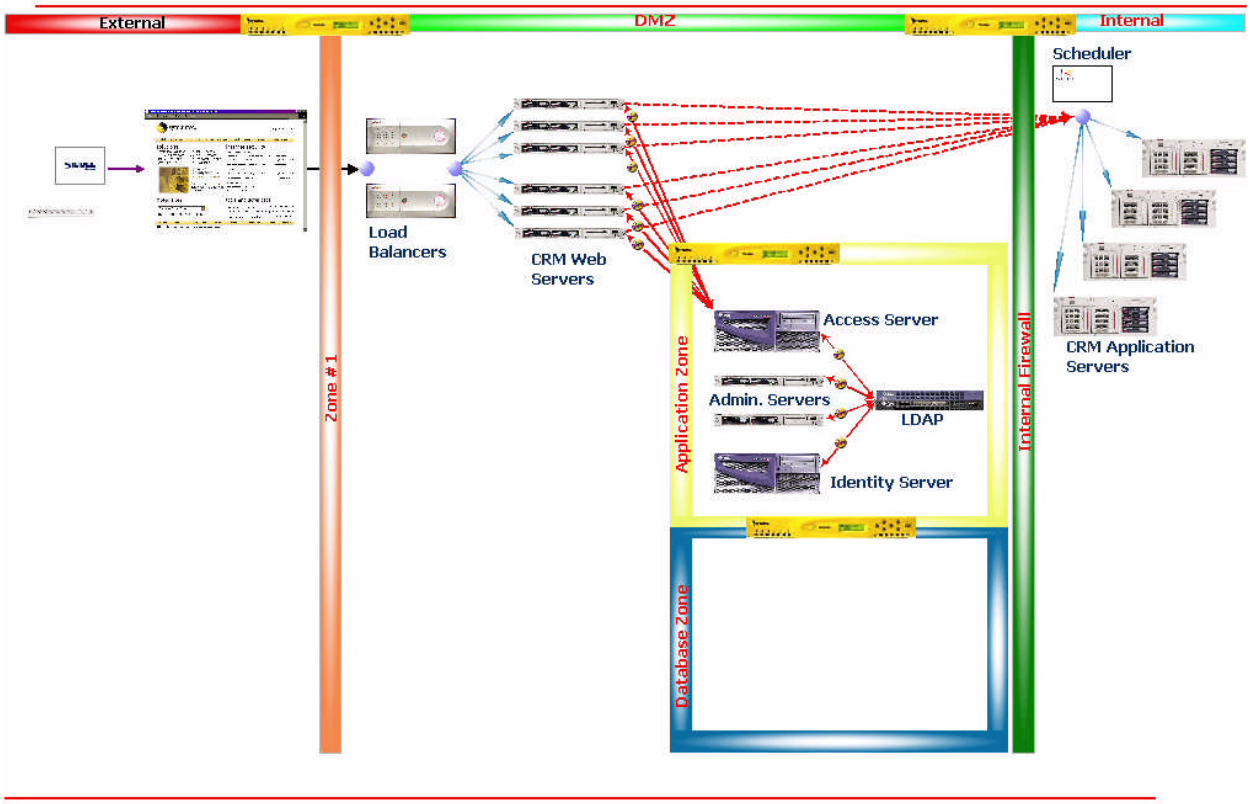

Figure 3: SSO Security Architecture. (Diagram designed and maintained by co-worker; "sanitized" by author.)

You can see the addition of another zone inside the "DeMilitarized Zone" (DMZ)[9] between the Oblix suggested deployment architecture (i.e., the "Application Zone"), and the actually deployed architecture. Additionally, the issues identified above necessitated the use of host-based vulnerability management and host-based intrusion detection systems (IDS) on all components of the architecture, as will be discussed below. These host-based security installations are in addition to the identified firewalls and network-based IDS in the DMZ to help ensure security.

Load Balancing Challenges: How to Make SSL Persistence Work

In addition to the security challenges in this SSO implementation, there were further operational challenges as well. The SSO product deployed uses encrypted session "cookies" (see RFC 2109, "HTTP State Management Mechanism") to authenticate users. That use of cookies unto itself was not an issue. The problem was interoperability with the already implemented local load balancing solution. Load balancing helps to make the most efficient use of multiple servers. To do this, however, some method for making sessions persistent must be used. While there are several such persistence methods available (e.g., source, server, virtual IP address (VIP)[10], SSL, cookie persistence, and destination address affinity), they all have their own benefits and

[9] "*DMZ*, which stands for *De-Militarized Zone* (named after the zone separating North and South Korea);" "Building Internet Firewalls" by D. Brent Chapman & Elizabeth D. Zwicky; First Edition, November 1995.
[10] VIPs are logical IP addresses.

drawbacks. In our case, the best choice at the time was to implement load balancing with cookie persistence. However, use of cookies to implement SSL persistence, brought up three important considerations:

1. where termination of SSL sessions occurs
2. use of NAT for security purposes (impact on IP addresses in cookies)
3. use of proxying for security purposes (impact on proprietary protocol operation)

For security purposes, it is preferable to terminate SSL sessions on the Web servers themselves. This ensures an end-to-end encryption tunnel from the client's browser to the Web server. However, termination of SSL sessions on the Web servers themselves raises a security consideration. For the load balancers to load balance SSL sessions, they can terminate those SSL sessions. While not required, this termination point is the best option for load balancing these sessions (using the SSL session IDs). That means that the logical connections behind the load balancers are unencrypted HTTP. Those unencrypted HTTP sessions from load balancers to Web servers are not a serious security issue, provided that other security measures have been implemented.

Another consideration in this implementation is the use of Network Address Translation (NAT)[11] by the load balancers. This is done partly for security reasons to hide the IP addresses of the Web servers logically behind the load balancers. This hiding of IP addresses makes it more difficult to map or reconnoiter a network in preparation for an attack. Additionally, the use of NAT allows non-routable IP addresses (see RFC #1918, "Address Allocation for Private Internets") to be used on the Web servers, thus conserving allotted routable IP addresses. However, the problem encountered with use of NAT by the load balancers is that the SSO implementation relies on authentication based in part on the IP addresses of the Web servers being accessed. This checking of authorized IP addresses allowed to access, uses information stored by the SSO application in the cookie it places on the user's system. Use of NAT causes these "two" IP addresses (i.e., the one(s) written by the SSO application in the cookie and the one presented to the SSO Access Server by the load balancer) to not match, thereby causing SSO to fail. Additionally, after several in-depth discussions between company implementing, SSO vendor's technical personnel, and load balancing vendor's technical personnel it was not entirely clear that the load balancing product was not trying to rewrite the SSO's cookie values, even though the SSO cookie is encrypted and has a different NAME from the load balancing cookie ("Obssocookie" versus "BIGipCookie" respectively).

The third operational consideration that had to be considered in this implementation of SSO and load balancing is the load balancers acting as proxy servers.[12] While proxying often involves the

[11] Network Address Translation: "An IETF standard that allows an organization to present itself to the Internet with one address. NAT converts the address of each LAN node into one IP address for the Internet and vice versa. It also serves as a firewall by keeping individual IP addresses hidden from the outside world." (CMP TechWeb's TechEncyclopedia)

[12] "Also called a 'proxy' or 'application level gateway,' it is an application that breaks the connection between sender and receiver. All input is forwarded out a different port, closing a straight path between two networks and preventing a cracker from obtaining internal addresses and details of a private network.

Proxy servers are available for common Internet services; for example, an HTTP proxy is used for Web

use of NAT, a further issue is the proxy's ability to properly "rewrite" the packet in each protocol presented to the load balancers. While all HTTP proxying *should* adhere to the HTTP specification (see RFC 2616, "Hypertext Transfer Protocol -- HTTP/1.1"), that does not mean that all products handle the Connection header field properly. And, it especially does not mean that all products handle cookies the same, as cookies are not even mentioned in the HTTP/1.1 specification – even if the vendors states compliance with HTTP specification.

While the three SSO operational issues discussed above are not strictly security-related, all are important security considerations. Additionally, these operational issues demand that the network operations, applications development, and information security personnel all work closely together to ensure that all really do understand what is happening. Failure of information security personnel to understand the application flow in detail, and/or to work closely with the network operations and applications development personnel may result in an implementation being less secure than desired.

Compensating Controls

Because of several factors, including:

- sensitivity of CRM data to be accessed by external parties
- security limitations in SSO product
- complicated topology of network configuration to support entire infrastructure
- operational considerations with load balancing introduced into overall deployment

it was decided that implementation of some compensating security controls was prudent. A "two-way belt and suspenders" approach was best to ensure security of the topology was agreed upon.

The first "belt and suspenders" was the decision to deploy a host-based security tools on every component of this implementation, except the load balancers. The "belt" here is use of a host-based vulnerability management tool that provides proactive protection of the servers by allowing for standardization of configurations to be checked and verified easily. Standardized configurations provide greater security by lessening the chance for misconfigurations, either through oversight or poor practice. Standardized configurations also ease the operational support burden, and decrease response time by allowing operational personnel to know what to expect in a configuration and being able to diagnose deviations faster. This proactive vulnerability management tool is also good to being able to ensure that a patch management program is adhered to, to assist with keeping up on implementation of operational and security-related patches from the vendor. In this particular implementation, patch management is even more important because Siebel's CRM effectively requires use of Microsoft's Internet Information Server (IIS). (Siebel has not certified its CRM product with any other Web server, and therefore will not support the product using any other Web server that IIS.) While all Web servers are

---

access....Proxies generally employ network address translation (NAT), which presents one organization-wide IP address to the Internet. It funnels all user requests to the Internet and fans responses back out to the appropriate users." (CMP TechWeb's TechEncyclopedia)

vulnerable to exploits, the record shows that Microsoft's IIS server is particularly so.[13] The "suspenders" portion of this security implementation is use of a host-based intrusion detection system tool provides reactive protection of the servers, by providing near real-time notification of, and response to, security-related problems on each server. While IDS is by definition reactive, timely notification of a security problem can greatly help to limit penetration and/or damage. "No news" is definitely bad news, which is one reason why IDS is an important tool.

The second "belt and suspenders" was the decision to deploy network-based security tools on the network segments of this implementation. The "belt" here is use of a network-based vulnerability management tool to provide proactive protection of the servers, by allowing for possible vulnerabilities of the components to be checked. The "suspenders" portion of this security implementation is use of a network-based intrusion detection system tool provides reactive protection of the servers, by providing near real-time notification of, and response to, security-related problems on network segments. While host-based IDS is important for telling if an attack on a specific server has been successful, or maybe is about to be successful, network-based IDS is important for providing the broader picture of what is happening on your network segments. Is it merely a single Web server that appears to be under attack, or is the problem much bigger than that (e.g., all of your external-facing CRM Web servers are being attacked – and by different methods or tools)?

The result is each component, except load balancers, has host-based vulnerability management (proactive) and intrusion detection system (reactive) protecting it, providing a view of each component. Additionally, there is network-based vulnerability management (proactive) and intrusion detection system (reactive) protecting the involved network segments, providing a wider view of the infrastructure. These deployments help to ensure that neither the Web servers nor the SSO components are compromised, and if so, there is at least near real-time indication of such.

Summary

Single Sign-On is a step in the right direction towards greater security. However, SSO is not a panacea. In fact, not only does SSO address only a limited portion of what should be your security concerns, but implementation of SSO introduces its own security considerations, which must be mitigated, and managed with the support of other IT groups as well as other business units.

[13] The National Institutes of Standards and Technology's (NIST's) ICAT metabase (http://icat.nist.gov/icat.cfm) backs up that vulnerability assertion.

Sources

D. Kristol and L. Montulli. "HTTP State Management Mechanism." February 1997. URL: http://www.ietf.org/rfc/rfc2109.txt?number=2109.

F5. "F5 Networks' BIG/ip Controller Provides Cookie Persistence." September 1999. URL: http://www.f5.com/solutions/techbriefs/cookie.pdf.

F5. "Maximum Flexibility With The BIG/ip Controller's Six Persistence Modes." August 1999. URL: http://www.f5.com/solutions/techbriefs/SSLpersistence.pdf.

Netegrity, Inc. "SiteMinder® Deployment Guide, Version 4.6." 2001.

Netegrity, Inc. "SiteMinder® Installation Guide, Version 4.6." 2001.

Oblix, Inc. "Oblix NetPoint: A Technical Overview." 2001. URL: http://www.oblix.com/public/other/wp/wp-netpoint-tech_100501.pdf.

Oblix, Inc. "Oblix NetPoint 5.1 Installation and Setup Guide Final Draft." 2001.

Oblix, Inc. "Oblix NetPoint 5.1 Administration Guide Final Draft." 2001.

R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. "Hypertext Transfer Protocol -- HTTP/1.1." June 1999. URL: http://www.ietf.org/rfc/rfc2616.txt?number=2616.

RSA Laboratories. "RSA Laboratories' Frequently Asked Questions About Today's Cryptography, Version 4.1." 2000. URL: http://www.rsasecurity.com/rsalabs/faq/index.html.

Russell L. Jones. "EAM Ain't Easy." Information Security. January 2002. URL: http://www.infosecuritymag.com/2002/jan/features_eam.shtml.

Schneier, Bruce. Applied Cryptography: Protocols, Algorithms, and Source Code in C, 2nd Edition. Reading: John Wiley & Sons, 1995. 216.