Global Information Assurance Certification Paper

Lee Anne Hart
GSEC v.1.4b
September 21, 2003

Single Sign-On: Holy Grail of Holy Crap!

Abstract
Single sign-on (SSO) is often referred to as the Holy Grail for authentication and authorization in the information technology industry - it is the ultimate solution almost every business is in search of, but only a few, if any, have successfully implemented. The overall goal of any SSO solution is one all inclusive user ID and password which provides access to all authorized systems, applications, and network resources.

Currently, there are three key methods used to create a SSO solution: tickets, password synchronization, and proxies. Reducing the number of user ID's and passwords that end-users must remember has its benefits as well as its risks. It is important to recognize and weigh those benefits and risks prior to purchasing a SSO solution. When evaluating the different SSO solutions on the market today, it is very important to understand how the vendor has accomplished SSO as well as to compare the different features offered.

What is Single Sign-on (SSO)

According to Webopedia, "single sign-on is an authentication process in a client/server relationship where the user or client can enter one name and password and have access to more than one application or access to a number of resources within an enterprise. SSO takes away the need for the user to enter further authentication when switching from one application to another". This definition can be interpreted in a variety of ways which is one reason why there are different methods for providing SSO.

There are three primary methods leading the development of SSO: tickets, synchronization, and proxies. In the ticketed method, the end-user authenticates to a central authentication server rather than to each server, application, or network resource they need separately. Upon successful login to the central authentication server, the end-user is issued an encrypted ticket which is then presented to a "ticket-enabled" back-end system (Kelley) (See Figure 1). The ticket contains an encrypted copy of temporary secret intended to be shared between a particular pair of computers. It is encrypted using a master key from the Key Distribution Center (KDC) and the ticket's intended recipient (Smith, 343). The most common instance of this type of SSO uses the Kerberos protocol. According to Webopedia, Kerberos is "an authentication system developed at the Massachusetts Institute of Technology (MIT). Kerberos is designed to enable two parties to exchange private information across an otherwise open network. It works by assigning a unique key, called a ticket, to each user that logs on to the network. The ticket is then embedded in messages

to identify the sender of the message". Actually, the term ticket originated with the Kerberos protocol (Smith, 343).

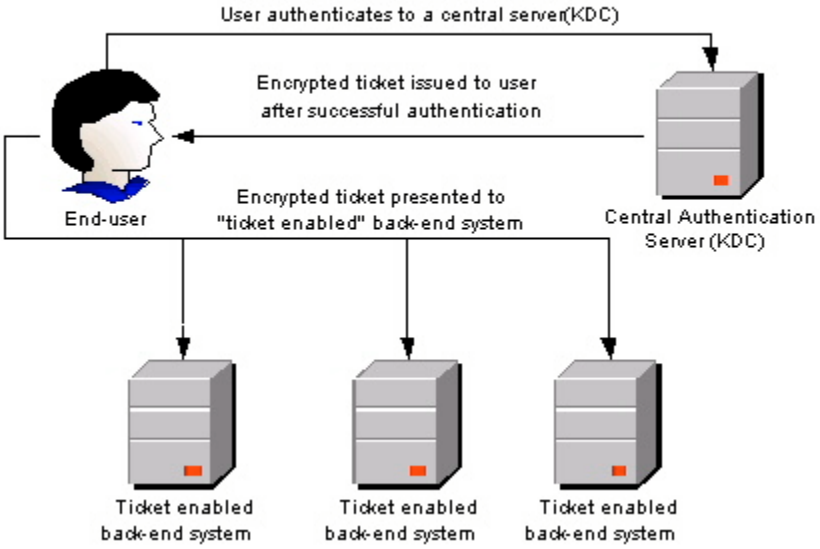

Figure 1 The ticket or token method to SSO.

The second method is password synchronization. Many people assume that SSO means all systems, applications, and network resources use the same password; however, this is not necessarily true. Password synchronization is a less-sophisticated method for creating a SSO. These systems issue and sync a main password to all other systems giving the illusion of a SSO (See Figure 2) (Taylor).

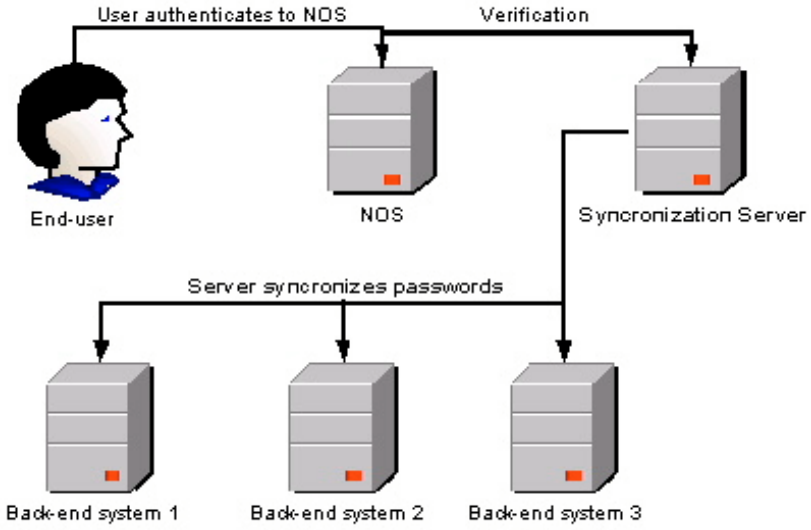

Figure 2 Password Synchronization

A third method for providing SSO, an end-user actually maintains a different password for each system, application and network resource by storing them in a database. This database can be kept on the end-users workstation or a server. Once the end-users successfully authenticates to the SSO client, the database is made available to them. When the end-user wants to access an application that requires authentication, the information is pulled from the database and transparently supplied to the application on behalf of the user (Posey) (See Figure 3). This proxy approach was developed to circumvent the need for "ticket enable" back-end systems (Kelley).

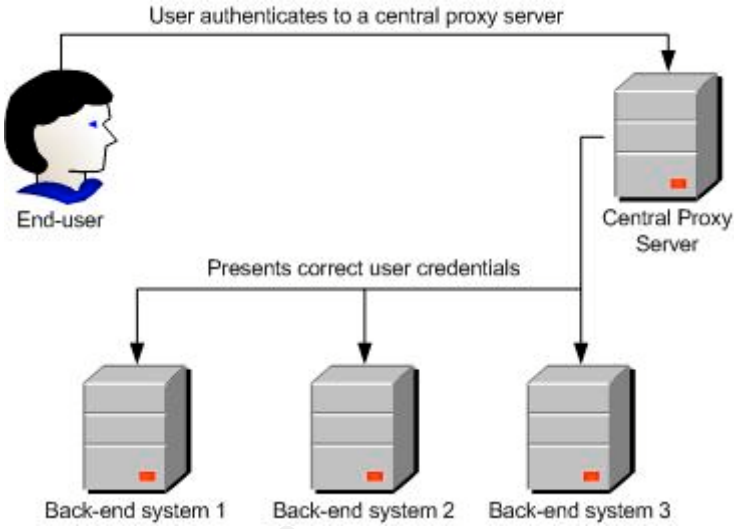

Figure 3 The proxy method to SSO.

A SSO solution that maintains a different password for each system is more secure than systems that synchronize all passwords because the possibility of exposing all applications with one password theft or breach is eliminated. For example, an organization has eight applications that require a user ID and password. If the organization implemented SSO using password synchronization, all eight applications would have the same password. If someone were to ascertain one of the passwords through password file theft, a Trojan horse, a sniffer program, etc. they would also have the password to the other seven applications and could login manually (Posey). Similarly if an end-user's SSO password is ascertained, it would provide access to all password-protected systems, applications, and network resources as well. However, the SSO password file or database, whether stored on a server or on the workstation, typically has or is capable of greater security than operating systems and applications. For example, NTLM and UNIX shadow password files are easily cracked with programs anyone can download free of charge from the Internet (Smith 47, 301).

Another weakness with password synchronization is that password construction requirements and abilities can vary greatly from system to system. In these situations, a least common denominator for password construction must be

defined that will work with all systems; in other words, the system with the weakest password construction ability becomes the standard for all systems (Yost).

To many, the ticket method is considered "true" SSO because the user actually only authenticates one time. The tickets issued after that authentication are used to grant access to resources needed. Password synchronization and the proxy method seem like a SSO because the user only types in their user ID and password once. But the user's credentials are transparently supplied to each subsequent system, application, and network resource by the SSO solution. Password synchronization and the proxy method reduce the number of user ID's and password that an end-users must remember, but unlike the ticket method, they do not reduce the number of times an end-user must be authenticated to access systems, applications, and network resources.

Benefits and Risks of SSO

There are several benefits to implementing a SSO solution as well as several risks. When deciding whether or not to implement a SSO solution, consider and weigh both the benefits and the risks involved. A few of the benefits include:

- simplifies the login process for end-users
- reduces operating cost
- potential to increase security

SSO simplifies the login process for end-users by providing one user ID and password to access all authorized systems, applications, and network resources. It is estimated that the average employee of a large corporation has between five and eight different login ID and passwords (Yost). End-users have become accustomed to this practice of having a different user ID and password for all resources they need to access. To make life easier, end-users will write down their user ID's and passwords in PDA's, on Post-It notes, and in planners. Others use the same password for all systems to limit the amount of time they spend looking for or recalling a password for a system.

SSO reduces the operating cost by reducing the number of calls to help desks for password resets. Gartner research estimates that password reset calls comprise 30% of all help desk calls and each of these calls has an associated operational cost of $32. With the increasing number of applications in use by businesses, they cannot afford the productivity lost or the cost of continuous password resets (Posey).

Average number of user ID's and passwords per user – 8
Average cost of help desk calls to reset passwords - $32
Percent of help desk calls to reset passwords – 30%
Successfully Implementing a Single Sign-On Solution = Priceless

SSO can also potentially increase security within an organization by reducing the need to write down ID and passwords for systems in order to remember them. Another potential increase in security is because end-users will have fewer passwords to remember, there is a greater chance that they will use strong passwords (Kelley). Also, some SSO products can work in conjunction with smart cards, PKI, and biometrics. Two-factor authentication can be accomplished without implementing SSO if all systems requiring authentication and authorization support multiple authentication methods. However, this type of authentication would need to be setup on each and every system. Using SSO in conjunction with another type of authorization increases security and only needs to be implemented on the SSO login.

Consolidating all logins into one all powerful user ID and password is very convenient for end-users but potentially very dangerous for system security. Proper risk management is very important when implementing a SSO solution. The risks must be weighed against the benefits. Security is the business of mitigating risk and one technique to mitigate risk is by using defense in depth or layers of protection. Some of the risks introduced by SSO include:

- single point of failure
- vendor reliability
- complexity to implement

SSO introduces a single point of failure in a couple of ways. First, if an end-users SSO password is discovered, it allows access to everything the end-user is authorized to access. Requiring strong passwords, the use of two-factor authentication, and security awareness training can help to mitigate these risks. Likewise, if the SSO server is attacked and broken into, the attacker potentially has access to all SSO passwords in the system. To help mitigate the vulnerability, ensure the server is "hardened" before it is brought online and keep it up-to-date with patches, service packs, and hot fixes. Next, if the SSO server goes down to hardware failure, operating systems crashes, etc., end-users will have to revert back to using individual ID's and passwords to access resources. To reduce this down time, ensure that regular backups are made of the server, utilize RAID technology, and keep spare hardware on-site for such an event.

Vendor reliability is often overlooked as a risk. Consumers want to ensure they will receive support when needed and that they will have access to updates and upgrades as they are released. Before purchasing any SSO solution, research the vendor carefully. According to Diana Kelly of the Baroudi Group, "some questions to ask:

- Has the vendor had the product reviewed by third-party audit?
- How long has the company been in business?
- How many customers does the vendor have?
- Will any customers act as a reference?

- Are there any published attacks against the vendor's product and if so has the vendor addressed and corrected the vulnerabilities"?
- Is the authentication method used by the SSO solution proprietary or based on emerging technologies such as LDAP or X.509?

The complexity of the implementation depends on the environment the SSO product is being implemented. This is where careful planning can make or break the implementation. There are often restrictions on what type of solutions can be put in place because of existing network architecture (Kelley). Legacy or non web-based systems are more difficult to implement than web-based because system changes may be required. For example, the "ticket method" requires that the back-end systems be "ticket-enabled". Also, it is important to know if the SSO solution uses proprietary authentication methods or emerging standards such as LDAP or X.509. If the SSO is not based on an emerging standard, it might mean throwing out the solution in a few years because of compatibility issues (Yost). Planning and research are critical to mitigate this type of risk as well as strong project management skills.

SSO Features

Besides offering various technology approaches to SSO, many vendors also incorporate different features in their SSO solution. A few of the common features include:

- Supported Platforms – What are the server and workstation operating systems requirements?
- Is the SSO based on password synchronization?
- Authentication method – Is the authentication method used based on emerging industry standards or proprietary methods?
- Transparent Implementation – Is there client software to be installed?
- Personalization – Are customized user specified features provided?
- Central Management – Can many SSO servers be controlled with one console? Can user access be controlled from one central location? This is very important for terminating employee access to systems and applications.
- Smart Card Support – Is authentication via smart card allowed?
- Scalability – Can the product be deployed in large and small organizations? Scalability is the ability of be deployed into a wide variety of various sized organizations.
- Intrusive – Are changes required to the applications the SSO will manage?
- Fault tolerance – Are the built-in methods for continuity if the SSO solution is unavailable?
- Logging – Are auditing features available?

The importance given to each of these features depends on how an organization plans to use the SSO solution. For example, if an organization uses primarily UNIX workstations, they would need a solution with UNIX client support.

Comparing features is equally important in planning for the future. If an organization is planning on implementing some form of two-factor authentication in the future, the SSO solutions they choice to implement now needs to be able to support two-factor authentication.

The chart below compares Novell's SecureLogin, Imprivata's OneSign and PassGo Technologies PassGo SSO. Novell and PassGo are two of the market leaders, while Imprivata is a newbie in the market. Novell has been in the IT business in since 1979. In 1983, Novell introduced NetWare, the first LAN software based on file-server technology. Novell has its headquarters in Utah. Imprivata, a Massachusetts startup, was founded in April 2000, however, OneSign, the company's first product, was not released until March of 2003. The unique feature of OneSign is that it is the first SSO solution to appear as an appliance (Messmer). Founded in 1983, the original PassGo company was acquired by AXENT and subsequently AXENT was acquired by Symantec. In August 2001, PassGo Technologies re-established itself as an independent company. PassGo's headquarters is in Pennsylvania.

	Novell SecureLogin 3.0	**Imprivata OneSign**	**PassGo SSO**
Server Platforms	Novell NetWare 4.x or later Windows NT4/2000 Solaris 2.6 or later Linux 2.2 with glibc 2.1.3 or later	A pair of synchronized, redundant Linux-based appliances	Windows NT4/2000 AIX, Solaris, HP-UX, OS/390 Mainframe
Workstation Platforms	Windows NT4/2000/XP Windows 98/ME	Windows NT4/2000/XP Windows 98	Windows 9x, Windows NT4/2000/XPIBM OS/2 2.1/Warp
SSO Method	Proxy	Proxy	Password Synchronization
Authentication Method	LDAP-compliant	LDAP-compliant	Proprietary
Client-side software required	Yes, SecureLogin's core software runs on the client. Only the actual data is stored on the server.	Yes, OneSign agent can be "pushed" using MSI-compatible installer, or can be downloaded from the appliance by users via URL sent to users by email "notify" feature.	No, application is completely server based.

	Novell SecureLogin 3.0	Imprivata OneSign	PassGo SSO
Personalization	Yes, SecureLogin supports roaming desktops.	No	No
Central management	Yes	Yes	Yes
Smart Card Support	Yes	Yes	Yes
Scalability	Novell claims that there is no limit to the number of supported clients or to the number of passwords that can be stored.	Each appliance can support up to 5000 users. Multiple appliances can be used in an environment.	PassGo's largest real world deployment consists of 30,000 users in a mainframe environment, and a few thousand users in a Windows environment.
Intrusive	No, the core product runs on the client end.	No, it requires no changes on the back-end applications.	No, application is completely server based for minimal deployment difficulties.
Fault tolerance	Yes, local encrypted caching to ensure that network downtime does not affect single sign-on performance	Yes, a pair of synchronized, redundant appliances. If the primary becomes unavailable it transfers to the failover. Also, SSO credentials are encrypted and cached on the client.	All passwords are synchronized so if the SSO server goes down, the end-user can login manually.
Logging/Audit	Yes	Yes	Yes

Conclusion

Defining SSO is simple; however getting a group of IT professional to agree on the meaning of SSO is difficult. To some, SSO is password synchronization. To others, it is authenticating to a password store which then handles all future logins transparently. And yet others see SSO as authenticating to a server which then issues a ticket to present to all others systems. All of these options have one feature in common--one user ID and password for each end-user. The goal of one all inclusive user ID and password can be very beneficial, especially to

end-users and the help desk, but it can also create a huge vulnerability in an organization security if it is not implemented properly. Several vendors including Novell, Imprivata, and PassGo have attempted to box and sell a SSO solution. While each of these products do accomplish the goal of one all inclusive user ID and password, they do so using different methods and offer different features to consumers. However you view SSO, Holy Grail or holy crap, it is important to recognize that the idea of SSO has been around for a long time and IT professional will continue search for the ultimate solution.

Works Cited

http://www.imprivata.com. 14 September 2003.

http://www.novell.com. 14 September 2003.

http://www.passgo.com. 19 September 2003.

Kelley, Diana. "Planning for Single Sign On (SSO) Aligning Expectations with Reality." 2002. July 2003 <http://www.baroudi.com/pdfs/single_sign_on.pdf>.

"Kerberos". Webopedia 19 September 2003 <http://www.webopedia.com/TERM/K/Kerberos.html>

Messer, Ellen and Denise Dubie. "Security Worries Give Startups Hope". Network World. 6 January 2003. 14 September 2003 <http://www.nwfusion.com/news/2003/0106securitystart.html>.

Posey, Brien. "Legacy Single Sign-On: A Competitive Analysis." Intranet Journal 14 October 2002. 31 July 2003 <http://www.intranetjournal.com/articles/200210/pij_10_14_02a.html>.

"Single signon." Webopedia 19 September 2003 http://www.webopedia.com/TERM/S/single_signon.html.

Smith, Richard. Authentication: From Passwords to Public Keys. Boston: Addison-Wesley, 2002.

Taylor, Laura. "Understanding Single Sign-on." Intranet Journal 28 May 2002. 7 July 2003 <http://www.intranetjournal.com/articles/200205/pse_05_28_02a.html>.

Yost, Guy C. "Planning for Single Sign-On: Part 1 – An Overview." Technical Support January 1999. August 2003 <http://www.naspa.com/PDF/99/T9901003.pdf>.