

“SINGLE SIGN ON UNDERNEATH THE HOOD” WHAT SENIOR MANAGERS NEED TO KNOW

This briefing is designed for senior managers wanting to know the implications of single sign on for their enterprise.

Copyright, 2001. [Guy Huntington, HVL](#).

So you want to do single sign on. Your users have too many usernames and passwords to remember. A large portion of your Help Desk's time is spent sorting out authentication and access issues. BUT...besides the apparent benefits for your users, have you really thought out the many implications of single sign on from a management, system and security perspective? That's what this paper discusses.

SINGLE SIGN ON OR SINGLE POINT OF FAILURE?

Vendors, managers, users and consultants loosely use the phrase “single sign on”. It's a very appealing concept to a user who has to cope with potentially several or more different log-on mechanisms in their workplace. However, have you and the users thought about what true single sign on implies?

Hypothetically, if your enterprise implemented the same authentication method for every application, you're potentially creating a single point of failure. If the single sign on is breached (malicious, criminal, ineptitude or otherwise) much of your security for all systems is in serious jeopardy.

What you're probably after as a manager is maybe not single sign on but:

- Reduced sign on for limited access to many applications
- Strong authentication for protecting applications
- Enhanced enterprise wide access control

REDUCED SIGN ON FOR LIMITED ACCESS TO MANY APPLICATIONS

The first thing you need to do as a security or IT manager contemplating single sign on is to conduct a risk analysis for each application. What happens if there is a breach? Can you control authorization to a fine-grained level? How is the identity created, stored, managed, terminated and archived? What authentication mechanisms will it support? What are the end-to-end business and security processes surrounding each application?

You're going to come up with a list of applications for which the risk warrants having a common authentication mechanism for limited basic access but that's only the start. Why? The answer revolves around the identity management, trust, authentication and authorization mechanisms.

IDENTITIES

The simple act of creating an identity is most often done differently in each application, usually in a database. The identity is stored in a different manner to no open standards. However, in your efforts to have a common authentication mechanism, you need a common way of agreeing on what the identity is. If each of your applications has no standard way of agreeing on who the identity Jane Doe is, you've got an information-plumbing problem.

Some web security vendors will use yet another database to centrally control the identity. This is problematic because this database will also have no open standards, thus making the identity hard to exchange between systems internally (like an intranet) or externally (like an extranet). Furthermore, there's a performance impact that the central identity repository will face.

Instead of each application authenticating the users themselves, you're going to implement a solution in which a central enterprise identity repository may incur up to thousands of hits per second doing identity checks. Unless you're using expensive high performance databases specially tuned for fast lookups, you're going to find the database solutions slower, more expensive, hard to scale and manage. That's where LDAP (Lightweight Directory Access Protocol) comes into play.

LDAP directories are relatively easy to scale, implement, partition and can tolerate high performance environments for identity lookups at reasonable cost. As importantly, LDAP directories offer ways to standardize high-level tombstone identity information such as how last name, first name, common name, e-mail and telephony details are stored. Therefore, you definitely want to look for a web security vendor that uses native LDAP since these solutions will be much easier to implement, coordinate and scale than database driven solutions.

The LDAP directory solution will not replace each of your databases within your applications. Instead, it will act as a central coordination point across your enterprise for identity tombstone level information.

Most of the identity tombstone information you place within the directory will probably come from other applications. You need to decide for each piece of directory information (called an "attribute") who will be the authoritative source for the attribute.

For example, you may decide that employee ID, name and other similar information may only be issued and maintained by your Human Resource Management System (HRMS), while the e-mail address will only be issued and removed by your IT department's applications. Once a change is made in these applications the directory is notified and updated. Other applications requiring this information are pointed to the directory (since it uses open standards) so that when a change is made in the directory they too automatically pick up the changes, thus reducing time, labor costs and improving efficiency.

You also need to be aware of the issues arising from getting many different applications to agree on who an identity is that's stored in the LDAP directory. Issues will include identity data cleansing (the same identity stored differently in different databases so that it looks like a separate identity when it actually isn't) and unique global identifiers (using a unique alphanumeric ID for the identity in the LDAP directory that coordinates with each unique database identity ID).

There's another issue to consider. What do you do with the identities for contractors, temps, customers and business partners? Who is going to be the definitive, authoritative source? This can be complex.

In the case of customers an obvious answer may be that the Customer Relationship Marketing (CRM) package will be authoritative. However, in many enterprises the customer information may be held in several applications or data repositories for which coordinating them with each other and then the directory is complicated.

You might think that contractors and temps should come from the HRMS application. This also is not always straightforward for several reasons.

HR management may not want to administer contractors and temps because of the increased management costs, time and complexity. In the United States, many HR managers don't want contractor or temp information placed in their HRMS since there are potential legal implications arising from court cases inferring that if this is done, the contractor or temps may be considered employees in some situations.

Often, the LDAP directory becomes a logical place to hold contractors and temps. The issue then becomes how to get their identities into the directory and integrated with business processes that are very loose for contractors and temps across many departments, systems and applications.

Business partners' employees' identities also pose challenges. A portal application may be the authoritative source for their identities. However, in many enterprises, there are often multiple applications and data repositories each storing portions of the identities. These must be coordinated at both business and security process levels with a directory (an example in managing extranet identities from a security perspective is discussed further in this paper).

Once you get the tombstone level information for your employees, contractors, temps, business partners and customers into the directory you might think your problems are over. This isn't the case. There are some significant security issues to consider.

The identity is one of the bedrocks of your access control system. From a security perspective, you want to lock down who can view or modify each identity attribute within the directory as well as who should be notified if an attribute value changes. The same applies from a legal perspective in many jurisdictions around the world where guarantee of personal information security is entrenched in legislation.

For example, your enterprise may decide that some portions of the employee information such as job title, can be viewed by everybody but only changed by the Director of HR or their designee. Sensitive information such as home phone numbers, employee pay bands, etc. may have tight restrictions on who can view and modify these attributes. When changes are made to the values, a notify request may be automatically be sent by the web security software in the form of an e-mail to a designated person or persons or code to a system alerting it to a change.

Surprisingly, some web security products don't have attribute by attribute control of view, modify and notify within the directory. The control is done at a higher level of who gets to see or modify all the attributes. Even though the vendor says they offer LDAP in addition to their database repositories for identity storage and management, the way they go about managing attribute level information makes the systems hard to manage and inflexible.

CENTRAL AUTHENTICATION

Next, you need to think about how your applications will trust a central authentication mechanism. Modern network operating systems such as Unix, Linux, Win2000, Novell etc., are LDAP aware and will accept using the directory as a source for identity authentication. Other applications such as more recent versions of PeopleSoft and SAP are now LDAP compatible. Unfortunately, you're going to find a lot of applications either aren't LDAP compatible or, you might not have the budget or time to upgrade to versions that are. Now you've got another information plumbing problem.

If you're going to make single sign on work from a user's perspective for those low risk applications that are not LDAP compliant, what are you going to do? The answer is what I call "fudged single sign on".

To solve the problem you will create an enterprise LDAP identity directory. As part of the directory design, you may also include storing attributes within the directory containing the authentication information that other non-LDAP compliant applications require. For example, let's assume that your manufacturing application isn't LDAP compliant but you still want to use a common authentication mechanism.

When Jane Doe logs in she authenticates against the central enterprise LDAP directory. Upon successful authentication, the directory passes a username and password on to the manufacturing application, which the manufacturing application understands and will use to re-authenticate the user with. Note however that the user doesn't see this so Jane thinks she has achieved single sign on. There are many things to consider when plumbing these solutions.

Is the communication between the user and the LDAP directory secure? If not, a malicious person positioning themselves electronically between the user and directory could see the usernames and passwords. Is the communication secure between the directory and the other applications it's passing values to? If you are using a fudged solution, how do you synchronize the directory and the manufacturing application such that an identity change in either the manufacturing application or the directory is passed on? How secure is the directory to an attack? How secure are individual attributes stored in the directory such as the manufacturing username and password?

Web security vendor products offer post authentication actions, usually in the form of HTTP actions. By using these, you can pass directory attribute information such as other usernames and passwords to the older applications. If these applications accept HTTP then you're in good shape. However, if they don't, your single sign on project may substantially slow down at this point. Programmers and analysts will wrestle with how to securely pass information to and from the directory to the application.

AUTHENTICATION SCHEMES, MAINTAINING STATE AND MULTIPLE DOMAINS

Then there is the issue of authentication schemes. Pushing aside for the moment how to achieve stronger authentication (see points further on in this paper), what if you want to use different authentication mechanisms as part of your authentication re-engineering?

For example, what if you decide to let Jane Doe use basic username and password authentication for the network and intranet access, but want her to use a digital certificate when she wants to access the financial application. Your web security solution needs to give you the tools to automatically route between secure and unsecured servers when desired and to seamlessly integrate with other authentication mechanisms.

What if you want to establish time out rules for your single sign on such that when Jane doesn't use the terminal or mouse for twenty minutes a re-authentication is forced? How is another application to know that Jane has already successfully authenticated to a level for which the application can accept her existing authentication?

These seemingly simple demands are made complex because the Internet is stateless. To solve this, web security vendors use encrypted cookies placed on the user's browser by the web security tool to establish state (i.e. when a session began). The cookie may also contain the authentication level for which the user has been successfully authenticated for during the session. You're going to want to revisit your enterprise cookie policy to make sure your practices conform to the policy.

Optionally, some vendors may use an SSL/TLS server (Secure Sockets Layer/Transport Security Layer) or an application server that can maintain state. As the secure session is initiated, the server logs the start of the session time. This may or may not work for your environment.

You may not want all sessions to be secure using encryption because of increased bandwidth, low risk or cost implications. Perhaps you don't want yet another application server in your network or, find that this adds cost and complexity when scaling systems.

What if this is made more complicated by the fact the financial application Jane wants to use is in a different domain within your company or, in a business partner's system? How do you trust the authentication from one system or domain to another?

The web security solution must provide a mechanism to recognize different domains for which no re-authentication will be required (see points later in this paper). The vendor should also be adhering to new emerging standards for passing authentication and authorization between systems "SAML" (Security Assertion Markup Language).

AUTHORIZATIONS

Then there is the issue of authorizations (who gets to access what). The goal you're probably after is to enable a few authentication mechanisms to provide basic access to many applications. The corollary to this is that as the user drills to more sensitive resources or depths within the applications, higher security will occur. This means you need multiple layers of authentication and authorization.

What if your applications are older and have poor authorization ability? What if they were made for days when only super-users would run the entire system and not hundreds, thousands or millions of users coming in via the internet, intranet and extranets?

Your single sign on project may hit the wall of reality at this point. You're going to have to decide how much money, time and effort are to be expended to extend single sign on to each application while maintaining acceptable risk levels. In some instances, the web security tools will be able to help.

Within the products you can create multiple authorization levels for accessing resources and applications by individuals, groups, roles, titles, positions and combinations thereof. For example, you may have an older telephony application for which the authorization can be done externally to the application i.e. you decide who can or cannot access the application. You'll use the web security product for the authentication as well as the authorization. The same would apply to sensitive resources such as web pages for which you want to set authorization levels for.

Setting authorization rules can be very complex. There can be many layers of rules created specifying who can get access to what. It's made even more complicated by the fact that life is full of exceptions.

For example, you might create an authorization rule for a particular set of web based reports generated by your data warehouse which are used by sales representatives in the Western Region division of the Sales group. There might be general rules specifying that only identities belonging to the group "western sales" in addition to the VP Sales, VP Marketing and CEO can access all reports at "acme.intranet.sales/western region". This general rule eliminates everybody else from being able to get at the Western Region's sales reports.

This rule however will be superceded by a more specific rule saying that each sales rep can only look at their own reports and not each others. The more specific rule must take precedence over the more general rule.

Then there is the exception that Jane Doe, who's a sales representative in the Western Region, is working on a special project with Mary Smith, another Western Region sales representative. Jane needs access to Mary Smith's reports in addition to Mary and Mary's manager being able to view the reports.

There's a couple of issues you need to address when choosing your web security vendor and then doing the actual implementation.

Single Sign On Underneath The Hood

How easy is it to create and understand the authorization rules? Some vendors have very complicated ways to create and then check the rules before implementation. Further the rules themselves may be hard to understand when viewed on the screen.

Then there is the issue of who gets to view rules? Later in this paper an example is given of an extranet where different people are able to view and/or modify authorization rules. In many cases, you'll want to have fine grained access control over the management of who gets to create, view and modify authorization rules. If you don't, you'll be creating a potentially slow and unwieldy system as the least little logical change to an authorization rule might have to be forwarded to many layers of management to finally get achieved.

In the example given above of Mary and Jane, it would be logical to have delegated the ability to their manager to make a change in who gets to view their sales reports. When this rule is changed a notification may be sent to the IT Security group noting the change to the authorization rule. Essentially what you want is the ability to create delegated authorization sandboxes where people are given enough authority to make changes within a limited set of policy rules.

The delegation should extend to multiple layers and not just one layer down. For example, the IT security group may hold final authority on rule creation, modification and deletion. It in turn may have delegated some of its authority to the Sales group so they could make limited changes to viewing and modifying access to certain resources and applications determined by the IT Security group. In turn, the Sales group would delegate portions of its authority to Mary and Jane's manager.

The authorization rule viewing also needs to extend to your Help Desk. When users such as Jane or Mary call in to complain they can't get at an application or resource, you want the flexibility to let the Help Desk see the rules (where you want them to) in order to assist the user. If you use this functionality, the rules need to be easily interpreted in order for the Help Desk to assist.

It's this kind of functionality that can make or break the implementation and maintenance of a web security system. If the systems are inflexible and hard to understand, the costs and time to implement increase, managers and users come to resent the tedious requirements, the rules are hard to interpret and inevitably, people will start to develop their own work-around to the inflexible systems. Some web security systems are very hard to scale quickly, easily or cost effectively.

Where the ability to make authorization rules in the web security product won't help is when you want to do multiple layers of authorization within an application and the application itself can't do it. If this is the case, your choices generally are to upgrade the application, replace it or decide to live with what you have.

These are just a few of the issues when considering implementing a reduced sign on initiative. As you can see, there is much more than meets the eye.

STRONG AUTHENTICATION FOR PROTECTING APPLICATIONS

If you're like many other IT security managers, you probably want to increase the strength of the authentication mechanisms being used including those for a single sign on initiative. There is much to consider.

First and foremost is the point that security is a process not a tool. Therefore, even if you have the most wonderful authentication method implemented into poor business and security processes, your authentication may amount to very little security. When reviewing authentication with your managers and users, first review the end-to-end processes surrounding the authentication. If you're up against a smart hacker either internally or externally, they're going to go after the weakest link in your authentication process.

Here is a brief review of some things to consider with different authentication methods:

USERNAME AND PASSWORD

Reducing the number of passwords and usernames users must remember should allow you to focus on establishing better security processes for strengthening the use of the remaining usernames and passwords. For example, you may require a longer alphanumeric for the password, require more frequent changes to it, never allow the username and password to travel in the clear and encrypt passwords in storage with strong security around the encryption keys. Users may be trained to avoid use of obvious passwords and to never leave them written down where others may find them.

A common problem with usernames and passwords is the fact users forget them and call the help desk. This results in significant costs as well as lost productivity. Some web security vendors offer solutions to help the user do self-service to remember their passwords before escalating them up to the help desk. Also, when a help desk does intervene, some vendors offer solutions such that help desk employees can assist the user but still be prevented from actually viewing the passwords.

SMART CARDS AND TOKENS

While steadily gaining ground in the marketplace, smart cards and tokens still suffer from interoperability issues, additional hardware requirements and higher management costs than traditional methods. Smart cards and tokens are often used in multifactor authentication. This means that something like a password and/or username, a biometric or digital certificate is required in addition to the smart card authentication being successful. Smart card and token deployments require careful planning on the logistics of creating, issuing, managing, replacing and terminating the authentication device.

DIGITAL CERTIFICATES

With Microsoft beginning to issue digital certificates for free, it is likely the digital certificate marketplace will both grow and change. However, certificates and the required Public Key Infrastructure (PKI) still have many challenges to overcome. These include the issues of certificate interoperability, costs of the certificates themselves, management costs/complexity in administering certificate solutions and trusting certificates across different domains.

Solutions that should be examined when addressing some of these issues should include consideration of outsourcing some certificate management functions, closer intertwining with identity management and enterprise identity lifecycle processes and seamless integration with web security software.

BIOMETRICS

Biometric authentication methods include retina and iris scans, fingers scans, digital fingerprinting, face recognition, hand geometry, digital signatures and voice recognition amongst others. Until recently, the price points were somewhat prohibitive for large-scale deployments.

Beyond price and hardware costs, you should think about something else. What happens to a person when their biometric data is somehow maliciously used? It's hard to reissue a finger or eyeball for a new biometric authentication.

Biometrics are often used as part of multifactor authentication including things like passwords, smart cards or digital certificates. The web security software should seamlessly allow you to build in multifactor authentication or use of biometrics as a stronger authentication method as the risk increases.

ENHANCED ENTERPRISE WIDE ACCESS CONTROL

If you're like many enterprises, you probably have many forms of access control (identity management, authentication, authorization and auditing) that are not well coordinated and/or connected. This creates the potential for many security lapses as information about identities, their authorization and termination may take a long time to pass between systems and processes.

Additionally, if you examine your labor component to manage the information exchange between systems as well as the cost in lost productivity waiting for access to systems, you'll likely realize enterprise access control not only makes good security sense but business sense as well. By streamlining business processes and automating portions of the identity, authentication and authorization management, you can reduce costs and increase productivity.

The foundation pieces of this are the identity and authentication management already discussed in this paper. As critical, is the need for fine-grained access control.

For example, Jane Doe uses her basic username and password to gain access to the enterprise intranet, networks, e-mail and desktop office applications. When she uses the employee portal to change her address information, the portal and/or the ERP (Enterprise Resource Planning) software perform their own levels of authorization to enable Jane only to what she is entitled to see and use.

When Jane tries to access the enterprise's financial package, a digital certificate is required as further proof of her identity authenticity. When Jane tries to access the enterprise's top-secret materials, she is required to log onto to separate systems using a password, smart card and biometric authentication.

Your web security software has to be able to cope with such a scenario. This means that after a successful authorization the software needs to be able to seamlessly pass Jane to an application like the ERP that controls the authorization. Even while this is going on however, you probably want the web security system to create an end-to-end session audit so you know or, can prove in a court of law, what applications Jane touched from the moment she logged on until she finished.

As importantly, your help desk and others such as the extranet managers will need access to the web security software to interpret why Jane can't access an application when she calls in to complain. Further, the web security software must give you the capability to delegate portions of the identity and/or authorization management to whatever level is required.

For instance, John Smith might be the manager for the enterprise extranet. He in turn may want to delegate portions of the extranet identity management to a business partner so that the business partner has control of which of their employees can get on and off the extranet. The business partner in turn may want to further delegate portions of the extranet identity administration to lower levels, including their own employees. Thus the business partner's employee may alter his or her own address info, telephony or e-mail contact information on the extranet without management assistance.

Further, John may want to let select members of the IT group looking after extranet maintenance have the ability to view or edit certain authorization privileges only within the extranet. However, neither John nor the extranet IT staff may be able to alter authentication or authorization rules for other enterprise applications and resources the web security software is protecting.

This can get even more complicated when a user like Jane get authenticated in one domain (for example her company), and then uses a portal (with a different domain) to access a business partner's application (in yet another domain). The portal and the business partner must be able to:

- Trust an authentication from another party (Jane's company)
- Accept an authorization level from another party (Jane's company or the portal)
- Track end to end audits while Jane moves around within different domains, networks and applications

As mentioned earlier, ensure that your solution is compatible with the SAML (Security Assertion Markup Language) initiatives from OASIS. This will help in seamlessly passing authentication and authorization between parties.

The web security software needs to be able to deal with many different authorization and audit situations. You need to ensure that even though the vendor claims they can do it, the solution will do so securely, easily, at a reasonable cost and scale quickly.

Unfortunately, many solutions are cumbersome to use, complicated, slow to install, hard to maintain without expensive specialized help and scale poorly. The reasons for this may include cumbersome database use for identity management, poor identity control, lack of delegatable control for authorization and identity management, complicated arrangements for configuring servers with the access control solution and hard to understand authorization screens or processes. Caveat emptor!

CONCLUSION

Single Sign On is not a product; it's part of a process. It requires significant thought, planning and the right infrastructure tools if it is to be a success from a security process, business process, cost, management and user perspective.

Sustained management commitment is required to fight some of the battles in the trenches. Getting managers of enterprise systems in various departments to agree to interface with the security solution can be tough. Much education may be required in the areas of LDAP directories, assuring system managers they still control their applications and what modern security practices and standards entail.

Be patient, get the infrastructure right (risk analysis, security process discovery, LDAP directories, web security tools and security process re-engineering), start small and scale quickly. If you do this, your single sign on initiative will not only succeed but also prosper.

USEFUL REFERENCES

[Five Minute CEO/CIO Briefing on Web Security- \(pdf\)](#) or [\(html\)](#)

The first in a series of papers covering web security by Guy Huntington, HVL. This briefing covers seven questions CEO/CIO's should be able to answer to understand web security.

[Reducing Total Cost of Ownership in Web Security- Six Points to Saving Money \(pdf\)](#) or [\(html\)](#)

The second in a series of briefings on web security by Guy Huntington, HVL. This briefing is designed to quickly educate senior managers where to look for cost savings in web security deployments.

[Web Security PowerPoint Presentations \(www.hvl.net/ebusiness.htm\):](#)

Put together by Guy Huntington, [HVL](#) and Derek Small, [Nulli Secundus](#) from their web security, directory and ERP experiences, these presentations cover general and specific issues in web security, identity management, single sign on, LDAP directories, PeopleSoft LDAP integrations and PeopleSoft Oblix integrations.

[“Secrets and Lies: Digital Security in a Networked World”:](#)

Bruce Schneier, CTO of CounterPane and a noted cryptography expert wrote an excellent book covering the issues of digital security. It's a must read for IT managers.

[“SAML” \(Security Assertion Markup Language\):](#)

This is the emerging standard for authentication and authorization. It's being put together by OASIS, the Organization for the Advancement of Structured Information Standards. OASIS is a nonprofit, international consortium that creates interoperable industry specifications based on public standards such as XML and SGML, as well as others that are related to structured information processing.

ABOUT THE AUTHOR

Guy Huntington, President of [HVL](#), specializes in leading enterprise re-engineering projects including B2B's, B2C's and intranet integrating web security and LDAP directory infrastructure into the many disparate systems within each enterprise including data warehouses, ERP's, etc. Guy has lead many projects over the years, authored numerous papers, on-line presentations, an electronic book and given many speeches on the use of the web in business. Guy can be contacted at 604-921-6797, guy@hvl.net or www.hvl.net.