Unleash the Power of Single Sign-On with Microsoft and SAP

White Paper

Authors

Tilo Boettcher, Microsoft Corp (tiloboet@microsoft.com)
Juergen Daiberl, Microsoft Corp (jdaiberl@microsoft.com)
André Fischer, SAP AG (andre.fischer@sap.com)
Lei Liu, Spell GmbH (lei.liu@spell-gmbh.com)

Summary

Single Sign-On (SSO) becomes more and more crucial for today's complex IT landscape driven by the ever-increasing request to integrate existing systems to support cross-organizational business processes. In context of Identity Management, it means to bridge different authentication models used by these systems, which results in high administrative overhead as well as low usability. To enforce a consistent authentication policy throughout the enterprise and to offer simplified and homogeneous logon experience for enterprise users, Microsoft and SAP provide a set of technologies to unleash the power of SSO for enterprises. In this white paper, we review the mainstream enabling technologies for authentication as well as Single Sign-On within the Microsoft/SAP context and outline their usage in some typical scenarios on the enterprise level.

Applies to

- SAP NetWeaver Application Server
- SAP NetWeaver Portal
- SAP GUI
- Microsoft .NET
- Microsoft Office SharePoint Server

Keywords

Single Sign-On, Authentication, Federation, SAML, SAP Logon Ticket, SAP Assertion Ticket, Kerberos, CardSpace

Level of difficulty

Technical consultants, Architects, Developers, IT Managers

Contact

This document is provided to you by SAP and Microsoft to drive interoperability. Please check the .NET interoperability area in the SAP Developer Network (http://sdn.sap.com) or the SAP interoperability section in the Microsoft and SAP Customer Information Center (http://www.microsoft.com/isv/sap) for any updates or further information.

Table of Contents

Introduction

Today's fast paced and ever changing enterprise enables the enforcement of a consistent authentication policy throughout the enterprise. The abilities to integrate with existing systems, to bridge different authentication models, and to offer simplified and homogeneous logon mechanism on desired client platform(s) are the primary goals of such a consistent authentication policy. It is to envision that the further convergence of enterprise systems, e.g. CRM, ERP, etc, towards integrated enterprise-level business applications demands for more convenient security model to enable user authentication. As an efficient way to facilitate simplified user authentication in enterprise, *Single Sign-On* (SSO) refers the mechanisms for enterprise users to authenticate themselves by a single authentication authority once and then gain access to other protected resources without re-authenticating.

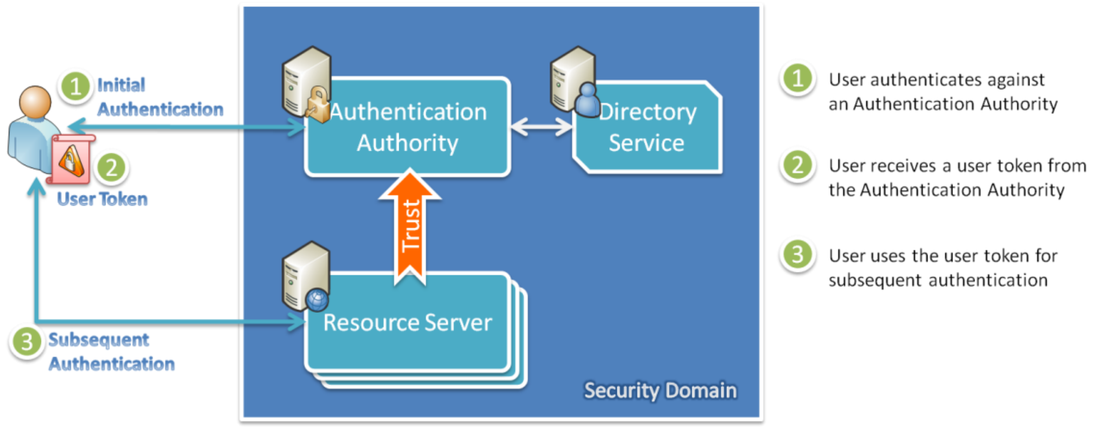
SSO is advantageous for both administrators and enterprise users. A user only needs to deal with a single set of credentials during the daily work, which increases indirectly a user's productivity. For an enterprise administrator, SSO means that all the authentication related information are centralized on a single security service provider, which enforces consistent authentication policy throughout the complete identity management process within the enterprise, including a centralized user repository and a simplified central user management.

In this whitepaper, we will review SSO as a security model and discuss the general security model for SSO in an enterprise environment. Furthermore, we will examine the existing SSO approaches in the SAP/Microsoft context, i.e. how to utilize SSO to access SAP systems from Microsoft platform as well as to access Windows-based applications from inside of SAP. Therefore, the remainder of this whitepaper is organized as follows: the introduction section discusses the basic concept of SSO. The state-of-the-art section reviews the existing approaches enabling SSO in the SAP/Microsoft context while the "SSO Usage Scenario" section shows some typical SSO scenarios from the real world. The last section concludes the whitepaper and provides an outlook concerning the further development of SSO.

The Anatomy of Single Sign-On

Historically, a traditional enterprise computing infrastructure has been assembled from various technical systems that act as independent security domains from each other. Each security domain comprises its own security model for authentication. Enterprise users have to authenticate themselves independently to the domains with which they wish to interact. This legacy security models lead to possible vulnerabilities in enterprise, since enterprise users are forced to deal with a large number of passwords. In this case, users tend to choose simple, easy-to-remember but also easy-to-guess passwords. For enterprise administrators, it means high administration cost due to high administration efforts for maintaining user data in various user repositories.

Overall, the goal of applying Single Sign-On to the enterprise is to reduce the overall administration effort and to enhance the usability of enterprise-level applications while meeting the same security requirements as the conventional authentication mechanisms have. Figure 1 depicts a typical implementation of Single Sign-On with all the essential components in an enterprise, including *users*, *authentication authority*, *resource servers*, *directory service* as well as *security token*. Comparing to the legacy security models mentioned above, SSO is characterized by the *trust*-relationship between *authentication authority* and *resource servers*, as illustrated in Figure 1. The trust-relationship specifies explicitly that the *resource servers* trust in the claims provided by the *authentication server*. In other words, after successful initial authentication, any claims asserted by a trusted *authentication server* are handled as credible by all the related *resource servers*. In this way, the *authentication authority* along with all the *resource servers* establishes a single security domain, within which all the enterprise resources are protected by the single *authentication authority*. A backend *directory service* provides the *authentication authority* with appropriate authentication information via some accessing protocol, e.g. LDAP.



1 Initial Authentication
2 User Token
3 Subsequent Authentication
Authentication Authority
Directory Service
Trust
Resource Server
Security Domain
1 User authenticates against an Authentication Authority
2 User receives a user token from the Authentication Authority
3 User uses the user token for subsequent authentication

Figure 1: General Security Model for Single Sign-On

An essential component in an SSO-enabled infrastructure is *security token* – any digital identity that has been verified by an authentication authority is represented as some kind of security token in the network. Technically, a security token is just some bytes that express some part of total information about an identity, e.g. name, e-mail, department, company, etc. All these information are saved as claims in the security token. Constructing such a security token with a collection of claims depends on the usage scenarios – some simple token may only have a single claim containing the username, such as the *HTTP header variable* containing user IDs for authentication; some other token may contain complex claim sets according to particular functional requirements, such as an *SAP Assertion Ticket*, a *X.509* certificate or a *SAML* token. In this way, security tokens are not focused on authenticating an identity anymore; instead, they can carry some useful information about an identity that can be of interest for the target resource servers. Table 1 summarizes the various components within an SSO infrastructure and maps each of them to sample technologies from the practice.

Furthermore, it is important to figure out that SSO is only a specific form of “authentication” and part of the enterprise-wide identity management infrastructure. Usually, the concept of SSO is confusedly thought as an “access control”-related technology. Comparing to authorization, SSO only defines the process to assure that the current requesting user’s identity is authentic. Although a security token may contains claims necessary for access control, it is not the task of SSO to decide whether a particular user can be granted access to particular resource on the server. Each resource server should have its own authorization mechanism to control access to its resources.

Component	Role	Sample Technologies
Authentication authority	Enterprise users authenticate themselves initially to an authentication authority. For this purpose, users submit their credentials, i.e. a username/password combination, or result of a cryptographic operation involving their credentials, i.e. hash-value of the password, to the authentication authority. It validates the credentials submitted. If the credentials are valid, the user's identity is considered as "authentic" and the authentication authority issues a security token back to the user.	SAP NetWeaver AS Java with User management Engine (UME) or Office SharePoint Server with integrated Windows authentication
Directory service	The directory service stores either a copy of user credentials or the result of some cryptographic operation based on the credentials. The authentication authority access the directory service via some predefined protocols, such as LDAP.	Active Directory, OpenLDAP,
Resource server	The resource server provides services to the end users. In general, a resource server does not feature its own authentication mechanism; instead, it uses the security token issued by the authentication authority as a proof of authentication in subsequent access.	SAP NetWeaver AS ABAP or a simple ASP .NET application
Security token	To prove that a user has been authenticated, the authentication authority issues a cryptographic security token to the user. These issued security token are used in the subsequent access to simplify the authentication process at the resource servers.	SAP Logon Ticket, Kerberos Ticket, X.509 Certificate, or SAML token

Table 1: Overview of Technical Components in an SSO Infrastructure

Enabling Technologies for SSO

In this section, we will review various enabling technologies to support SSO in Microsoft/SAP context. In the following, we will review at first the authentication infrastructure on both platforms. After this, we will go on reviewing various enabling technologies for SSO and checking how they can work together to access backend systems on Microsoft/SAP platforms.

Authentication Infrastructure

Both Microsoft and SAP provide a set of authentication mechanisms in their product line. Based on functional- and non-functional requirements, on usage scenarios, and existing security infrastructure, these authentication methods provide support to ensure different security level for user authentication.

Authentication on Windows Platform

As the major application server on the Windows platform, Internet Information Server (IIS) provides a set of build-in support for user authentication: Anonymous Authentication, Basic Authentication, Digest Authentication, Certificate Authentication, Integrated Windows Authentication as well as .NET Passport Authentication[1]. Furthermore, for ASP.NET applications such as Microsoft Office SharePoint Server (MOSS) running on IIS, ASP.NET also allows to involve external mechanisms for user authentication, such as to authenticate users via external LDAP server. In this way, one can delegate authentication to an external mechanism. Similarly, Windows Communication Foundation (WCF) provides a "plug-in" mechanism to enable external user authentication. Table 2 provides some further reading about authentication on Windows platform.

Summary: Authentication on Windows Platform	
Further Information	Microsoft TechNet: Authentication Methods Support in IIS 6.0 (http://www.microsoft.com/technet/prodtechnol/WindowsServer2003/Library/IIS/523ae943-5e6a-4200-9103-9808baa00157.mspx?mfr=true) Authentication in ASP.NET: .NET Security Guidance (http://msdn2.microsoft.com/en-us/library/ms978378.aspx) MSDN Magazine – Security Briefs: Security in Windows Communication Foundation (http://msdn.microsoft.com/msdnmag/issues/06/08/securitybriefs/default.aspx) MSDN: WCF – Custom Credentials and Credential Validation (http://msdn2.microsoft.com/en-us/library/ms734697.aspx)

Table 2: Overview - Authentication on Windows

Authentication on SAP Platform

SAP provides also a set of possible authentication methods on the SAP platform. Figure 2 illustrates the general architecture of SAP NetWeaver Application Server. A SAP NetWeaver AS exposes functionalities through two interfaces: an interface exposed by the *ABAP Engine* to enable the classical access through SAP GUI and a second Web-based interface exposed by the *Internet Communication Manager* (ICM). Outwards, ICM establishes a Web-based connection by supporting various Internet protocols like HTTP, HTTPs, SOAP as well as SMTP. Internally, the ICM dispatches incoming requests to various dispatchers of the *J2EE Engine* or the *ABAP Engine*.

[1] More information about various authentication mechanisms on IIS and how to configure them is available under: http://www.microsoft.com/technet/prodtechnol/WindowsServer2003/Library/IIS/523ae943-5e6a-4200-9103-9808baa00157.mspx?mfr=true

Based on the different ways in which an SAP NetWeaver AS is accessed – either Web-based or through SAP GUI, there are different mechanisms that can be applied to enable SSO.

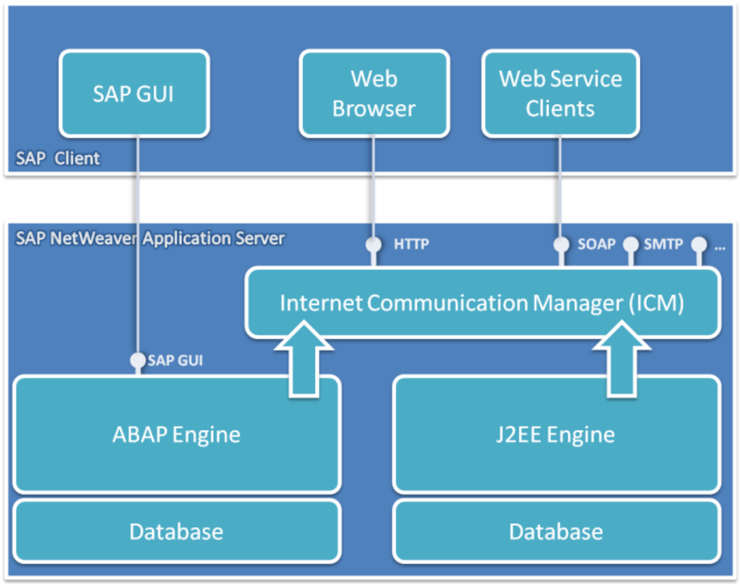


Figure 2: SAP NetWeaver Application Server Architecture[2]

Both ABAP Engine and J2EE Engine support a set of internal as well as external authentication methods. As standard authentication methods on SAP platform, both of them support user authentication on the base of User ID/Password, SAP Logon Tickets, and X.509 Client Certificate. Additionally, both of them also provide programming APIs to include external authentication mechanisms into existing security infrastructure on the SAP platform.

Pluggable Authentication Services (PAS)

As an approach to enable external authentication in combination with the SAP *Internet Transaction Server* (ITS), PAS provides a programming API interface allowing developers to plug in third-party authentication into existing SAP landscapes. Figure 3 illustrates the concept of PAS with an external ITS. Users submit their credentials directly to a Web server with an enabled ITS-WGate. Now, the external user authentication can be performed on either of the ITS's component – AGate or WGate via PAS. The PAS informs the ITS about the authentication result. If the credentials submitted by the user are valid, the ITS issues an SAP Logon Ticket to the user for subsequent authentication.

With PAS, it is possible to integrate existing security infrastructure into SAP systems. The WGate-Variant supports user authentication based on NTLM, X.509 Client Certificate as well as any mechanisms that can set a User ID as an HTTP header variable. The AGate-Variant supports any mechanisms that can authenticate User ID/Password, for instance against a domain controller or an external LDAP server. However, PAS requires the availability of an external ITS for operation. Since an external ITS is only available on SAP Web Application Server 6.20 (or lower) and meanwhile ITS becomes an integral part of SAP NW AS, PAS interface is not supported any more by the SAP application server from SAP NW 2004 and up. Table 3 summarizes PAS and provides further reading for it.

[2] Adapted from SAP Library:
http://help.sap.com/saphelp_nw70/helpdata/en/84/54953fc405330ee10000000a114084/frameset.htm

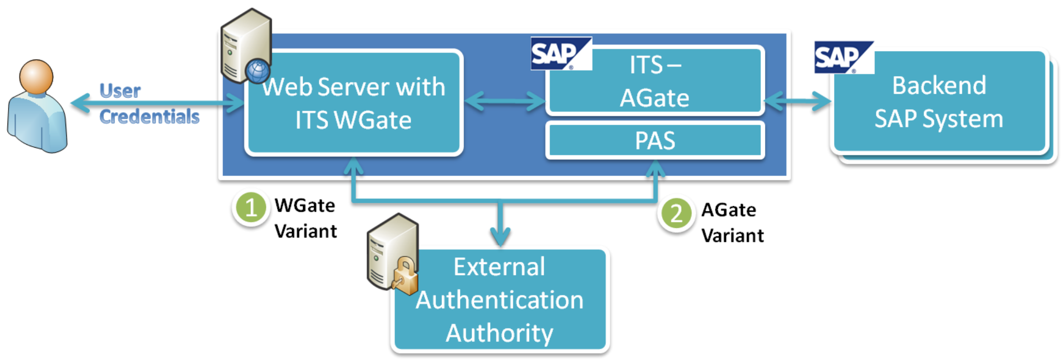


Figure 3: External Authentication with Pluggable Authentication Service (PAS)

Summary: Pluggable Authentication Service (PAS)	
Overview	PAS in combination with an external ITS allows integrating external authentication mechanisms into existing SAP landscape.
Availability	Only supported by SAP Web Application Server ABAP 6.20 or lower
SSO	PAS provides a possibility to perform initial user authentication
Further Information	SAP Library: Pluggable Authentication Service for External Authentication Mechanisms (https://www.sdn.sap.com/irj/servlet/prt/portal/prtroot/docs/library/uuid/e7e89032-0a01-0010-a397-926f70359db3)

Table 3: Pluggable Authentication Service in a Nutshell

Java Authentication and Authorization Service (JAAS)

Defined as part of the Java SE Development Kit (JDK) standard[3], JAAS contains a set of APIs that allows Java applications performing authentication and authorization upon users in a pluggable fashion. Similarly to the PAS interface for SAP ITS, integrating JAAS into SAP NetWeaver AS Java Engine allows Java applications to be independent from the underlying authentication mechanisms – new or updated authentication mechanisms can be plugged in to existing applications without any modification on them.

The authentication functionality is organized in a "Login Module", which implements a set of predefined methods, such as *initialize*(), *login*(), *commit*() and *abort*()[4]. On SAP NW AS Java Engine, various login modules are bundles into "login module stacks", which can contain one or more login modules with different JAAS control flags attached to each module. The JAAS control flags defines the priority of each login module in the stack and may vary from "required", "requisite", "sufficient" to "optional". The overall authentication result of the complete login module stack depends not only on the authentication result of a particular authentication module in the stack, but also on the priority of particular module's control flag. To perform authentication at runtime, all the login modules in the stack are processed. Therefore, a "failure" in a "required" login module leads to a "failure" of the overall login module stack at once, while a "failure" in an "optional" login module may not cause the overall login process to

[3] JAAS is originally introduced as an optional package for J2SE 1.3. Since J2SE 1.4, JAAS has been integrated into JDK. Further information about JAAS is available under http://java.sun.com/products/jaas/.

[4] Please refer to the technical documentation on Sun Developer Network for more information about how to implement JAAS Login Module: http://java.sun.com/j2se/1.4.2/docs/guide/security/jaas/JAASRefGuide.html .

fail, if the credentials submitted can pass another login module in the stack with control flag of higher priority, e.g. “requisite”[5].

Meanwhile, SAP provides a number of predefined login modules for J2EE Engine. Next to some standard login modules such as *BasicPasswordLoginModule*, or *EvaluateTicketLoginModule*, there are also login modules for external authentication, such as *DigestLoginModule* or *SPNegoLoginModule*. The *Simple and Protected GSS API Negotiation Mechanism (SPNego)* enables the SAP Java NetWeaver AS to negotiate Kerberos authentication with Web clients, such as Web browsers, thus allowing Windows Integrated Authentication.

Another noteworthy login module is the *CreateTicketLoginModule*, which can be used to issue SAP Logon Tickets after successful authentication. With this login module, the J2EE Engine can issue SAP Logon Tickets for other SAP Systems, such as for an ABAP Engine. In other words, the J2EE Engine can be used as an “authentication authority” to any ABAP-based applications, if an external authentication method that is not supported by the ABAP Engine has to be used in the scenario. Actually, this is also a typical scenario for SSO with the J2EE Engine as the authentication authority for the initial authentication. Table 4 provides a summary of JAAS.

Summary: Java Authentication and Authorization Server (JAAS)	
Overview	JAAS is a JDK standard for integrating external authentication methods into existing Java application in a “pluggable” fashion. SAP NW AS Java Engine implements/extends this concept to allow using existing authentication infrastructure for initial authentication.
Availability	SAP NetWeaver Application Server Java Engine (as well as SAP Web Application Server Java)
SSO	JAAS provides a possibility to perform initial user authentication by using external authentication mechanisms.
Further Information	Sun Developer Network: Java Authentication and Authorization Service (JAAS) Overview (http://java.sun.com/products/jaas/overview.html) Sun Developer Network: Java Authentication and Authorization Service (JAAS) Reference Guide (http://java.sun.com/j2se/1.4.2/docs/guide/security/jaas/JAASRefGuide.html) Sun Developer Network: Java Authentication and Authorization Service (JAAS) LoginModule Developer’s Guide (http://java.sun.com/j2se/1.4.2/docs/guide/security/jaas/JAASLMDevGuide.html) SDN: Authentication for Web Application Users on the J2EE Engine (http://help.sap.com/saphelp_nw70/helpdata/en/b9/9482887ddb3e47bd1a738c3e900195/frameset.htm) SDN: Authentication on J2EE Engine (http://help.sap.com/saphelp_nw04s/helpdata/en/b1/07dd3aeedb7445e10000000a114084/frameset.htm)

Table 4: Java Authentication and Authorization Service (JAAS) in a Nutshell

[5] SDN provides detailed description about the various control flags. You can find a sample there to see how the overall authentication result can be calculated from the results of each login module in the stack: http://help.sap.com/saphelp_nw04s/helpdata/en/b1/07dd3aeedb7445e10000000a114084/frameset.htm.

Single Sign-On for Web-Based Scenarios on SAP

In a Web-based scenario within SAP context, SAP NetWeaver AS is always somewhere involved – either as an authentication authority or as a resource server. To accelerate SSO in such scenarios, SAP NetWeaver AS provides a set of mechanisms for authenticating users. Among all these mechanisms, some of them are only available on SAP NW AS Java while some other ones are available on both SAP NW AS Java as well as SAP NW AS ABAP; some ones are only in effect for initial authentication while some other ones can be used for both initial authentication as well as SSO after that. In the following, we will review all the mechanisms one after another and try to provide a clear picture about the SSO-related technologies discussed in this section within the SAP/Microsoft context.

User Authentication

User authentication aims to ensure a user's identity before granting that user further access to resources. In order to check a user's identity, that user has to provide his credentials, for instance in form of user ID/password or an X.509 Ticket, to the authentication authority. The authentication authority verifies the submitted identity and, if applicable, grant that user access to the backend system, including SAP and Non-SAP systems. Both Microsoft and SAP provide a set of authentication methods in their product line, as described in this following section.

User ID/Password based Authentication

Using credential sets consisting of user ID and password is an often-used and simplest approach for authenticating users. Generally, a user submits his user credentials - user ID and password - via browser or other client applications to the destination application to authenticate himself. User credentials are transferred e.g. via HTTP(s) protocols to the application server. At the server side, the application server compares the values submitted by the user with the values stored in the database or in the directory server. To enhance the usability and significance level of this approach in practice, passwords are often encrypted by building their hash values before they are saved to the database. The irreversible nature of hash algorithms ensures that only the user knows the password. Even administrators with access to the database cannot retrieve user passwords from hash value stored there.

Both Microsoft and SAP support User ID/Password based authentication in their Web-based products. In practice, several authentication schemes support this authentication method:

- *Basic Authentication*: originally defined by a set of *Request for Comments* (RFCs)[6] within the context of HTTP, Basic Authentication defines the standard interaction between browser and a HTTP server – e.g. IIS Server or SAP NW AS – to allow browsers providing credentials in the form of a user ID and password when making a request. This means that users will be prompted to enter their credentials in browsers' standard pop-up windows before they can be granted with access to an application protected by Basic Authentication. Meanwhile, almost all popular browsers support this authentication scheme. However, it is rarely used in Internet-based Web applications but often in intranet-based Web applications due to the limited customizability of the pop-up window in browsers.

[6] The basic authentication scheme is defined within the RFC 1945 for Hypertext Transfer Protocol – HTTP/1.0, RFC 2616 for Hypertext Transfer Protocol – HTTP/1.1 as well as RFC 2617 for HTTP Authentication with Basic and Digest Access Authentication.

- *Form-based Authentication*: Comparing to Basic Authentication, Form-based authentication provides more possibilities to customize the logon interface based on other functional requirements for authentication, for instance, a logon interface with corporate design or users have to provide some other information next to the standard user ID and password about themselves, such as email address or corresponding department ID. In both Basic Authentication as well as Form-based Authentication, login information is transferred as plaintext via the Internet. Therefore, it is recommended to use an encrypted communication between clients and application servers by applying SSL to the connection.
- *Digest Access Authentication*: to prevent man-in-the-middle attack[7] between browser and application server, even if an encrypted communication path does not exist between them, Digest Access Authentication based on Basic Authentication is introduced by RFC 2617 of the IETF[8]. It improves the authentication process by allowing browser to build a so-called "Digest", which is the MD5 hash value of the login information submitted by end users, directly at the client side. In this way, sensible information such as password is not transferred as plaintext via the Internet any more. Instead, a digest calculated out of a set of values that are only known to client browser and application server, for instance, user ID, password, security realm, etc, is transferred between them.

Briefly, authentication process based on User ID/Password is normally the default mechanism for user authentication for many application (portal) servers, regardless of the frontend browser used at the client side. It is easy to implement for administrators and easy to use for end users. The only potential security concern is to prevent the man-in-the-middle attack along the communication path during an authentication process. In this case, it is recommended to use encrypted communication ways with HTTPS between browser and application servers. Table 5 summarizes this authentication method.

[7] In network security, a man-in-the-middle attack refers to an attack, where an attacker can read, modify, replace as well as fake messages between two communication partners, without knowledge of either of them. Generally, this type of attack can be prevented by using some additional mechanism on the transport layer, such as digitally signing the messages with the sender's private key and verifying them with the sender's public key.

[8] RFC2617 – HTTP Authentication: Basic and Digest Access Authentication. Available under http://tools.ietf.org/html/rfc2617

Summary: User Authentication using User ID/Password	
Overview	Allowing end users to provide their user ID and passwords directly in the browser application to authentication themselves against an application server. This approach is supported by nearly every Internet-enabled application server, such as SAP NW AS or MOSS
Consideration	Since submitted credentials (except Digest Access Authentication) are transmitted as plaintext directly via the Internet, it is strongly recommended to establish an encrypted communication path between client browser and servers by applying SSL to it. This procedure can effectively avoid the man-in-the-middle attacks on the communication path.
Availability	*Basic Authentication* - Microsoft SharePoint Portal Server 2003/2007, SAP NetWeaver Application Server Java (BasicPasswordLoginModule as JAAS login module) *Form-based Authentication* - Microsoft SharePoint Portal Server 2003/2007, SAP NetWeaver Application Server Java (BasicPasswordLoginModule as JAAS login module), SAP Web Application Server ABAP 6.20 and higher releases *Digest Access Authentication* - Microsoft SharePoint Portal Server 2003/2007, SAP NetWeaver Application Server Java (DigestLoginModule as JAAS login module)
SSO	Authentication based on User ID/Password often serves as initial user authentication in SSO, since it is supported by most of application servers as well as popular browsers on the client side.
Further Information	*Basic Authentication –* SAP Library: Using Basic Authentication (http://help.sap.com/saphelp_nw70/helpdata/en/f3/a193e2f6ee1b45ac2e386468d3c272/frameset.htm) SAP Library: Logon and Password Security in the SAP System (http://help.sap.com/saphelp_nw04s/helpdata/en/52/6717ed439b11d1896f0000e8322d00/frameset.htm) SAP SDN Document: Securing Web Applications by Basic Authentication (https://www.sdn.sap.com/irj/servlet/prt/portal/prtroot/docs/library/uuid/4ecd6eb1-0301-0010-61bd-a3d6bee102c0) Microsoft TechNet: Configure Web Site authentication (http://technet2.microsoft.com/Office/ja-JP/library/0edcb1af-6418-4c65-ab30-602c0c7f85951041.mspx?mfr=true) *Form-based Authentication –* SAP Library: Authentication for Web Applications Users on the J2EE Engine (http://help.sap.com/saphelp_nw04/helpdata/en/b9/9482887ddb3e47bd1a738c3e900195/frameset.htm) MSDN: Forms Authentication Provider (http://msdn2.microsoft.com/en-us/library/9wff0kyh.aspx) *Digest Access Authentication –* SAP Library: Authentication for Web Applications Users on the J2EE Engine (http://help.sap.com/saphelp_nw04/helpdata/en/b9/9482887ddb3e47bd1a738c3e900195/frameset.htm) Microsoft TechNet: Configure digest authentication (http://technet2.microsoft.com/Office/en-us/library/45daf889-d7cc-4527-8147-e537e5d214ac1033.mspx?mfr=true)

Table 5: User ID/Password based Authentication in a Nutshell

HTTP Header Variable Authentication

In HTTP Header Variable authentication, the application server does not authenticate user's credentials. Instead, this task is "outsourced" to an intermediate authentication authority called "Web Access Management" (WAM) product. In other words, initial authentication takes place outside backend systems. As illustrated in Figure 4, users submit their request together with their credentials to the intermediate authentication authority. The authentication authority validates these user credentials and forwards the HTTP request to the backend resource servers by adding the user ID of the authenticated user to the current HTTP request header. The backend resource server compares the user ID with values stored in its own user database and grants the user access to the requested resource(s) if it can find a match in the database.

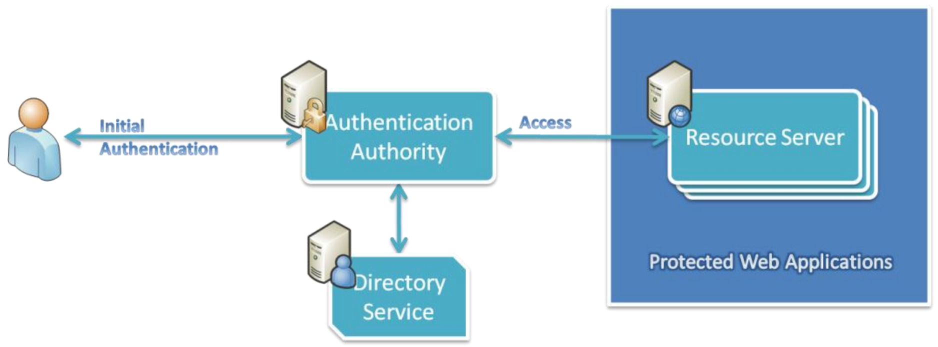


Figure 4: HTTP Header Variable Authentication Overview

HTTP Header Variable authentication is a simple but flexible approach to perform user authentication in combination with an existing Web Access Management infrastructure, which applies to both SAP and non-SAP Web servers. However, there are some security concerns to take into account during the implementation. Because backend Web server does not perform the actual authentication process but only verifies the validity of the HTTP header in the request, it is easy to bypass the intermediate authentication authority by sending the HTTP request with an HTTP header directly to the backend server. Therefore, the communication path between the intermediate authentication authority and the backend Web servers must be secured by enabling mutual authentication between them. Furthermore, the backend Web server must be configured to only accept HTTP request from the intermediate authentication authority and deny any other illegal requests directly from client applications. Table 6 summarizes the HTTP Header Variable authentication.

Summary: User Authentication using HTTP Header Variable	
Overview	HTTP Header Variable authentication uses an external WAM product as intermediate authentication authority and transmits the user's identity in a HTTP header variable to the backend systems.
Consideration	Communication path between intermediate authentication authority and the backend systems must be secured by applying mutual authentication, for instance, with IP-based access control, to avoid illegal direct access to backend systems by bypassing the intermediate authentication authority.
Availability	SAP NW AS Java (HeaderVariableLoginModule as JAAS login module)
SSO	HTTP Header Variable authentication is not only a method for user authentication. It also exhibits the characters of SSO to enable subsequent authentication to a set of backend systems that supports HTTP header authentication, which is a simple but flexible approach to allow access to both SAP and non-SAP systems.
Further Information	SAP Library: Using Header Variables for User Authentication (http://help.sap.com/saphelp_nw04s/helpdata/en/f3/a193e2f6ee1b45ac2e386468d3c272/frameset.htm) SAP Library: Adjusting the Login Module Stacks for Using Header Variables (http://help.sap.com/saphelp_nw04s/helpdata/en/68/5ddc40132a8531e10000000a1550b0/content.htm)

Table 6: HTTP Header Variable Authentication in a Nutshell

X.509 Client Certificate Authentication

X.509 Client Certificate Authentication refers to the authentication process, during which user submit a X.509 certificate instead of user ID/Password as their credentials to backend servers. A backend server verifies the validity of incoming client certificates and checks out if the user information contained in the certificate matches any entry in its user database. In case that an entry with the same user ID is found in the database, this user is logged on to the corresponding system.

Generally, a X.509 client certificate is a digitally signed statement that contains information about an entity. Only trusted organization called Certificate Authority (CA) can generate a public/private key pair, where public key together with information about the entity, such as name, organization, etc., is incorporated into a certificate. The corresponding private key to the certificate must be safeguarded by the entity as the certificate owner. A X.509 client certificate is signed by the CA's private key during its creation process, which ensures that anyone that trusts the CA and has access to the CA's public key can verify the authenticity of an X.509 client certificate. This process establishes the foundation for client certificate based authentication and is referred as part of the "Public Key Infrastructure" (PKI)[9] in practice.

During the initial authentication process with certificates, each party authenticates the other one by verifying the authenticity of the corresponding certificate from the other party. The prerequisite for this authentication method is that users have imported the valid client certificate distributed by PKI into their browser application. On the server side, Web servers must support HTTPs and SSL protocols and have a valid certificate distributed by the CA.

[9] More information about Certificate Authority as well as Public Key Infrastructure can be found under http://www.microsoft.com/windowsserver2003/technologies/pki/default.mspx.

Furthermore, both browser and Web server have valid copies of the issuing CA’s certificate (public key).

At the beginning of the mutual authentication process, a client browser sends the server a request to indicate that it wishes to mutually authenticate with the server. After that, both parties exchange their certificates. Each party verifies the authenticity of the certificate by decrypting the certificate with the CA’s public key. Once both parties are satisfied with the identity of the other party, they step into the next phase to establish a secure connection between the client browser and the server by negotiating a session key for the connection. Till now, the mutual authentication is completed. Oftentimes, for example, within the context of SAP NW AS, the Web server has to verify the identity contained in the client certificate in another step to check, if the distinguished name in the certificate matches any entry in the server’s user database. If there is a match, the end user is logged on to the server with his distinguished name.

Compared to the other approaches, client certificate authentication ensures a secure connection between client and server with strong authentication scheme. Users are not required to provide both user ID and password any more. Instead, they only need to provide their client certificate for authentication. Furthermore, client certificate authentication provides a two-way authentication of server and client at the same time. However, client certificate authentication requires more administration efforts in comparison with other approaches. At first, it requires the availability of a PKI either internally or externally to enroll, issue, distribute and verify certificates. Further, it requires on the client side that the browser application must support mutual authentication on the protocol level. On the server side, the Web server must also be configured to support mutual authentication in combination with SSL/TLS. Furthermore, this approach does not support delegation of security credentials. In other words, the Web server cannot forward a client certificate to any other backend system to enable subsequent SSO. Table 7 summarizes the X.509 client certificate based authentication.

Summary: User Authentication using X.509 Client Certificate	
Overview	X.509 Client Certificate Authentication provides a strong authentication scheme by applying mutual authentication to both client and server. It requires the availability of a PKI to work.
Consideration	This authentication method provides secure user authentication on the highest level. However, it requires more administration efforts and is cumbersome to configure. Both client and server must support mutual authentication with client/server certificates. Furthermore, this approach does not support delegation of security credentials.
Availability	Microsoft Internet Information Server (IIS) SAP NetWeaver AS Java/ABAP (ClientCertificateLoginModule as JAAS login module)
SSO	This approach is suitable for both initial authentication as well as subsequent SSO to any systems that supports client certificate authentication.
Further Information	SAP Library: Using X.509 Client Certificates (http://help.sap.com/saphelp_nw04s/helpdata/en/b1/07dd3aeedb7445e10000000a114084/content.htm) MSDN: Building Secure ASP.NET Application – Authentication, Authorization and Secure Communication (http://msdn2.microsoft.com/en-us/library/aa302412.aspx) Microsoft TechNet: Plan Authentication Methods (Office SharePoint Server) (http://technet2.microsoft.com/Office/en-us/library/40117fda-70a0-4e3d-8cd3-0def768da16c1033.mspx?mfr=true) MSDN: IIS Authentication (http://msdn2.microsoft.com/en-US/library/aa292114(VS.71).aspx)

Table 7: X.509 Client Certificate Authentication in a Nutshell

Integrated Windows Authentication

With Integrated Windows Authentication (IWA), the authentication task is delegated to Windows platform with NTLM or Kerberos based on Windows Domain Controller. The basic idea is to leverage the logon information on the Windows operating system for authentication on backend application servers.

In the past SAP supported IWA through the IISProxy Module. The IIS Proxy module was developed as an ISAPI filter for Microsoft Internet Information Server (IIS). At runtime, if an end user sends a request to an IIS with IIS Proxy module enabled, the IIS Proxy module reuses the user authentication information from the current Windows session and forwarded this information as additional HTTP header together with the original HTTP request to the backend SAP Java NetWeaver AS that was configured to accept HTTP Header Variable Authentication. However, SAP has stopped supporting the IIS Proxy module December 2006[10].

The current more comprehensive and platform-independent approach to support IWA by SAP is the SPNegoLoginModule. Originally, SPNego refers to "**S**imple and **P**rotected GSSAPI **Nego**tiation" mechanism[11], which defines a protocol for communication partners to determine at runtime which GSSAPI mechanism (e.g. a dominant implementation of GSSAPI is Kerberos)

[10] Please refer to SAP Note 886214 "End of Maintenance of IISProxy ISAPI module" for more information.
[11] SPNego is defined by IETF as industry standard with the following two RFCs: RFC 2478 (http://tools.ietf.org/html/rfc2478) as well as RFC 4178 (http://tools.ietf.org/html/rfc4178).

should be used to secure the communication context between them. In this approach, SPNego is used as wrapper for underlying authentication mechanism.

Figure 5 illustrates a simplified authentication scenario for IWA with SPNego in the context of SAP. One of the prerequisites in this scenario is the availability of Windows Active Directory as Domain Controller, because SPNegoLoginModule uses Kerberos as the underlying authentication mechanism, which needs Active Directory as Key Distribution Center (KDC) (also called as Ticket Granting Server (TGS) in Kerberos) to issue session tickets. It is assumed that as starting point, the user has logged on to the Windows workstation with his credentials. To start the authentication process, the browser simply requests a resource on the J2EE engine by sending the initial HTTP Get verb to the J2EE engine. The J2EE engine with SPNego enabled requires authentication and issues a "401 Request Denied" response with "WWW-Authenticate: Negotiate" in the HTTP header back to the client browser. This response informs the client browser that the J2EE engine requested belongs to a Kerberos realm. To establish a security context with the J2EE engine, the browser sends a request to the AD (acting as KDC) to get a Kerberos Client/Server Session Ticket for the session. AD supplies the browser with the necessary Kerberos Session Ticket wrapped in a SPNego token (assuming that the user is authorized to consume services on the J2EE engine). Then, the client browser resends the HTTP request together with the SPNego token containing the Kerberos Session Ticket in a HTTP header "Authorization: Negotiate ...[token block]..." to the J2EE engine. To verify the SPNego token, the SPNegoLoginModule uses the Kerberos implementation of the underlying Java SDK to check the validity of the token. If the user is authenticated, the J2EE engine responds the client browser with the requested resource.

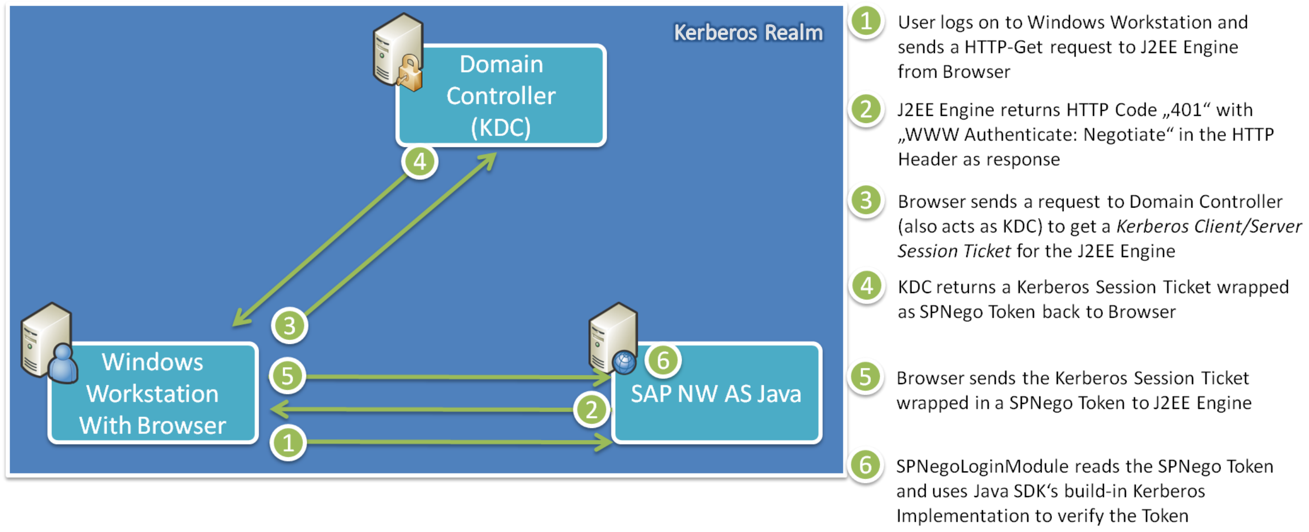

Kerberos Realm
Domain Controller (KDC)
Windows Workstation With Browser
SAP NW AS Java
1
2
3
4
5
6
1 User logs on to Windows Workstation and sends a HTTP-Get request to J2EE Engine from Browser
2 J2EE Engine returns HTTP Code „401“ with „WWW Authenticate: Negotiate“ in the HTTP Header as response
3 Browser sends a request to Domain Controller (also acts as KDC) to get a *Kerberos Client/Server Session Ticket* for the J2EE Engine
4 KDC returns a Kerberos Session Ticket wrapped as SPNego Token back to Browser
5 Browser sends the Kerberos Session Ticket wrapped in a SPNego Token to J2EE Engine
6 SPNegoLoginModule reads the SPNego Token and uses Java SDK's build-in Kerberos Implementation to verify the Token

Figure 5: Integrated Windows Authentication with SPNego[12]

Compared to the IIS Proxy ISAPI module, SPNego provides a platform-independent approach to leverage Windows logon information. While IIS Proxy still requires IIS as the intermediate middleware and thus separate physical server, SPNegoLoginModule can be deployed on all possible platforms, such as UNIX on the Java NetWeaver AS itself. Furthermore, in comparison with other authentication methods introduced afore, IWA using SPNego is the best

[12] Figure adapted from MSDN article "HTTP-Based Cross-Platform Authentication via the Negotiate Protocol – Network Infrastructure" (http://msdn2.microsoft.com/en-us/library/ms995329.aspx)

authentication scheme in the Intranet, if users are using Windows Domain accounts in combination with Kerberos for logon. With IWA, users can access the J2EE engine without re-typing their passwords for authentication. Table 8 provides a summary for IWA with SPNegoLoginModule.

Summary: User Authentication using SPNegoLoginModule	
Overview	SPNegoLoginModule uses SPNego to negotiate shared authentication mechanism (in this case Kerberos) between client browsers and J2EE engine. This allows a platform-independent authentication scheme for intranet-based scenarios.
Consideration	SPNego requires the availability of Windows Kerberos environments with AD. On the client side, browser applications have to be configured to use IWA. On the server side, J2EE engine must be configured to use SPNego. If the J2EE engine runs on a non-windows platform, it is necessary to integrate that server component into the existing Windows Kerberos infrastructure. Furthermore, SPNego does not provide any mechanisms to ensure transport layer security, therefore, it is recommended to enable SSL between browser and J2EE engine.
Availability	SAP NetWeaver AS Java (as a JAAS Login Module)
SSO	This approach can perform initial authentication based on IWA.
Further Information	MSDN: HTTP-Based Cross-Platform Authentication via the Negotiate Protocol (Network Infrastructure, SPNego Tokens and the Negotiate Protocol, SPNego Token Handler API) (http://msdn2.microsoft.com/en-us/library/ms995329.aspx) SAP Library: Using Kerberos Authentication for Single Sign-On (http://help.sap.com/saphelp_nw04s/helpdata/en/76/55ee41c334c717e10000000a155106/frameset.htm) SAP Note: 994791 – Wizard-based SPNego configuration[13]

Table 8: Integrated Windows Authentication with SPNego in a Nutshell

Authentication based on SAP Logon Ticket

SAP Logon Ticket is a SAP's proprietary solution for SSO. In fact, SAP Logon Ticket is a security token that can be used for user authentication. To obtain a SAP Logon Ticket for authentication, users have to authenticate themselves against a so-called "Ticket Issuing Server" – for instance, SAP NW AS Java engine – by using some authentication mechanism. After a successful initial authentication, users are issued SAP Logon Tickets in the form of non-persistent cookies with the name "MYSAPSSO2" in their browser session. In addition to the user ID of the current user, the SAP Logon Ticket issued contains also the authentication scheme, validity period, issuing system as well as the digital signature of the issuing system. This ticket remains valid for successive calls not only to the ticket-issuing server but also to other systems that accepts SAP Logon Ticket from the particular ticket-issuing server, unless the user logs off or the validity period of the logon ticket exceeds.

For successive calls to other systems in the landscape, the browser simply sends the HTTP request together with the SAP Logon Ticket to the target system. To authenticate the user, the target system has to check the validity of the ticket by verifying the digital signature of the ticket with the issuing system's public key. This also ensures that the ticket was issued by a trusted ticket-issuing server. In the next step, the target system verifies the validity period of the ticket

[13] Login required for accessing this note on SAP

to ensure that it is not expired. If the verification is successful in the previous steps, the current user is granted access to the target system, whose ID is contained in the logon ticket.

Currently, SAP Logon Ticket is the most recommended way to facilitate SSO in a complex and heterogeneous system landscape. Both SAP NW AS ABAP Engine as well as SAP NW AS Java Engine uses SAP Logon Ticket in combination with an arbitrary authentication method supported for SSO. For Windows-based applications, SAP Logon Ticket can also be used for SSO with them. For these purpose, SAP provides a set of components as well as programming libraries to integrate SAP Logon Ticket into external Windows applications to enable user authentication for SSO:

- Web Server Filter for IIS 5.0 on Windows 2000 and IIS 6.0 on Windows 2003: this Web Server Filter operates as a “token”-converter for Windows-based applications running on IIS– the ISAPI filter verifies information contained in an SAP Logon Ticket, then extracts the user ID out from the logon ticket and forwards the user ID in an additional in the HTTP header variable “*remote_user_alias*” with the original HTTP request to the target application. Therefore, the target application should support HTTP Header based authentication to consume the identity information delivered in the HTTP header. However the development of this filter has ended and therefore there will be no support for future versions of IIS and Windows Server.[14]
- Web Server Filter with Delegation for Windows Server 2003: as Microsoft introduced “Kerberos Protocol Transition and Constrained Delegation”[15] as extension to the existing Kerberos authentication protocol in Windows Server 2003, SAP introduced a new ISAPI module “SSO22KerbMap” to allow mapping authentication information from SAP Logon Ticket to Kerberos Ticket for the target application. Generally, this new feature of the Microsoft Kerberos implementation allows applications to use non-windows authentication method – in this case SAP Logon Ticket – to authenticate user and then use Kerberos authentication and delegation to access downstream Windows-based applications. To initialize the authentication process, a client browser calls an application on IIS with a SAP Logon Ticket in the request. The ISAPI module verifies the validity of the SAP Logon Ticket submitted by the client browser. Afterwards, if the SAP Logon Ticket is valid, the ISAPI module extracts the user ID from the logon ticket and verifies it in the backend AD. If the user ID is valid, the ISAPI module performs then impersonation by acquiring a *constrained Kerberos Ticket* from AD. Following access to backend applications is then carried out via Kerberos authentication with the issued constrained Kerberos Ticket.
- DLL “SAPSSOEXT” for verifying SAP Logon Tickets: the last approach is to perform ticket verification by applications themselves. For this purpose, SAP provides the “SAPSSOEXT” DLL for Windows-based application. This library encapsulates a set of operations to handle SAP Logon Tickets, so that developers can easily integrate the ticket verification and accepting capabilities into their own applications.

[14] See also SAP Note 442401 - Web server filter for SSO to third-party systems

[15] More information about Windows Server 2003 support for Kerberos Protocol Transition and Constrained Delegation is available under: http://technet2.microsoft.com/windowsserver/en/library/0d3a0555-9a68-4cc6-ad55-fd076d037fcf1033.mspx?mfr=true

Overall, with the components provided by SAP for Windows-based Web applications, SAP Logon Ticket delivers the basis for an integrated Web-based SSO environment across SAP and Microsoft. Compared to X.509 Client Certificate Authentication, which also provides a cross-platform authentication as well as SSO support to Web applications, this approach requires fewer administration efforts and is therefore more efficient. Furthermore, authentication based on SAP Logon Ticket demands fewer cryptographic operations than Certificate-based authentication, and additional mechanisms such as caching valid SAP Logon Ticket improves the performance of this approach, too. Table 9 summarizes the main points of SAP Logon Ticket.

Summary: Single Sign-On using SAP Logon Ticket	
Overview	SAP Logon Ticket facilitates SSO for Web-based applications across both Microsoft- and SAP platforms.
Consideration	As a prerequisite for using SAP Logon Ticket, it is required that users always have the same user ID in all the SAP systems involved in SSO using SAP Logon Tickets. To allow digitally signing the SAP Logon Tickets by the ticket-issuing server, the issuing server has to acquire a public/private key pair from PKI and distribute the public key to all ticket-accepting servers. To enforce authenticity and integrity of the tickets, it is recommended to use SSL along the communication path between all parties involved.
Availability	Single Sign-On is available on SAP NW AS ABAP, SAP NW AS Java as well as SAP NW Portal, both as ticket issuing system as well as ticket accepting system. Windows-based Web applications can also utilize SAP Logon Tickets for SSO by using either the IIS ISAPI filters or the SAPSSOEXT library.
SSO	This approach can only be used for SSO. It is not suitable for initial authentication.
Further Information	SAP Library: Using Logon Ticket for Single Sign-On (http://help.sap.com/saphelp_nw04s/helpdata/en/53/695b3ebd564644e10000000a114084/frameset.htm) SDN: Using SAP Logon Tickets for Single Sign On to Microsoft based web applications (https://www.sdn.sap.com/irj/sdn/go/portal/prtroot/docs/library/uuid/47d0cd90-0201-0010-4c86-f81b1c812e50) SDN: Java and .NET Code Samples for SAP Logon Ticket Verification (https://www.sdn.sap.com/irj/servlet/prt/portal/prtroot/docs/library/uuid/ec82ec90-0201-0010-72bc-88ef150211ff) SDN: Enabling Single Sign-On for ASP.NET Applications in Enterprise Portal 6 (https://www.sdn.sap.com/irj/sdn/go/portal/prtroot/docs/library/uuid/edb8a190-0201-0010-d398-c23e34f30295)

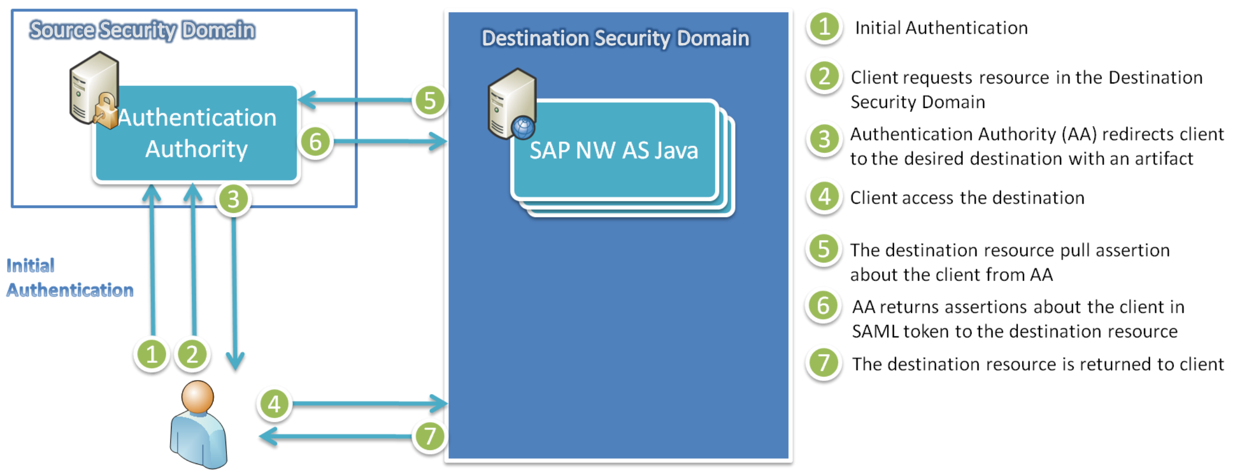
Table 9: SAP Logon Ticket based Authentication in a Nutshell

Authentication based on SAML Token

SAML refers to “Security Assertion Markup Language” and defines an XML standard for exchanging authentication and authorization information between security realms[16]. The major goal of SAML is to enable Web-based SSO across domain boundaries by exchanging security information between them. The highlight of SAML-based authentication is that it is an industry standard and delivers the basis for an interoperable security solution across domains.

[16] SAML is maintained by OASIS Security Services (SAML) TC available under: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security

In an SAML-based authentication scenario, there are two major components except client applications. The one is the authentication authority acting as an identity provider. It is responsible for performing initial authentication for users as well as for providing assertions about users (encapsulated in SAML token) on request. The other component is the resource server in the destination security domain (also called as "Service Provider" in SAML specification). Figure 6 illustrates the simplified message flow of an SAML-based authentication process within SAP context. Initially, users authenticate themselves against the Authentication Authority (AA). Afterwards, the client application sends a request to the AA to indicate that it is willing to consume an application in the Destination Security Domain. The AA receives the request and constructs assertions as well as artifact for the current user, where the artifact is used to index the assertions. Then, the AA redirects the client request together with the assertion artifact to the desired application. After the J2EE engine receives the request, it uses the SAMLLoginModule to verify the request. The login module evaluates at first the assertion artifact included in the request to determine the AA. Afterwards, it asks the AA for assertions about the user. The AA returns the assertions encapsulated in the SAML token back to the login module, which evaluates the token and authenticate the user based on information contained in the token.



Source Security Domain
Authentication Authority
Destination Security Domain
SAP NW AS Java
Initial Authentication
1 Initial Authentication
2 Client requests resource in the Destination Security Domain
3 Authentication Authority (AA) redirects client to the desired destination with an artifact
4 Client access the destination
5 The destination resource pull assertion about the client from AA
6 AA returns assertions about the client in SAML token to the destination resource
7 The destination resource is returned to client

Figure 6: SAML-based User Authentication

Overall, just like SAP Logon Tickets, the SAML token defines a way to exchange authentication information across various security domains. Therefore, SAML can be used to enable SSO across security domains that do not need to run on only homogeneous platform - either SAP or Windows platforms - or within a single domain. Furthermore, the SAML token can only be used for subsequent authentication for SSO. In other words, it needs some authentication scheme for initial authentication. Meanwhile, SAP supports SAML V1.x Browser/Artifact profile for SSO with the J2EE Engine by using the JAAS login module SAMLLoginModule with constraints. Currently the J2EE Engine can only be used as Service Provider accepting SAML tokens at runtime; it cannot operate as an Identity Provider for issuing SAML tokens with assertions about users. Therefore, a third party application supporting the role of Identity Provider is required to enable SAML-based SSO in the SAP context. Furthermore, the J2EE Engine only uses SAML token for authentication. Authorization information contained in SAML tokens is simply ignored by the J2EE Engine at runtime. Table 10 summarizes the SAML-based authentication.

Summary: Single Sign-On using SAML	
Overview	SAML is an industry standard for exchanging authentication/authorization information between security domains. With SAML, it is possible to enable Web-based SSO across different platform- as well as organizational boundaries.
Consideration	SAML does not define any security mechanisms to enforce transport-level security; therefore, it has to be used in combination with some transport-level security mechanism like SSL or WS-Security in an SOA-based environment.
Availability	SAP NW AS Java (with SAMLLoginModule as a JAAS login module)
SSO	SAML can only be used for SSO. It does not provide support for initial authentication.
Further Information	SAP Library: Using SAML Assertions for Single Sign-On (http://help.sap.com/saphelp_nw04s/helpdata/en/94/695b3ebd564644e10000000a114084/frameset.htm)

Table 10: SAML-based Authentication in a Nutshell

User Mapping

User Mapping is a simple but efficient way to enable SSO between two systems. Technically, the source system maps the external credentials of a user for accessing the target system to his user ID in its database. To access the target system, the source system simply sends the mapped credentials together with the request to the target system. Therefore, this authentication mechanism is often used in combination with User ID/Password based authentication at the target system. For instance, in the context of SAP, if the backend system does not support any other SSO technologies, such as SAP Logon Ticket or X.509 Client Certificate, then one can use User Mapping to enable SSO out of SAP NW Portal to the backend system. The implementation of User Mapping is also simple – user ID and password for accessing the external system is stored encrypted in the NW Portal database. Either administrator or users themselves can maintain this mapping between the Portal user ID and the user credentials for external systems. The only prerequisite of this approach is that the target system must accept user authentication based on user ID/password – such as Basic Authentication or Form-based authentication. Furthermore, it is also recommended to enable transport layer security like SSL to guarantee the message integrity and authenticity along the communication path.

The usage of user mapping has several drawbacks. First, it requires that a SAP portal user has to maintain a user ID and password for each backend application using user mapping. Furthermore, if the password in a backend application changes, the user has to maintain the mapped credentials in the portal, too. Therefore, user mapping can be used as an option where no other solution might work, which usually causes a significant administrative overhead. Table 11 provides a summary for this SSO mechanism.

Summary: Single Sign-On using User Mapping

Overview	User Mapping is application-specific and can be used to support SSO to SAP backend systems.
Consideration	The SAP backend system must support user ID/Password based authentication. Furthermore, transport layer security has to be enabled to ensure message integrity. User mapping usually causes a high administrative overhead.
Availability	SAP NW Portal
SSO	This approach can be used to enable SSO to backend system for SAP NW Portal where no other option to enable SSO can be used.
Further Information	SAP Library: User Mapping (http://help.sap.com/saphelp_nw2004s/helpdata/en/f8/3b514ca29011d5bdeb006094191908/frameset.htm)

Table 11: User Mapping based SSO in a Nutshell

Single Sign-On for SAP GUI for Windows

If the SAP NW AS ABAP is accessed by SAP GUI for Windows, there are no predefined mechanisms to ensure SSO as well as transport layer security. For this purpose, SAP provides the Secure Network Communication to provide end-to-end support for SSO.

Secure Network Communication (SNC)

SNC[17] aims to integrate external security products with SAP systems in a pluggable fashion. Similarly, to JAAS or PAS, SNC allows businesses to use security mechanisms that are not directly available on the SAP platform in combination with SAP backend systems. Technically, SNC provides user authentication as well as data protection at the same time. For authenticating users, SNC utilizes an external product to delegate authentication tasks. Based on the external authentication result, SNC can grant users the corresponding access to the backend SAP system. Furthermore, it also secures the communication paths between the SAP GUI as frontend and the backend SAP system.

SAP Runs on Windows Platform

In a Windows environment, where both SAP GUI and backend ABAP systems are running on Windows platform, SNC can use Integrated Windows Authentication – either NTLM or Kerberos – for SSO with the backend systems. SAP provides two implementations of the SNC interface based on Integrated Windows Authentication to enable SSO: the gssntlm.dll library using NTLM for SSO as well as gsskrb5.dll library using Windows Kerberos implementation for SSO. In either case, both the frontend SAP GUI and the backend SAP system must have the library deployed on the physical server to build a connection secured by SNC between them.

SAP Runs on a non-Windows Platform

In case that the SAP system is installed on a Non-Windows system like HP-UX, AIX, Solaris, True64, Linux or AS/400 SAP does not provide a library for using SNC with Kerberos for SSO. The message format for the Kerberos GSSAPI is well documented and defined and therefore a couple of Kerberos libraries are available for Non-Windows platforms. Many customers use the

[17] Further information about SNC is available at SAP Library under: http://help.sap.com/saphelp_nw04s/helpdata/en/e6/56f466e99a11d1a5b00000e835363f/frameset.htm

original MIT Kerberos Release, which runs on all UNIX and AS/400 platforms[18]. The MIT Kerberos implementation also ships with a couple of UNIX operating systems. SAP does not offer a certification for Kerberos but they offer a certification test tool[19], which allows you to test the Kerberos implementation on your platform.

SAP has incorporated standards based SSO technologies such as Kerberos and the GSS API into SAPGUI. Windows 2000 Servers and Clients onwards natively support Kerberos and SAP provide a GSSAPI wrapper for Windows servers and clients. Non-Windows SAP servers such as HP-UX, AIX, Solaris, Tru64, Linux, AS/400 and zOS are able to interpret Kerberos tickets sent from Windows clients, as the Windows Kerberos implementation is compatible with the Kerberos RFC. Most UNIX vendors are now bundling Kerberos with their operating system releases. Many UNIX vendors are now also shipping a GSS API. SAP provides a tool to test vendor GSS API[20]. It is not recommended to use the MIT Kerberos release unless the operating system does not include a Kerberos5 implementation or a third party Kerberos implementation is not available.

If a vendor GSS API[21] is not available or missing required parameters or features SAP publish the source code of their SNC adapter on SDN – search for "SNC Adapter". The readme file explains how to compile the source code for each platform. As the UNIX server is not a member of the Active Directory domain a local keytab file must be generated using the KTPASS tool so the UNIX server can receive and validate Kerberos tickets from Windows clients.[22]

In case you want to use an OS-Vendor supplied Kerberos5 implementation and in particular in a heterogeneous environment with Microsoft W2K or XP Workstations then please ask your OS-vendor for support, documentation on installation and an interoperability statement with SAP BC-SNC certifiable interface and Microsoft's Kerberos.[23]

Summary

In this section, we have introduced a set of mainstream enabling technologies for SSO in the Microsoft/SAP context. Figure 7 provides an overview of the technologies. Generally, we distinguish the technologies by the ways to access SAP backend systems: either through Web-based approaches or via the SAP GUI for Windows. Then, we classify the technologies by their major operational domains – if the technologies can be used for initial authentication in SSO, for SSO, or for both of them. All the technologies have their strength and weakness for applying to the real-world applications. Therefore, the question, which technologies should/can be deployed to enable SSO, depends on several factors of target usage scenarios, such as existing security infrastructure, capabilities of existing applications/servers for SSO, the usability of the target implementation, as well as desired security level of the target implementation. In the

[18] For more information on the MIT Kerberos release see for example http://web.mit.edu/kerberos/www/krb5-1.1/index.html

[19] GSSTEST, see https://www.sdn.sap.com/irj/sdn/sdnservices/icc?rid=/webcontent/uuid/e112cb72-0501-0010-63a3-f45326c176ae

[20] Please refer to SAP Note 1043694 – "Performance Impact when using SNC for Single Sign-On" for more information

[21] More information on GSS can be found here: http://en.wikipedia.org/wiki/GSSAPI

[22] The precise configuration steps vary according to the UNIX implementation and very different for AS/400 and zOS. For more information please contact sapcoe@microsoft.com

[23] See SAP Note 352295 - Microsoft Windows Single Sign-On options

next section, we will outlines usages of the technologies introduced in this section by applying them to some typical usage scenarios from daily business.

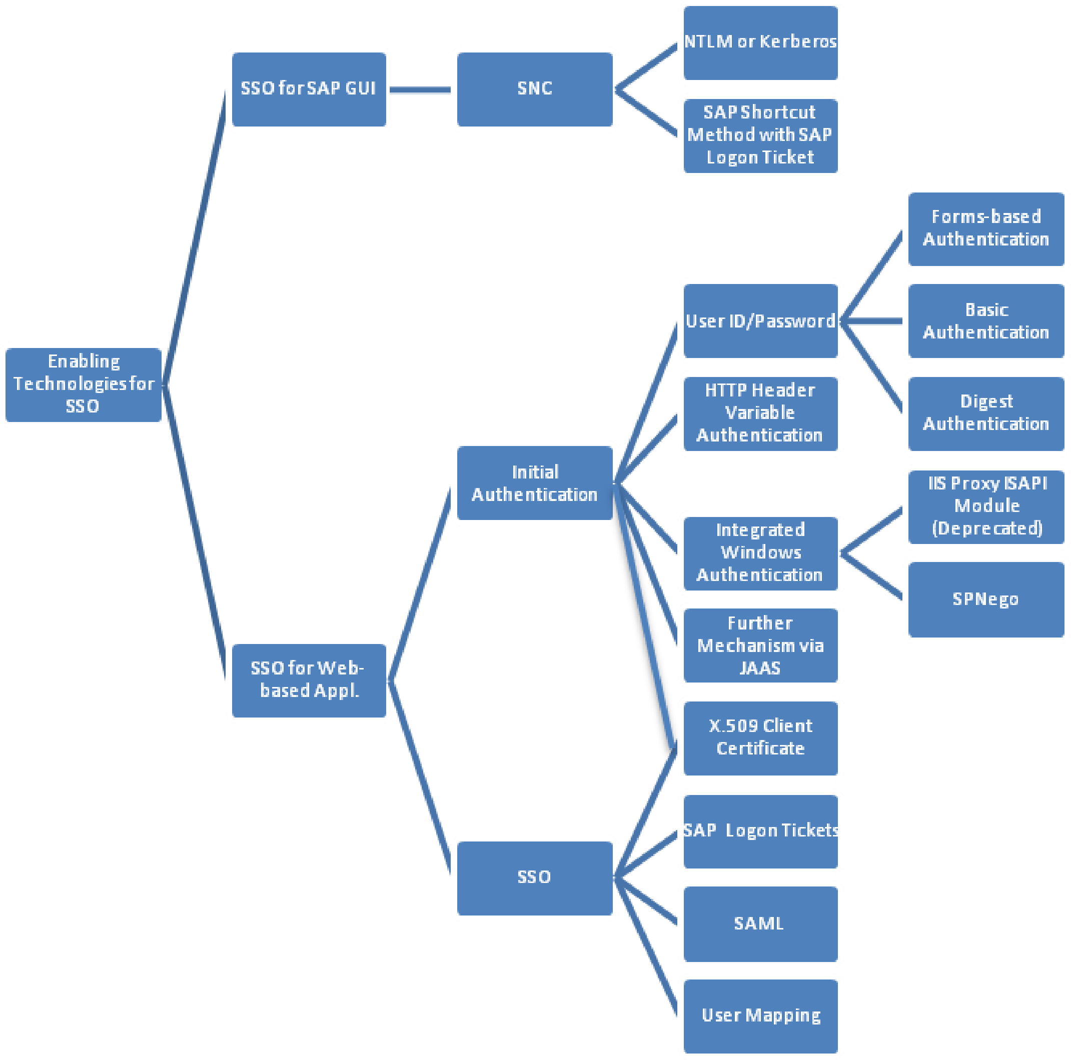

Enabling Technologies for SSO
SSO for SAP GUI
SNC
NTLM or Kerberos
SAP Shortcut Method with SAP Logon Ticket
SSO for Web-based Appl.
Initial Authentication
User ID/Password
Forms-based Authentication
Basic Authentication
Digest Authentication
HTTP Header Variable Authentication
Integrated Windows Authentication
IIS Proxy ISAPI Module (Deprecated)
SPNego
Further Mechanism via JAAS
X.509 Client Certificate
SSO
SAP Logon Tickets
SAML
User Mapping

Figure 7: Classification of the Technologies for SSO in Microsoft/SAP Context

SSO Usage Scenarios

As we have introduced in the last section, there are a set of technologies for SSO, which are different in their capabilities, prerequisites on infrastructure, operational/administrational overhead due to solution's complexity as well as security level that can be achieved. To determine which technology is most suitable for particular usage scenarios is also a process that requires a lot of analysis of the corresponding scenarios. In this section, we will review some typical usage scenarios from Microsoft/SAP context to outline how the technologies can be applied to real world applications.

SSO from Windows-based Workstations to SAP using SAP Frontends

The most widely used scenario is to use SAP frontends to access SAP applications from windows based workstations. As mentioned in the previous paragraph, the most straightforward way to perform initial authentication is to reuse the authentication information from Windows – namely by using Integrated Windows Authentication. In the past, different approaches had to be taken to achieve SSO using windows integrated authentication for a browser based or SAPGUI based access.

As mentioned above in a Windows environment, where both SAP GUI and backend ABAP systems are running on Windows platform, SNC can use Integrated Windows Authentication – either NTLM or Kerberos – for SSO with the backend systems. If the backend systems are based on UNIX SAP's support for SSO for SAPGUI is limited. Customers are forced to either use a third party SNC solution or go for a Kerberos implementation on the UNIX side. Both approaches have their therefore disadvantages due to additional licensing costs and support.

For browser-based access, SSO using the SPNego Login Module could be used for Windows platform and for non-Windows platforms.

Here the new SAP NetWeaver Business Client comes into play. The SAP NetWeaver Business Client is SAP's next generation windows desktop client using the latest smart client technology. It is using the Portal services infrastructure for role-based access to SAP systems and consistent navigation capabilities and it can host existing SAP UIs in the "canvas" area (including SAP GUI and WebDynpro) as well as any other web-based content.

The SAP NetWeaver Business Client supports Integrated Windows Authentication as the initial authentication if the SAP NetWeaver Portal services infrastructure used is configured to use the SPNego Login Module. A great advantage of using the SAP NetWeaver Business Client is that it supports SSO especially for SAPGUI based UI, even if the SAP backend applications are running on UNIX. The limitations described above do not apply in this case. The SAP NetWeaver Business Client is available with SAP ERP 6.0 Enhancement Package 2.

After the user has been successfully authenticated against the portal infrastructure, a SAP Logon Ticket is issued. This SAP Logon Ticket is used by the NetWeaver Business Client to access all of SAP's business applications that are part of the portal infrastructure using SSO.

The NetWeaver Business Client runs not only new Web Dynpro or Visual Composer applications but also classic UIs at SAP that are implemented as Dynpros. Given the long tail of application development, it can to envision that over many years there will still be SAPGUI transactions that are very specialized and required for years to come.

The SAP NetWeaver Business client does also support access to BSP pages, analytic dashboards, iViews, portal pages, etc. and bridges the gap of today's thick clients (e.g. SAPGUI) and thin HTML clients, while catering to the high demands of business users. It especially provides them with the option to use SSO to all SAP's business applications that are part of a real world scenario without the need to use any 3rd party software.

SSO from Windows-based Applications to SAP

A typical scenario from daily business is to use Windows-based applications as frontend for the backend SAP systems, as illustrated in Figure 8. In this scenario, the Windows workstation belongs to a domain with an Active Directory in the backend. There are a set of applications that fall into the category “Windows applications” illustrated in the figure – Smart Clients like Office-based applications consuming SAP Web services, Rich Clients like .NET-based stand-alone applications, or Browser Applications. As backend systems, there are SAP NW AS ABAP/Java Engines deployed to deliver business-related data for frontend applications. These ABAP/Java Engines have different roles regarding SSO in the scenario: some of them operate as ticket issuing systems for SAP Logon Tickets while some other ones accept the SAP Logon Tickets as ticket accepting systems. Now, we will try to find a SSO solution that is most suitable for this scenario.

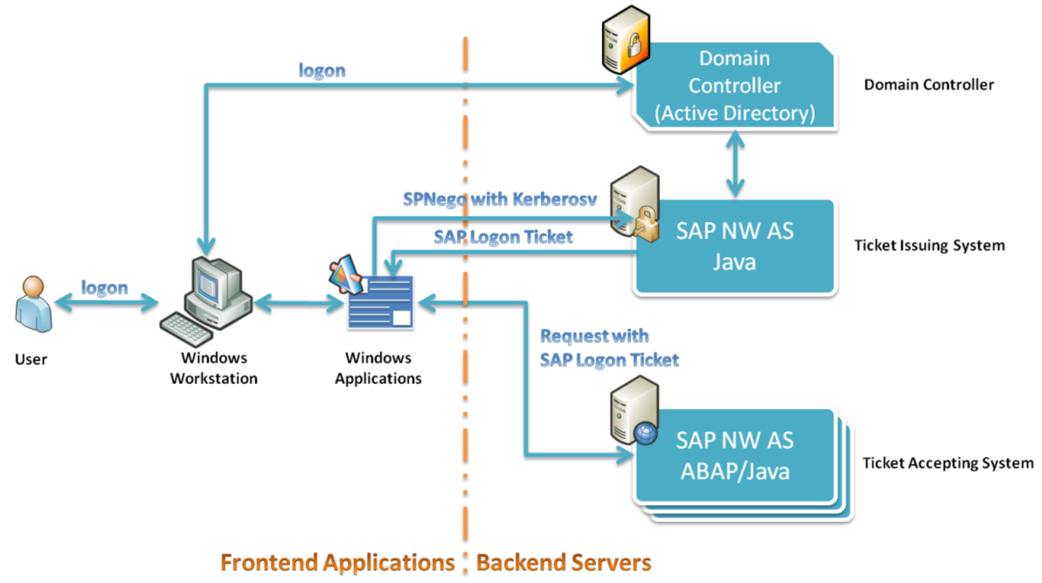


Figure 8: SSO Usage Scenario from Microsoft to SAP

The first task is to determine the authentication mechanism for initial authentication. In this scenario, we have a Windows workstation that is a member in a Windows domain. To start the Windows applications that access SAP, users have to logon to the Workstation at first by submitting their user ID and password. In other words, if a user has logged on to the workstation, he has also been authenticated by the AD in the domain. Comparing the technologies for initial authentication reviewed in the last section, we can find out that the most straightforward way to perform initial authentication is to reuse the authentication information from Windows – namely by using integrated Windows Authentication. Of course, there are other authentication mechanisms that can be used in this scenario, e.g. Basic Authentication. In this case, users have to re-enter their user ID and password again for authentication (we assume that users have the same user ID in the SAP backend systems as their Windows logon ID).

The second task is to determine the SSO mechanisms for accessing the backend SAP systems. In the scenario, the client applications may require access to several SAP systems. The invocation path may even contain several SSO-hops (e.g. calling a Web service running on J2EE engine that has to access ABAP engine to gather necessary business data, in this case, the number of SSO-hops is 2). Therefore, it is reasonable to use SAP Logon Tickets for enabling SSO. X.509 client certificate-based authentication would require in this case the availability of a PKI infrastructure, which is missing in this scenario. The SAML-based approach needs an authentication authority supporting the SAML Browser/Artifact profile, which is also missing in this case. Therefore, SAP Logon Ticket based SSO is the only suitable technology in this scenario.

Now, the final concept to enabling SSO in this scenario is as follows: the user has to logon at first to the Windows workstation by authenticating himself against the backend AD. Then, the user starts the application that is calling Web services on the backend SAP systems. The SAP NW AS J2EE engine that acts as a ticket-issuing server in the landscape has the JAAS login module SPNegoLoginModule as well as CreateTicketLoginModule deployed in its authentication stack. The initial authentication is carried out by utilizing integrated Windows Authentication with the SPNego as well as the Kerberos protocol between the Windows applications, the AD, and the SAP NW AS J2EE engine. Afterwards, the Windows application gets an SAP Logon Ticket issued by the ticket-issuing server for subsequent access to other backend SAP systems in the landscape. To access the servers operating as the ticket accepting servers in the landscape, the Windows applications send the HTTP (or SOAP) request to the services on those servers together with the SAP Logon Ticket for SSO.

The solution suggested in this section is only a possible concept for realizing SSO for Windows-based applications accessing SAP systems – with minimal administration efforts and maximal usability for end users. As we already mentioned afore, the decision about which technologies are most suitable for particular case must be made according to the operating environment where the solution will be deployed later. For example, another typical scenario in daily business is to access SAP NW Portal out of Windows workstation. In this case, we only have a single SAP NW Portal (SAP NW AS Java) that must act as ticket issuing server and simultaneously as ticket accepting server, if we can use SAP Logon Ticket for SSO in this case.

Further Reading

SDN: Single Sign-On of Windows-Based Web Service Clients using SAP Logon Tickets (https://www.sdn.sap.com/irj/sdn/go/portal/prtroot/docs/library/uuid/9bdcb279-0e01-0010-b5ac-ef7f99e44c68)
SDN: Sample Application: SSO with a .NET-based Web Service Client using SAP Logon Tickets (https://www.sdn.sap.com/irj/servlet/prt/portal/prtroot/com.sap.km.cm.docs/library/uuid/5bc7e899-0e01-0010-cca9-84f45118dd17)

SSO from SAP Applications to Microsoft

Another typical scenario is to integrate Windows applications into SAP enterprise applications in an extranet scenario. A typical example for this scenario is to integrate Exchange Server (OWA) into SAP NW Portal. In such a scenario, users have been already authenticated by the SAP applications. However, SSO to Windows-based backend systems in extranet scenarios cannot make use of Windows Integrated Authentication as it is possible in an Intranet scenario, because the user frontends are not part of the same Active Directory forest and are usually not trusted. Hence, the major task is to find a SSO mechanism that is also accepted by the Windows-platform.

Let us go back to the scenario with OWA integration in SAP NW Portal. As we discussed in the last section, IIS as well as ASP.NET support a set of build-in authentication methods: incl. integrated Windows authentication, basic authentication with user ID/password, and X.509 client certificate authentication. In this case, we know that if the OWA is running on Windows server 2003, we can use integrated Window authentication for SSO.

The only technical difficulty is to convert the SAP Logon Ticket to a Kerberos Ticket for authentication at the Exchange server. This capability is exactly what SAP's IIS ISAPI filter called *SSO22KerbMap Module* can provide – it evaluates the SAP Logon Ticket submitted by the SAP NW Portal for validity and acquires a Kerberos ticket for SSO on the Exchange server. Another approach for integrating ASP.NET applications into SAP NW Portal is to use User Mapping between both systems. A typical example for this scenario is to integrate Window Live Hotmail into SAP NW Portal. Windows Live Hotmail uses the Microsoft proprietary Passport for SSO, while SAP NW Portal uses the SAP's proprietary SAP Logon Tickets for SSO. Therefore, the only way to enable SSO between Windows Live Hotmail and SAP NW Portal is to use User Mapping – users maintain their Windows Live Hotmail credentials in their iViews, which uses HTTP Post to submit the credentials to Windows Live Hotmail for authentication.

Another approach is to give the ASP.NET application the ability to verify SAP Logon Tickets. A typical scenario for this approach is to consuming a Windows-based Web service – e.g. ASP.NET Web service or WCF Web service – out from the J2EE/ABAP engine. SAP provides the necessary basic library for verifying SAP Logon Tickets. This approach can be taken into consideration, if it is possible to modify the ASP.NET application that should accept the SAP Logon Tickets – meanwhile, this procedure also becomes unnecessary with the new pluggable authentication concept of WCF that allows to simply modifying the authentication stack of a WCF Web service by changing the configuration file. Next to the build-in support for standard security credentials such as user ID/password, X.509 certificate or SAML token, WCF allows developers extending its security infrastructure by creating modules to validate custom client credentials – for instance, SAP Logon Tickets. For such custom client credentials, the WCF programming model supports to specify credential validator using service behavior. More information about the extensible security programming model of WCF is available on the MSDN.

Further Reading

SDN: Using SAP Logon Tickets for Single Sign On to Microsoft based web applications (https://www.sdn.sap.com/irj/sdn/go/portal/prtroot/docs/library/uuid/47d0cd90-0201-0010-4c86-f81b1c812e50)

SDN: Integration of Windows File Servers into the SAP KM platform using SSO and the WebDAV repository manager (https://www.sdn.sap.com/irj/servlet/prt/portal/prtroot/docs/library/uuid/e1f93f5c-0301-0010-5c83-9681791f78ec)

SDN: Enabling Single Sign-On for ASP.NET Applications in Enterprise Portal (https://www.sdn.sap.com/irj/sdn/go/portal/prtroot/docs/library/uuid/edb8a190-0201-0010-d398-c23e34f30295)

SDN: Single Sign-On to a Microsoft Exchange Cluster (https://www.sdn.sap.com/irj/servlet/prt/portal/prtroot/docs/library/uuid/9bdcb279-0e01-0010-b5ac-ef7f99e44c68)

MSDN: Custom Credential and Credential Validation (http://msdn2.microsoft.com/en-us/library/ms734697.aspx)

MSDN: WCF Security Terminology (http://msdn2.microsoft.com/en-us/library/ms731846.aspx)

Conclusion & Outlook

In this whitepaper, we have reviewed a number of enabling technologies for SSO within Microsoft/SAP context. SSO is the technology that brings both end users as well as enterprise a set of advantages. For end users, SSO means a consistent experience to access resources deployed everywhere in the business without having to remember passwords as well as re-enter them for every resource they access. This also enhances the productivity of end users indirectly. For enterprise, it means a cost-effective solution for access control due to lower administration cost for user database(s), high productivity of the employees as well as the high security level of the enterprise IT infrastructure. Nevertheless, SSO also means the "key to the kingdom", which remains a potential security threat for the enterprise IT – if one can obtain the SSO credentials, he gets also access to all resources secured by them. This risk remains irreducible, unless an additional stronger authentication scheme is used for areas requiring higher security levels, such as using smart card, or fingerprint.

Conclusion

Table 12 provides an overview of the technologies. Thereby, we classify the technologies by the following three aspects:

- *Objective*: each technology has its own operation area – some of them can be use to perform the initial authentication upon the users, while some of them can only be used as security tokens to enable SSO. Therefore, we categorize the technologies in this aspect to the following two areas: *initial authentication* and *SSO*.
- *Availability*: among all the technologies we discussed in this whitepaper, there are common technologies that are supported by both Microsoft and SAP platforms. However, there are also some proprietary technologies that are only available on SAP platform. In this aspect, we categorize the technologies into the following areas: *SAP NW AS ABAP*, *SAP NW AS JAVA*, and *Microsoft IIS/ASP.NET Application/WCF*.
- *SSO to each other*: we have discussed in this whitepaper, how the SSO technologies can be applied to enable SSO between Microsoft and SAP platforms. Therefore, in this aspect, we categorize the technologies into the following areas: *SSO – from Microsoft to SAP* and *SSO – from SAP to Microsoft*. Note: this classification only applies to technologies that are capable for SSO.

Technologies	Objective		Availability			SSO to each other	
	Initial Auth.	SSO	SAP NW AS ABAP	SAP NW AS Java	Microsoft IIS/ ASP.NET App./ WCF	SSO - SAP to Windows	SSO - Windows to SAP
User ID/Password							
- Form-based	*			*	*		
- Basic Auth.	*		*	*	*		
- Digest	*			*	*		
HTTP Header Variab.	*			*			
Integrated Windows Authentication							
- IIS Proxy	*			*			
- SPNego	*			*	*[24]		
SAP Logon Ticket		*	*	*	*[25]	*[26]	*[27]
SAML		*	*		*[28]	*	*
X.509 Client Cert.	*	*	*	*	*	*	*
User Mapping		*		*[29]	*	*	*[30]
SNC	*	*	*				*
JAAS	(*)	(*)		*			
PAS	(*)	(*)	*				

Table 12: Overview of SSO Technologies in Microsoft/SAP Context

Outlook

Driven by the increasing trend to business processes that span both organizational and technical boundaries, SSO will remain a strategic topic for the enterprise. As two leading vendors in enterprise-level software market, both Microsoft and SAP are working on their product line to follow this trend. As Kim Cameron described in his article "The Laws of Identity"[31], what we need is a "*unifying identity metasystem that can protect applications from the internal complexities of specific implementations and allow digital identity to become loosely couple*".

Federation with Windows Platform

Meanwhile, Microsoft has made several efforts to federated identity across organizational boundaries. One of them is the Active Directory Federation Services (AD FS) to provide extensible and secure Web-based SSO that can operate across platforms, including Windows and non-Windows environment. Similar to the SAML specification of OASIS, AD FS aims to provide the basis for exchanging authentication as well as authorization information across several domains in the Internet. The basic idea of both approaches is to enable token-aware applications based on standard Web Service specifications, so that an application can evaluate tokens issued by trusted identity providers for authentication, which

[24] IIS supports only integrated Windows Authentication based on NTLM or Kerberos tickets.
[25] Requires the SAPSSOEXT.dll library to validate the SAP Logon Tickets
[26] Requires additional components
[27] Application has to acquire a SAP Logon Ticket from a ticket-issuing server at first.
[28] WCF supports only SAML token within the context of WS-Federation
[29] Supported by SAP NW Portal running on J2EE Engine
[30] Only with constraints
[31] The Laws of Identity by Kim Cameron is available under http://msdn2.microsoft.com/en-us/library/ms996456.aspx. In this work, Cameron explains the necessary for a unifying identity metasystem that can offer the Internet the identity layer it needs.

build up ultimately a federated digital identity of a person across all the systems involved. In the Windows Server 2008 (previously called Windows Server "Longhorn"), AD FS provides a better support for WS-Federation in MOSS 2007 as well as Active Directory Rights Management Services (AD RMS)[32].

Another effort regarding federated identity is the CardSpace[33] introduced within the new .NET Framework 3.0. It aims to provide the end user a simple but secure logon experience across all Web-based applications. The basic idea of CardSpace is to provide a unified und centralized area for users to organize their digital identities. A digital identity is stored as a "card" in the CardSpace, which is issued by various identity providers such as employer, bank, or any other membership organization. If a Website requires an identity for authentication, CardSpace allows end user to choose a card to present to the target applications. Generally, CardSpace has a similar concept as the X.509 client certificate, but it demands fewer administration efforts and is interoperable due to the Web service standards that it uses.

[32]More information about AD FS is available under http://technet2.microsoft.com/windowsserver2008/en/library/f5e12c1f-a3fa-453d-98ce-be29352afaca1033.mspx?mfr=true

[33] Further information about Windows CardSpace is available under http://msdn2.microsoft.com/en-us/netframework/aa663320.aspx

Table of Figures