A BIO-key® Solution.

Tech Brief

SAML Single Sign-On

Table of Contents

Table of Contents

Summary

Putting an end to user complaints about having to remember multiple passwords is an objective in many organizations. With multiple web applications being accessed, IT staff often struggles to manage multiple user repositories. A common complication that arises is when a password is changed in one repository, but is not updated in the others. This can lead to both security and support issues that make it even more difficult to implement a password security policy across multiple systems.

To solve these issues, and streamline access for users by eliminating multiple password prompts, you may look towards Single Sign-On (SSO). However, many SSO solutions can be costly, difficult to implement, and unable to effectively handle all user access scenarios. Integration is especially difficult when attempting to allow the Single Sign-On experience to continue for external users - from staff to customers or partners - who all want seamless access to hosted web applications.

Without Single Sign-On, for example, a typical scenario might unfold like so: You log in to your locally managed application – with your active directory credentials – and you also decide to check your email through Office 365. In order to do so, you must manually login to Office 365, despite having already authenticated to your local directory. If you were to decide to also check on your stocks or other information accessed through an external application, you would be required to authenticate yet again.

With a Single Sign-On solution in place, you would be able to check your emails in Office 365, view your current stock situation, and access any other necessary applications based on the strength of your initial verification to your locally managed application. In order to provide a seamless experience – while maintaining high levels of security and both internal and external compliance - this integration is absolutely essential.

The solution to these common login frustrations is a product that can create a single or federated authentication process to handle multiple local and cloud applications, while providing a centralized point of secure access.

"Before PortalGuard, our students had several passwords to remember. With PortalGuard, they only have one password, and if they forget it, they can reset it on their own."

-Andy

Clermont Northeastern Schools

The Basics

Identity Federation

Identity federation is the concept of linking the identity of a particular user across multiple systems or servers. When two servers are federated, the authentication against one can be leveraged to verify the identity of the user to the other. Some application servers in the secondary role can allow this without requiring the user to register an account.

[Identity federation typically entails some level of Single Sign-On (SSO). Once authentication has been performed against a primary server, the user's session with that server can then be used as a launch point for SSO-based access to other federated services. This can be used to realize the common business requirement of reducing access barriers without compromising the security of the systems involved.]

There are multiple protocols that can be used to apply the successful authentication against one system to another, but Security Assertion Markup Language (SAML) has emerged as a clear front-runner.

Security Assertion Markup Language (SAML)

Originally developed by the OASIS Security Services Technical Committee, Security Assertion Markup Language (SAML) is leading the way in providing seamless, web-based SSO as an open, widely implemented, industry standard protocol. SAML is an XML-based authentication protocol that passes assertions between SAML-enabled applications.

Once the end-user requests access to a resource, an online identity provider creates a SAML token containing the end-user's identity assertions. Once the resource server validates those assertions, the end-user is granted access without any further password prompts.

SAML is heavily leveraged today for numerous reasons:

- It works for cloud-based services that are typically hosted offsite as well as "on premise" services.
- It is typically wrapped in the HTTP/HTTPS protocols which ensures it can be used by any client device regardless of operating system (e.g. Windows, Mac, and Linux) or architecture (PC, iPad, smart phones).

- The use of HTTP/HTTPS also allows for easier network administration since these ports are more frequently open in server or client firewalls.

- Manual user authentication for multiple services can be redirected and always be performed against a single Identity Provider (IdP). This “choke point” allows for network and access policies to be controlled at a single point, making them much easier to implement and enforce

- Users can be authenticated against virtually any user repository using any required method(s) without impacting the downstream servers, which always receives a SAML assertion.

The PortalGuard Identity Provider (IdP)

The PortalGuard Identity Provider (IdP) acts as a SAML-based portal, using a single set of user credentials for the portal login itself to then grant access to various web-based applications. When using SAML, the end-user will no longer be interrupted with multiple login prompts. As a result, administrative and IT costs associated with performing password management related tasks - such as password resets, synchronizing numerous sets of password quality rules, and creating and disabling accounts - will be greatly reduced.

PortalGuard provides seamless integration with web-based applications: whether they are cloud-based, private, on-premises, or behind a firewall. This integration allows organizations to streamline access for all end-users, while still maintaining strong and secure authentication.

Achieving Stronger Authentication

Along with providing a central point of access from which end users can login to various applications, the PortalGuard IdP can easily be configured for numerous Multifactor Authentication methods - vastly increasing the strength of the central login.

Far from allowing for a single point of failure, the combination of the PortalGuard IdP and SAML SSO provides both end users and administrators with adequate control to keep attackers at bay.

Some Features that may be used to strengthen a SAML-based login are:

- Advanced Reporting Functionality
- Contextual Authentication
- Forced Enrollment of Challenge Questions
- Support for 11 Different 2FA Methods
- Self-Service Password Management

NOTE: Although many web-based applications are already SAML- enabled, PortalGuard supports numerous alternative SSO protocols as well, such as: WS-Federation, forms-based SSO, Kerberos, CAS or Shibboleth. Each of these SSO protocols is supported by PortalGuard to provide your environment with the best option for Single Sign-On. (For More Information, See the Alternative SSO Methods Section Below)

Flexible Usage Scenarios

The inherent flexibility of PortalGuard allows you to choose the appropriate authentication method for each user, group, or application by leveraging Contextual Authentication. Varying access scenarios in every organization drive the need for this type of authentication. For instance, users on your Local Area Network (LAN) may only need to provide strong passwords; whereas, a traveling salesperson or external user is presented with Two-Factor Authentication.

Consistent Authentication Interface

When the user is always forced to login to your SAML-enabled applications using PortalGuard, a consistent authentication interface and process can be enforced. This reduces end user training and frustrations associated with managing multiple accounts through multiple websites.

Enforcing Self-Service Enrollment

Using SAML SSO offers a seamless way to provide users the ability to unlock their account, enroll answers to challenge questions, reset or recover their forgotten passwords, and manage their mobile device for use in alternate or Multifactor Authentication.

Multifactor Authentication

Because the end user communicates directly with the PortalGuard server, authentication decisions made by PortalGuard are strictly enforced. This ensures a high level of security and consistency.

Benefits of SAML SSO

- Eliminate the need to develop and maintain your own portal
- Reduce the number of passwords users are required to remember and manage
- Implement and enforce configurable password policies
- Remove the need to manage external users' credentials
- Optionally increase security using any combination of transparent barriers
- Add stronger authentication using Two-Factor and/or Knowledge-Based Authentication for select users or groups of users (e.g. Administrators)
- Reduce password-related Help Desk calls related to password and access issues

How it Works

The following steps and screenshots show how the PortalGuard SAML IdP works using two different SSO methods: SP-Initiated and IdP-initiated.

SP-Initiated SSO

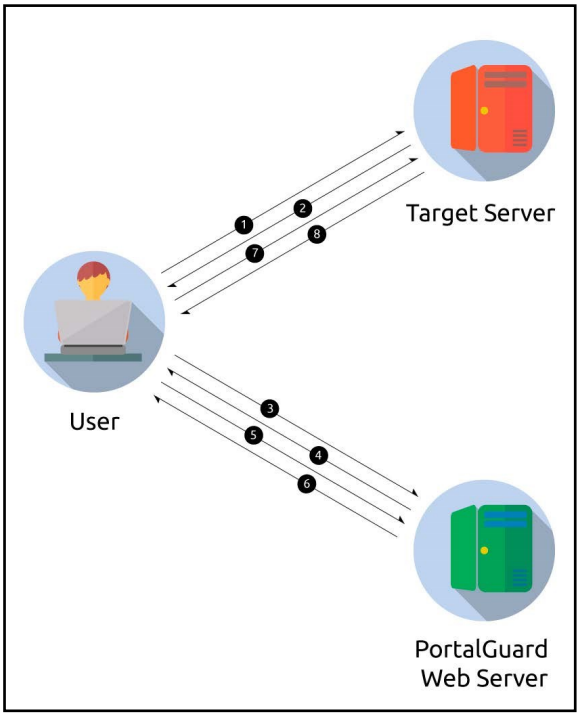
SP-Initiated SSO occurs when a user attempts to login directly to the desired service, without first authenticating to the local Identity Provider. If the user has not yet authenticated, they are required to login using local credentials, which suffice to grant access to the requested service. The steps for achieving this are outlined below.

STEP 1: The user opens the browser on the client machine and accesses the target server; e.g. http:// mail.google.com/a/example. com

STEP 2: The target server sees that the user has not yet authenticated, it generates a SAML request and returns it alongside the originally requested URL (the "RelayState") to the client machine as hidden input fields in an HTML-form response.

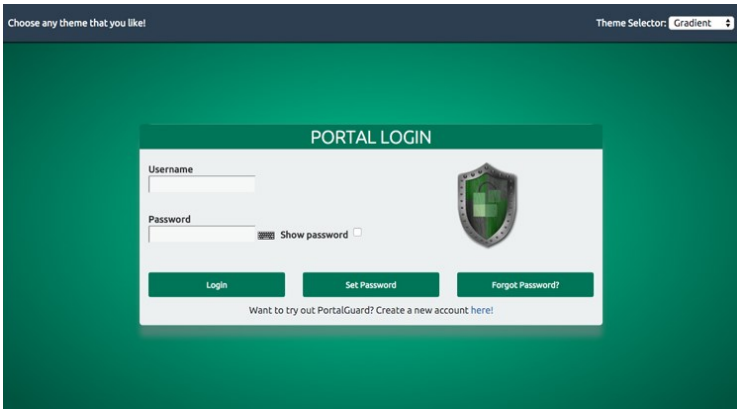
STEP 3: JavaScript in the response automatically submits the form to the PortalGuard Identity Provider (IdP).

Note: The user can be forced to login using any of PortalGuard's authentication methods - including Knowledge-Based or full Two-Factor Authentication (2FA).



STEP 4: The user is presented with the PortalGuard login screen. The user enters the appropriate username/password and clicks "Login".

Note: this login screen can be fully customized to match the specific branding of your organization, creating a seamless experience for the user. The user can optionally reset a forgotten password from this screen too.

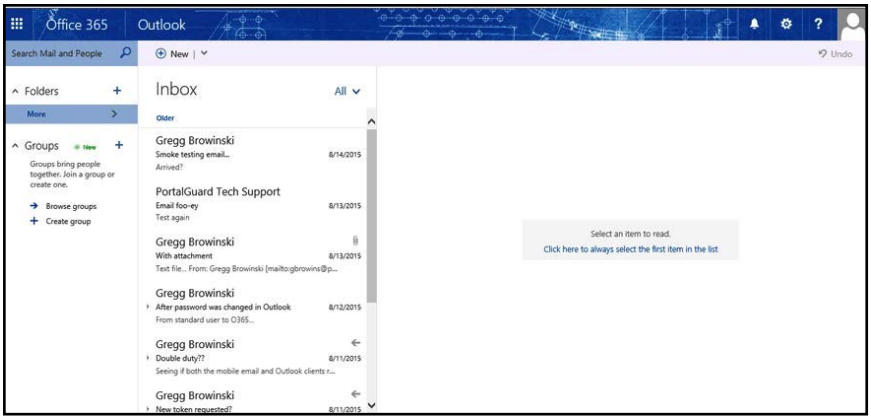


STEP 5: PortalGuard validates the submitted username and password in real-time against the appropriate directory. If correct, the client machine will have established a session with the PortalGuard web server.

STEP 6: The original SAML request is now serviced by the PortalGuard IdP. It generates a SAML response and sends it along with the "RelayState" back to the end user's browser, wrapped in an HTML form.

STEP 7: JavaScript in the HTML response automatically submits the form to the target server's Assertion Consumer Service (ACS). Both the SAML Response and "RelayState" are included in this form data.

STEP 8: The target server parses and validates the SAML response. It uses the embedded identity claims to verify the user's identity and then grants the user access to the application.



IdP-Initiated SSO

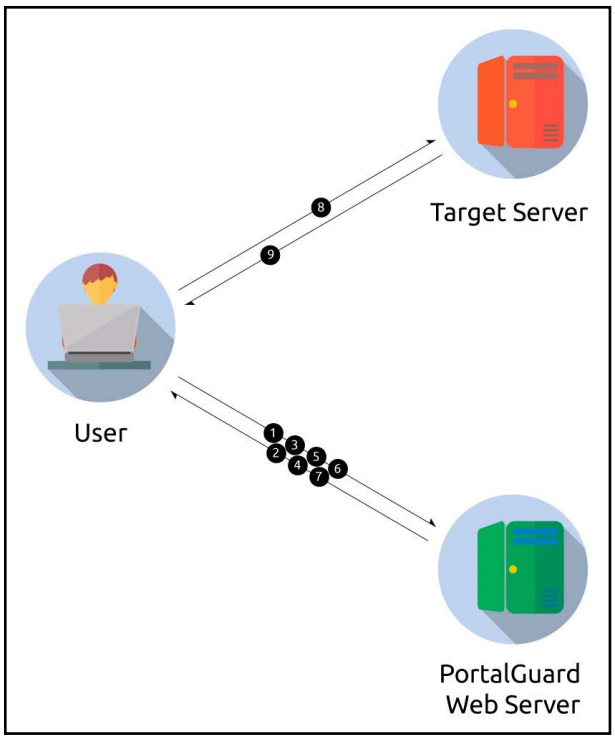
IdP Initiated SSO typically occurs when a user accesses a locally managed jump-page. By authenticating directly to the IdP first, the user can choose from a host of services to access without the need to input any additional credentials. The technical process for achieving IdP-Initiated SSO is outlined in the steps below.

STEP 1: The user opens the browser on the client machine and accesses the target server; e.g. http:// mail.google.com/a/example. com

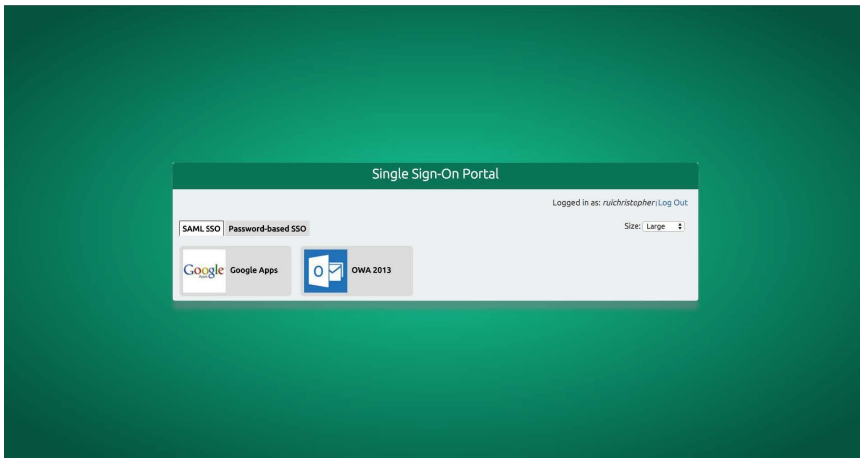
STEP 2: The PortalGuard login screen is presented to the user if they do not already have an active session.

STEP 3: The user enters his/her username and password and clicks "Login."

STEP 4: PortalGuard validates the submitted username and password in real-time against the appropriate directory. If correct, the client machine will have established a session with the PortalGuard web server.



STEP 5: The user is granted access to the SAML IdP jump page.



STEP 6: The user clicks a displayed application. ***Note***: the applications available to the user are configurable by the administrator.

STEP 7: The click is serviced by the PortalGuard IdP, which generates a SAML response and sends it back to the end user wrapped in an HTML form.

STEP 8: JavaScript in the response automatically submits the form to the target server's Assertion Consumer Service (ACS).

STEP 9: The target server parses and validates the SAML response. It uses the embedded identity claims to determine the user's identity and grant the user access to the application.

Deployment

Implementation of the PortalGuard platform is seamless and requires no changes to the existing Active Directory/LDAP schema. A server-side software installation is required on at least one Windows server on the network which is running Microsoft IIS.

PortalGuard is a flexible authentication platform with multiple layers of available functionality to help you achieve your authentication goals. These layers include:

- Self-Service Password Reset
- Web-based Single Sign-On
- Two-Factor Authentication
- Contextual Authentication
- Password Synchronization
- Password Management

PortalGuard is easy to install, configure and brand. It has given us the ability to allow our agents to self-service their Active Directory accounts, giving us a level of security we've needed but were unable to achieve before PortalGuard.

-Faye

Canal Insurance

System Requirements

PortalGuard supports both direct access and authentication to cloud/web-based applications. PortalGuard has the following requirements:

Requirements For On-Premise

- .NET 2.0 framework or later
- (64-bit OS only) Microsoft Visual C++ 2005 SP1 Redistributable Package (x64)

PortalGuard is fully supported for installation on physical and virtual machines. PortalGuard supports running on the following platforms:

- Microsoft Windows Server 2008 / R2
- Microsoft Windows Server 2012 / R2

Requirements For Nebula

When connecting an Active Directory or LDAP user repository to the Nebula Servers, the following must be done:

- SSL must be enabled on the Domain Controller
- Export the public certificate from the Certificate Authority (CA) that created the SSL Certificate.
- Create a service account in the user repository and delegate the appropriate read and write privileges to it.

In addition to the necessary user repository changes, a change to the local firewall/ gateway must be made.

- Access to Port 636 must be open, to provide a communication channel between the user repository and the Nebula servers.

  NOTE: This can be additionally secured using IPSec between the DC and the Nebula servers using machine certificates.

Common Requirements For the PortalGuard IdP

The PortalGuard IdP requires the following:

- The target server must support either IDP or SP-initiated SAML SSO using the SAML POST binding method.

- The target server must be configured to not allow manual authentication. Otherwise, users could use that method and bypass the interactions with PortalGuard (typically this is implicit when enabling SAML).

- A trust must be configured between the PortalGuard Identity Provider and the target server/Service Provider by importing the PortalGuard public signing certificate.

- The end user must have network connectivity (typically HTTPS) to both the PortalGuard server and the target server.

- The PortalGuard server does not need network connectivity to the target server since the user's browser delivers all SAML messages.

Alternate SSO Methods

Central Authentication Service (CAS)

Developed by Shawn Bayern at Yale university, CAS differs from typical SAML SSO by enacting Server to Server communication. The Client Machine is used to initiate the token request, but the final verification is handled by a back-end communication between the CAS server and the Service Provider. CAS is a typical SSO protocol used in education organizations because of reliance on that extra, more direct verification. Like SAML, no passwords are exchanged through the SSO token.

Shibboleth SSO

Shibboleth is another SSO protocol typically seen in educational organizations - specifically where a high number of institutions are federated to share applications and/or services. Shibboleth is built with SAML as a foundation, but uses Discovery Service to improve upon SAML's organization of data from a large number of sources. Additionally, Shibboleth helps to automate parsing of meta data to handle security certificate updates and other configurations that may be set by individual institutions within a federation

Cookie-Based SSO

Works by using Web based HTTP Cookies to transport user credentials from browser to server without input from the user. Existing credentials on the client machine are gathered and encrypted before being stored in the cookie and sent to the destination server. The server receives the cookie, extracts and decrypts the credentials and validates them against the internal server directory of users.

Claims-Based SSO

Claims (aka "assertions") are created by a claims issuer that is trusted by multiple parties. Claims are typically packaged into a digitally signed token that can be sent over the network using Security Assertion Markup Language (SAML).

NTLM-Based SSO

It is possible for a user to prove they know their password without actually providing the password itself. NTLM achieves this using a challenge and response protocol that rst determines what type of NTLM and encryption mechanisms the client and server mutually support, then cryptographically hashes the user's password and sends it to the server requiring authentication.

Kerberos-based SSO

Kerberos enables users to log into their Windows domain accounts and then receive SSO to internal applications. Kerberos requires the user to have connectivity to a central Key Distribution Center (KDC). In Windows, each Active Directory domain controller acts as a KDC. Users authenticate themselves to services (e.g. web servers) by first authenticating to the KDC, then requesting encrypted service tickets from the KDC for the specific service they wish to use. This happens automatically in all major browsers using SPNEGO (see below).

SPNEGO-based SSO

There are instances when the client application and remote server do not know what types of authentication the other one supports. This is when SPNEGO (Simple and Protected GSSAPI Negotiation Mechanism) can be used to find out what authentication mechanisms are mutually available. Some of these mechanisms can include Kerberos and NTLM authentication.

Reduced SSO

Reduced Single Sign-On is widely used for limiting the number of times a user will be required to enter in their credentials to access different applications. With critical applications, reduced SSO also offers a technique to make sure that a user is not signed on without a second factor of authentication, having been provided by the user.

Enrollment-Based SSO

A user logging into a website may choose to have their credentials permanently remembered for that site. This is accomplished by creating an encrypted cookie on the user's machine for that web browser that contains the user's credentials. This cookie persists across different browser sessions and restarts of the machine, but will be set to expire after a set period. The next time the user accesses the website, the server recognizes the cookie, decrypts it to obtain the user's credentials and completely bypasses the login screen after validating them successfully.

Form-Filling SSO

Form-filling allows for the secure storage of information that is normally filled into a form. For users that repeatedly fill out forms (especially for security access), this technology will remember/store all relevant information and secure it with a single password. To access the information, the user only has to remember one password and the Form-filling technology can take care of lling in the forms.