

A BIO-key® Solution.

Tech Brief

SAML Single Sign-On for Higher Education

Table of Contents

Summary	3
The Basics	4
Identity Federation	4
Security Assertion Markup Language (SAML)	4
PortalGuard's SAML-enable Portal	6
Usage Scenarios	6
Benefits	8
How it Works	9
SP-Initiated SSO	9
IdP-Initiated SSO	11
Configuration	13
Deployment	14
IIS Installation	15
System Requirements	15
Alternative SSO Methods	16
Platform Layers	18

Summary

Putting an end to faculty and student complaints about the requirement to remember multiple passwords is an objective at many universities. With multiple web applications being accessed, IT staff also struggle to manage multiple user repositories. For example, when a password changes in one repository, it typically isn't updated in the others. This can lead to security and support issues that make it even more difficult to implement a password security policy across many systems.

To solve these issues you may look towards Single Sign-On (SSO) as a solution to eliminate multiple password prompts and streamline access for students and faculty. However, many SSO solutions are costly and difficult to implement to effectively handle all access scenarios. Integration is especially difficult when attempting to allow the Single Sign-On experience to continue for external users, such as commuting or roaming students and faculty, who all want seamless access to hosted web applications.

For example, if a student logged into their BlackBoard account to view an online class posting, and then decided to access their Outlook Web App (OWA) email account, without SSO the student would be prompted to login again to the OWA application. With SSO in place, the students would be able to enter the OWA cloud-based application without being prompted again. This type of integration is essential for providing students/faculty with seamless application access as well as maintaining unified security and compliance standards to protect university data.

Universities opting to use cloud infrastructures are struggling to achieve seamless access for their students/faculty. Integration is the key challenge -- they are left with forcing their students/faculty and administrators to login multiple times or figure out how to federate the identity in the cloud while maintaining regulatory compliance.

The solution is a product that can create a single or federated sign-in process to handle an organization's multiple cloud applications and provide a centralized point of secure access.

The Basics

Identity Federation

Identity federation is the concept of linking a user's identity across multiple systems or servers. When two servers are federated, the authentication against one can be leveraged to prove the user's identity 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 the clear front-runner.

Using SAML SSO offers a seamless way to provide students and faculty with 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 Multi-Factor Authentication.

Security Assertion Markup Language (SAML)

Originally developed by 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 hosted SAML enabled applications

Once the user requests access to a hosted resource, an online identity provider creates a SAML token containing the user's identity assertions. Once those assertions are validated by the hosted resource the 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 receive a SAML assertion.

Alternatives such as cookie-based Single Sign-On or other established authentication mechanisms such as Kerberos can be provided by PortalGuard to enable SSO for non -SAML applications

PortalGuard's SAML-enabled Portal

PortalGuard's SAML Identity Provider (IdP) acts as a SAML based portal - that uses a single set of credentials for the portal login itself and then grants access to your web-based applications, such as the BlackBoard Learn platform. When using SAML the student/faculty member will no longer be prompted for multiple passwords. As a result, administrative and IT costs associated with performing password resets in several places, synchronizing numerous sets of password quality rules that may or may not overlap and creating and disabling accounts will be greatly reduced.

The main objective with using PortalGuard's SAML-based portal is to give students/faculty members one place to login using stronger authentication, and thereby granting them access to web-based applications, the university portal and/or to disparate sites where they are authenticated without having to sign on again.

PortalGuard provides seamless integration with web-based applications whether cloud-based, private, on-premise, or behind a firewall.

This integration allows you to use PortalGuard not only as your single authentication point but to provide stronger authentication, including tokenless Two-Factor Authentication and Knowledge-Based Authentication, plus all the necessary Self-Service Password Management functionality needed to increase usability in order to reduce Help Desk calls.

Although many web-based applications are already SAML-enabled, alternatives such as cookie-based Single Sign-On or other established authentication mechanisms such as Kerberos can be provided by PortalGuard to enable SSO for non-SAML applications.

The PortalGuard authentication platform incorporates and supports numerous authentication layers, providing IT staff with the flexibility to meet their requirements in a highly cost-effective manner.

Usage Scenarios

PortalGuard's flexibility allows you to choose the appropriate authentication method for each user, group or application, by leveraging Contextual Authentication. Varying access scenarios in every university drive the need for this type of authentication. For instance, students on campus may only need to provide strong passwords whereas commuting or roaming students are presented with Two-Factor Authentication.

Consistent Authentication Interface

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

Enforcing Self-Service Enrollment

Using SAML SSO offers a seamless way to provide students/faculty with 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 Multi-Factor Authentication.

Multi-Factor Authentication

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

PortalGuard's flexibility allows you to choose the appropriate authentication method for each user, group or application, by leveraging Contextual Authentication.

Benefits

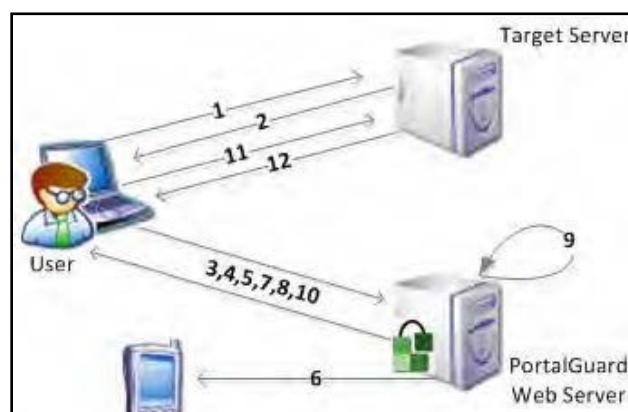
- Eliminate the need to develop and maintain your own portal
- Reduce the number of passwords the student/faculty member is required to remember and manage
- Implement configurable corporate password policies
- Remove the need to manage external students/faculty credentials
- Optionally increase security using any combination of transparent barriers
- Add stronger authentication using tokenless Two-Factor Authentication and/or Knowledge-Based Authentication
- Reduce password-related Help Desk calls related to resets and recoveries

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.

****Note:** for the following steps the term “user” refers to students and/or faculty members.

SP-Initiated SSO

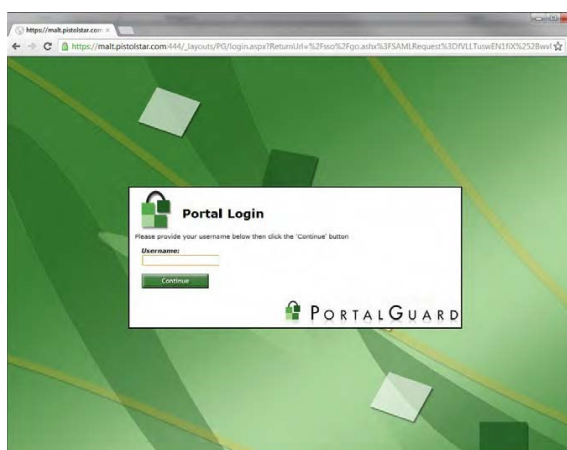


Step 1: The user opens their browser and accesses the target server, e.g. <https://blackboard.abc.edu/webapps/portal/example.jsp>.

Step 2: The target server sees the user has not yet authenticated so it generates a SAML request and returns it and the originally requested URL (the “RelayState”) as hidden input fields in a 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. This demo describes Two-Factor Authentication (2FA).

Step 4: PortalGuard’s login screen is presented to the user. The user enters their username and clicks “Continue”. Note: This login screen can be fully customized to match your organization’s branding, creating a seamless experience for the user.



Step 5: The user is then prompted for their normal password and clicks “Log On”.



Step 6: PortalGuard validates the username and password in real-time then sends a random One-Time Password (OTP) to the user’s mobile phone in the form of an SMS. Note: PortalGuard can send the OTP via SMS, email, Skype, printer or transparent token.

Step 7: The user is then prompted to enter the OTP.



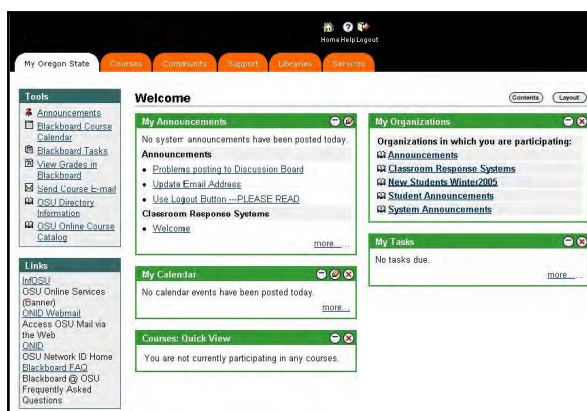
Step 8: The user enters the OTP they received and clicks "Log On".

Step 9: PortalGuard validates the OTP and the user has now established a session with the PortalGuard web server.

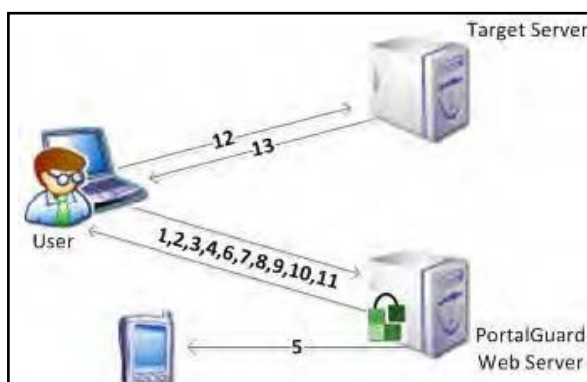
Step 10: The original SAML request is now serviced by the PortalGuard IdP application. It generates a SAML response and sends it and the "RelayState" back to the end user wrapped in a HTML form.

Step 11: Javascript in the response automatically submits the form to the target server's Assertion Consumer Service (ACS).

Step 12: The target server parses and validates the SAML response. It uses the embedded identity claims to determine the user's identity and allow the user access to the application, in this case Blackboard.



IdP-Initiated SSO



Step 1: The user opens their browser and accesses the PortalGuard IdP application, e.g. "portalguard.abc.edu/sso".

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

Step 3: The user enters their username and clicks "Continue"

Step 4: The user is then prompted for their normal password and clicks "Log On".

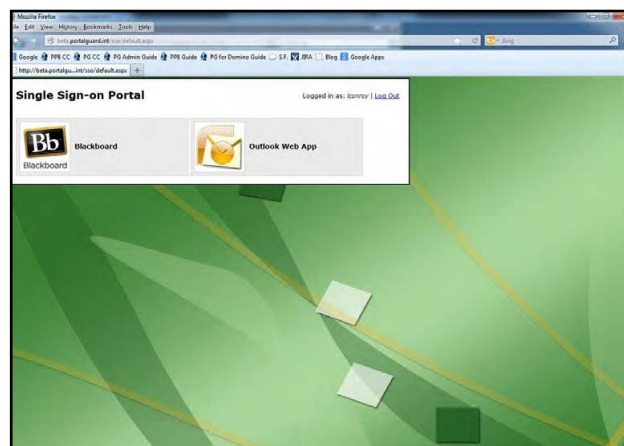
Step 5: PortalGuard validates the username and password in real-time then sends a random OTP to the user's mobile phone in the form of an SMS.

Step 6: The user is then prompted to enter the OTP

Step 7: The user enters the OTP they received and clicks "Log On"

Step 8: PortalGuard validates the OTP and the user has now established a session with the PortalGuard web server.

Step 9: The user gains access to the SAML IdP jump page



Step 10: User clicks a displayed application. *Note: The applications available to the user are configurable by the admin.*

Step 11: The click is serviced by the PortalGuard IdP application. It generates a SAML response and sends it back to the end user wrapped in a HTML form.

Step 12: Javascript in the response automatically submits the form to the target server's Assertion Consumer Service (ACS).

Step 13: The target server parses and validates the SAML response. It uses the embedded identity claims to determine the user's identity and allow the user access to the application, in this case Blackboard.

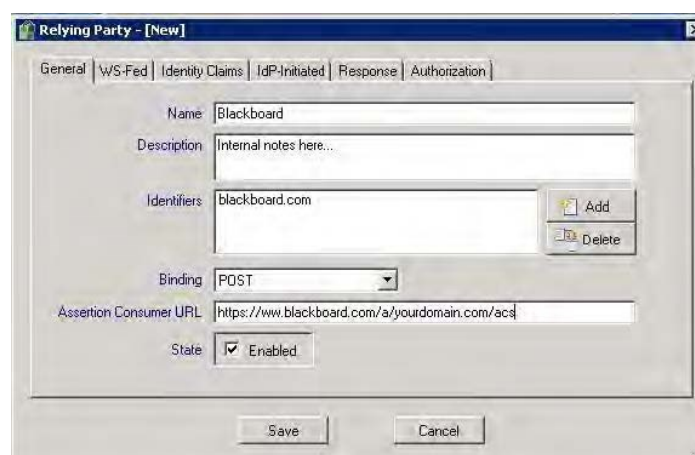
Configuration

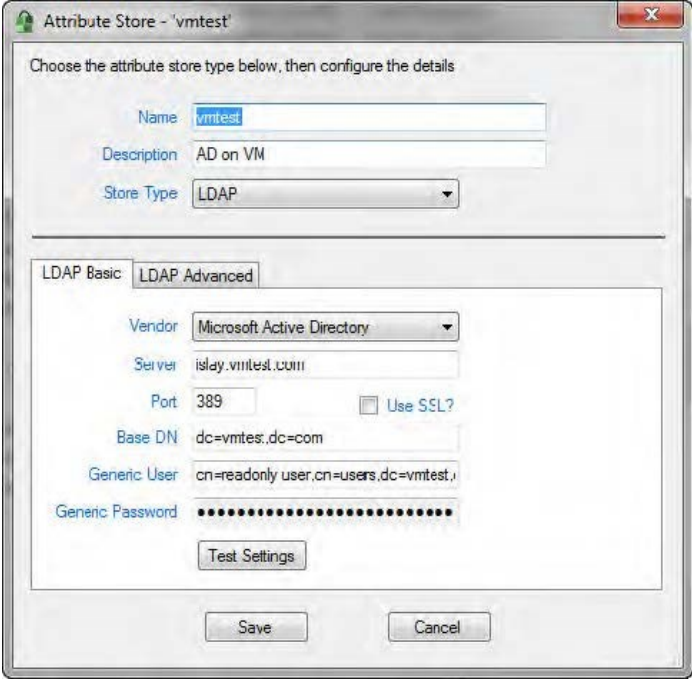
NOTE: All the following settings are application specific, so you can have different permissions for individual users

Configurable through the PortalGuard Configuration Utility:

Main

- Relying parties
- Attribute Stores
- IdP Configuration





Attribute Store - 'vmtest'

Choose the attribute store type below, then configure the details

Name:

Description:

Store Type:

LDAP Basic | LDAP Advanced

Vendor:

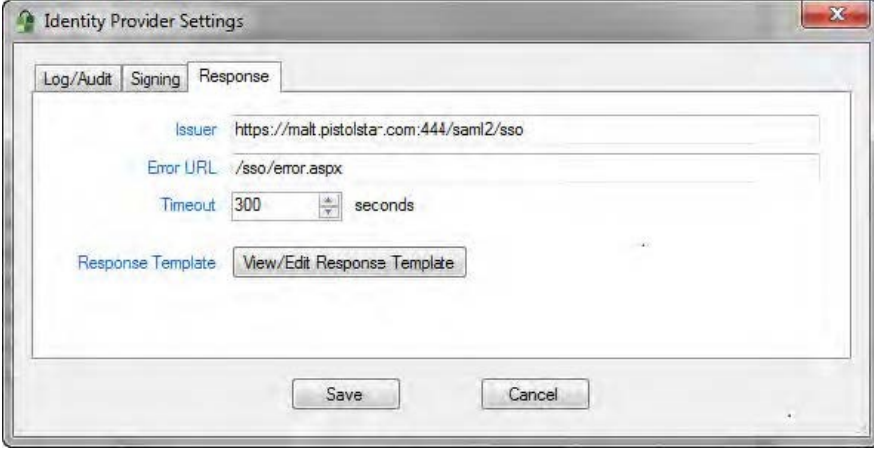
Server:

Port: ☐ Use SSL?

Base DN:

Generic User:

Generic Password:



Identity Provider Settings

Log/Audit | Signing | Response

Issuer:

Error URL:

Timeout: seconds

Response Template:

Deployment

Implementation of the PortalGuard platform is seamless and requires no changes to Active Directory/LDAP schema. A server-side software installation is required on at least one IIS server on the network. Additional client-side software is not required. All major Web browsers, including Microsoft Internet Explorer, Mozilla Firefox, Google Chrome and Apple Safari, are capable of performing SAML authentication to Web servers.

IIS Installation

A MSI is used to install PortalGuard on IIS 6 or 7.x. If installing PortalGuard on IIS 7.x/ Windows Server 2008, make sure to have installed the following feature roles prior to launching the MSI:

1. All the Web Server Management Tools role services
2. All the Application Development role services
3. All IIS 6 Management Compatibility role services

The MSI is a wizard-based install which will quickly guide you through the installation.

System Requirements

PortalGuard can be installed directly on the following web servers:

- IBM WebSphere/WebSphere Portal v5.1 or higher
- Microsoft IIS 6.0 or higher
- Microsoft Windows SharePoint Services 3.0 or higher
- Microsoft Office SharePoint Server 2007 or later

The PortalGuard IdP requires the following:

- The target server must support SP-initiated SAML SSO using the SAML v2.0 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.
- 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 (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.

The PortalGuard Web server also has the following requirements on Windows operating systems:

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

PortalGuard is fully supported for installation on virtual machines. Furthermore, PortalGuard can currently be installed on the following platforms:

- Microsoft Windows Server 2000
- Microsoft Windows Server 2003 (32 or 64-bit)
- Microsoft Windows Server 2008 (32 or 64-bit)
- Microsoft Windows Server 2008 R2

If you have a platform not listed here, please contact us at sales@portalguard.com to see if we have recently added support for your platform.

Alternative SSO Methods

This tech brief is intended to highlight SAML-based SSO; however the following Single Sign-On methods are also supported:

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.

Kerberos-based SSO: Kerberos enables a user to log into their Windows domain account and then receive SSO to their 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 which happens automatically in all major browsers using SPNEGO (see below).

Screen Scraping: Screen scraping deals with recognizing and remembering the dialog boxes that prompt users for their credentials. The idea of "scraping" the fields and layout of these dialogs and automatically entering the user's credentials are the basis behind this SSO type.

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).

Smart Card-based SSO: With Smart Card SSO the user is required to enter their smart card into a reader, which then allows them to Sign-On to the desired application. The smart card information is then used to allow SSO to other applications that the user attempts to access.

Biometric-based SSO: Existing biometric authentication can be combined with SSO technologies to further secure and simplify the life of the daily work force. Biometric authentication can include the use of fingerprints, retinal scans, facial scans, hand geometry, and even DNA.

Form-filling SSO: Form-filling allows for the secure storage of information that is normally filled into a form. For users that repetitively fill out forms, especially for security access, this technology will remember/store all this 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 filling in the forms.

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 first 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.

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.

Cross-domain SSO: Credentials may need to be transferred between different servers and domains, which is referred to as Cross-Domain Single Sign-On (CDSSO). By not relying on a master authentication server, CDSSO allows the user to easily be authenticated between separate secure domains, when requesting resources from each.

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.

Session-based SSO: A user's initial logon to a HTTP server through a web browser results in a session token being generated on the server and returned to the user as an in-memory cookie. From then on, SSO is accomplished by presenting the session token to the server as proof of authentication.

True SSO: True SSO is the ability to provide your credentials one time during your computer session and not be asked for credentials again. This is done by leveraging those credentials when navigating to other resources during your session that will also require you to verify your identity.

Platform Layers

Beyond Single Sign-On, PortalGuard is a flexible authentication platform with multiple layers of available functionality to help you achieve your authentication goals:

[Contextual Authentication](#)

[Self-Service Password Reset](#)

[Two-Factor Authentication](#)

[Password Management](#)

[Password Synchronization](#)

PortalGuard: A Contextual Authentication Platform

