TECH BRIEF

PortalGuard®
A BIO-key® Solution

SAML Single Sign-On

Table of Contents

Summary 4

The Basics 5

- Identity Federation 5
- Security Assertion Markup Language 6
- The PortalGuard Identity Provider 7

Flexible Usage Scenarios 8

- Consistent Authentication Interface 8
- Enforcing Self-Service Enrollment 8
- Multi-Factor Authentication 8

Benefits of SAML SSO 9

How SAML SSO Works? 10

- SP-Initiated SSO 10
- IdP-Initiated SSO 12

Deployment 14

Table of Contents

System Requirements 15

- Requirements for On-Premises 15
- Requirements for IDaaS 16
- Common Requirements for the PortalGuard IdP 17

Alternate SSO Methods 18

- WS-Federation 18
- Central Authentication Service (CAS) 18
- Open Authorization (OAUTH) 18
- OpenID Connect (OIDC) 19
- Shibboleth SSO 19

Summary

Many organizations aim to end-user complaints about having to remember multiple passwords. With numerous web applications being accessed, IT staff often struggle to manage various user repositories. A common complication arises when a password is changed in one repository but not updated in the others. This can lead to security and support issues, making it even more challenging to implement a password security policy across multiple systems.

To solve these issues and streamline user access 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 handle all user access scenarios effectively. Integration is especially difficult when attempting to allow the SSO experience to continue for external users—from staff to customers or partners—who all want seamless access to hosted web applications.

Without SSO, for example, a typical scenario might unfold like this: You log in to your locally managed application— with your active directory credentials—and decide to check your email through Office 365. To do so, you must manually log in to Office 365 despite having already authenticated to your local directory. If you decide to also check on your stocks or other information accessed through an external application, you must authenticate again.

With an SSO solution in place, you can check your email 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. This integration is absolutely essential to provide a seamless experience while maintaining high levels of security and both internal and external compliance.

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.

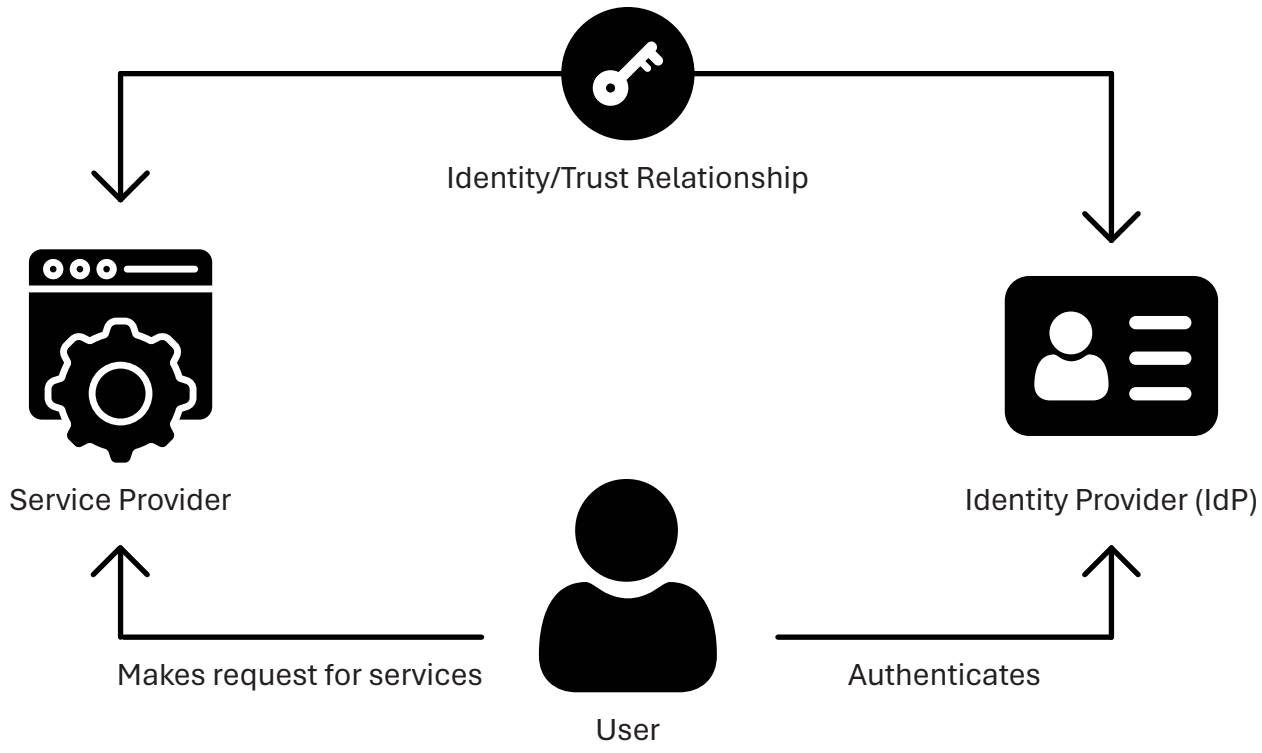
The Basics

Identity Federation

Identity federation is the concept of linking a particular user's identity across multiple systems or servers. When two servers are federated, authentication against one can be leveraged to verify the user's identity on 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 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 systems' security.

Multiple protocols can be used to authenticate from one system to another successfully, 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 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 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-premises" services.

It is typically wrapped in the HTTP/HTTPS protocols, which ensure it can be used by any client device regardless of operating system (e.g., Windows, Mac, and Linux) or architecture (PC, iPad, smartphones).

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 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.

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, multiple login prompts will no longer interrupt the end-user. 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 significantly reduced.

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

Achieving Stronger Authentication

Along with providing a central point of access from which end users can log in to various applications, the PortalGuard IdP can easily be configured for numerous multi-factor 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

Self-Service Password Management

Contextual Authentication

Support for 17 Different Multi-factor Authentication Methods

Passwordless Authentication

NOTE: Although many web-based applications are already SAML- enabled, PortalGuard supports numerous alternative SSO protocols as well, such as: WS-Federation, CAS, OAUTH, OpenID Connect, 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

PortalGuard's inherent flexibility 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 log in 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 required and/or optional multi-factor authentication (MFA) methods, reset or recover their forgotten passwords, and manage their mobile device for use in alternate or Multi-Factor Authentication.

Multi-Factor Authentication

PortalGuard offers a secure and consistent authentication process by enabling direct communication between the end-user and its server. This ensures a high level of security as authentication decisions made by PortalGuard are strictly enforced. Moreover, apps that do not have native support for MFA can leverage PortalGuard to implement a standardized and secure authentication process across all applications.

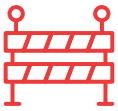
Benefits of SAML SSO

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 Passwordless Authentication for select users or groups of users (e.g., Administrators).

Reduce password-related Help Desk calls related to password and access issues.

How SAML SSO 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 stands for Service Provider and is the application you authenticate into using PortalGuard. SP-Initiated SSO occurs when a user attempts to log in directly to the desired service without first authenticating to the local Identity Provider (IdP). If the user has not yet authenticated, they are redirected to the IdP (PortalGuard) to first authenticate, which suffices 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. https:// mail.google.com/a/example. com

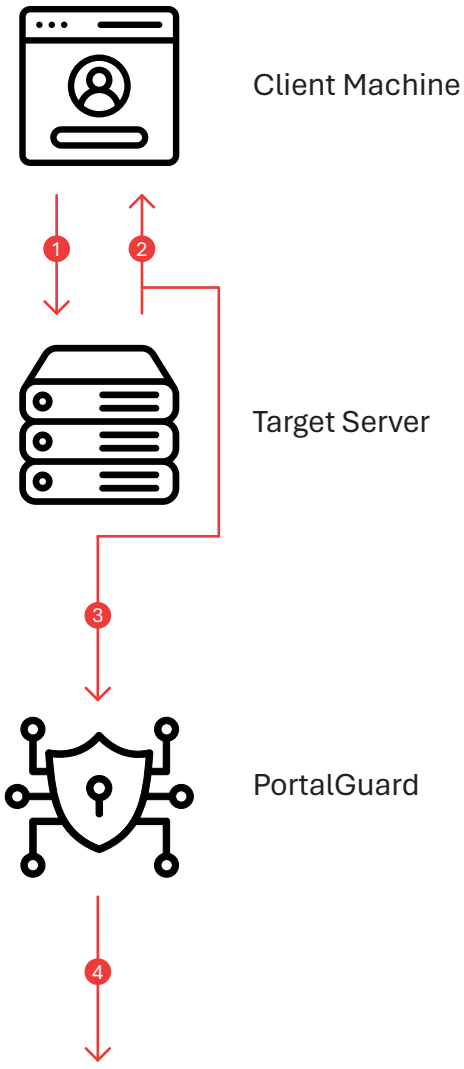
STEP 2: The target server sees that the user has not yet authenticated; it generates a SAML request and returns it alongside the originally requested URL (the “RelayState”) to the client machine as hidden input fields in a HTML-form response.

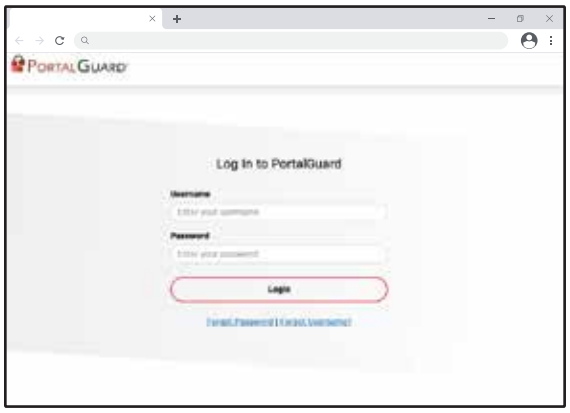
STEP 3: JavaScript in the response automatically submits the form to the PortalGuard Identity Provider (IdP).

Note: The user can be forced to log in using any of PortalGuard’s MFA methods.

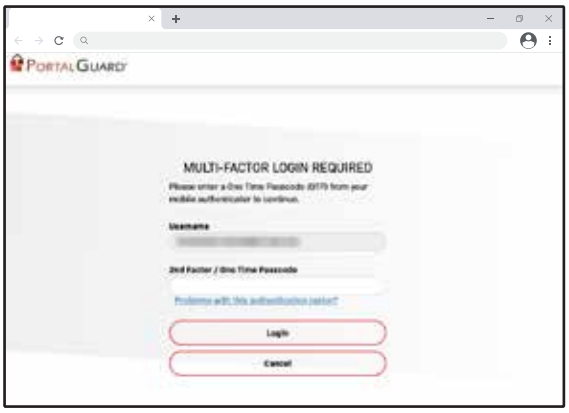
STEP 4: The user is presented with the PortalGuard login screen. The user is required to complete MFA if enforced by the security policy.

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.





PortalGuard login screen



PortalGuard MFA screen

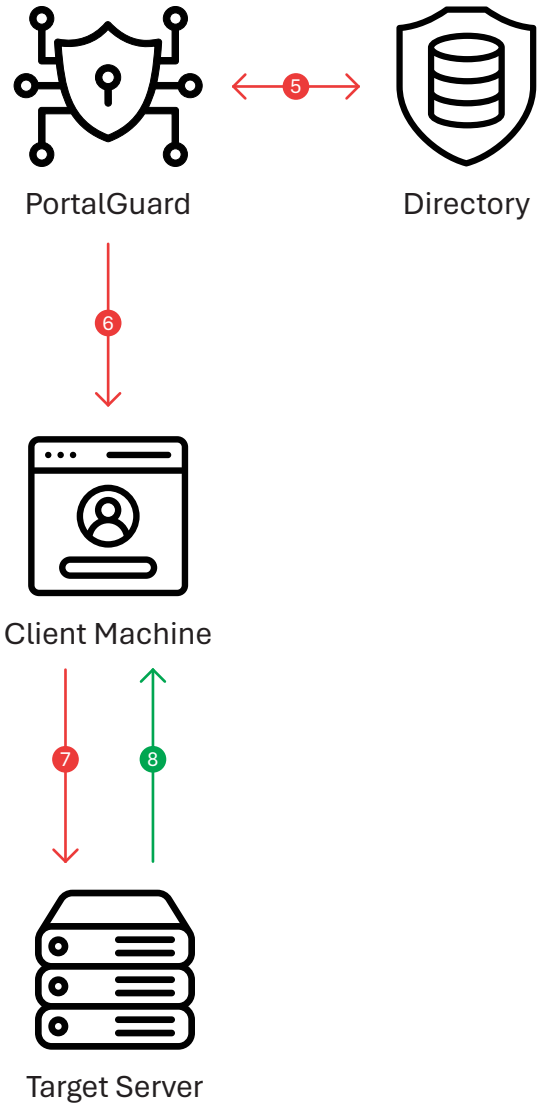
STEP 5: PortalGuard validates the submitted username and password against the appropriate directory in real-time. If correct, the client machine will have established a session with the PortalGuard web server.

STEP 6: The PortalGuard IdP now services the original SAML request. It generates a SAML response and sends it 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.

Note: All SAML Requests and SAML Responses are digitally signed with a certificate to ensure validity.

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

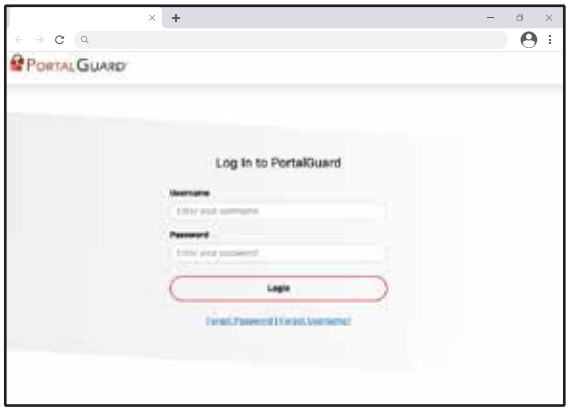
IdP stands for Identity Provider, and PortalGuard acts as one. IdP Initiated SSO typically occurs when a user accesses a locally managed jump-page. By authenticating directly to the IdP first, the user can choose from a host of services to access without the need to input any additional credentials.

It is important to note that not all applications support the IdP-Initiated SAML flow. Please check the application's documentation on SSO first.

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 Identity Provider web page directly. (e.g., https://portal.acme.com)

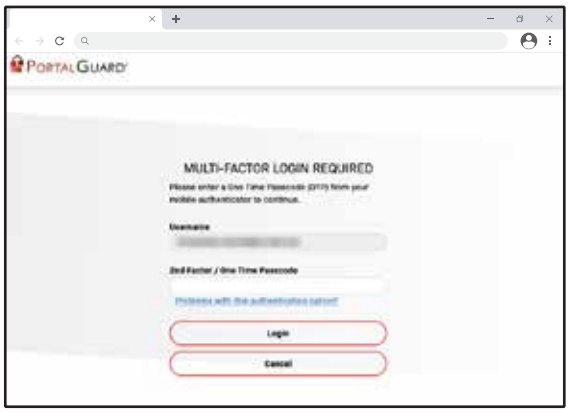
STEP 2: If the user does not already have an active session, the PortalGuard login screen is presented to them.



PortalGuard login screen

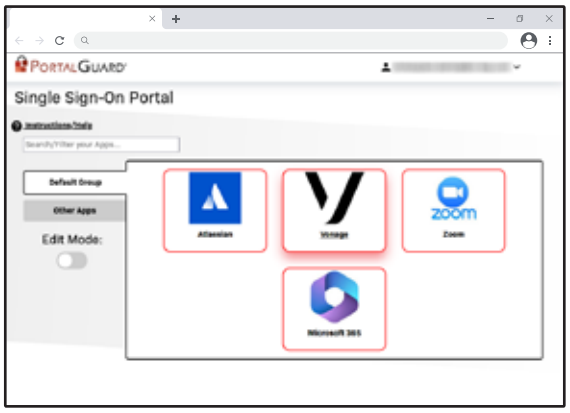
STEP 3: The user enters his/her username and password and clicks "Login." They are then prompted to complete an MFA if required.

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



PortalGuard MFA screen

STEP 5: Depending on how PortalGuard is configured, users are either brought directly to the SSO Jump Page or must go to it using the navigation menu in PortalGuard.

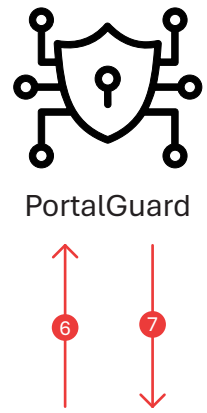


PortalGuard SSO Jump Page

STEP 6: The user clicks a displayed application.

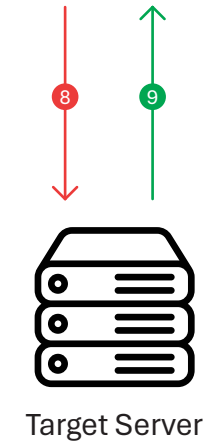
Note: The applications available to the user depend on the permissions configured by the administrator. Whitelisting can be done by individual username, group, or OU.

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).

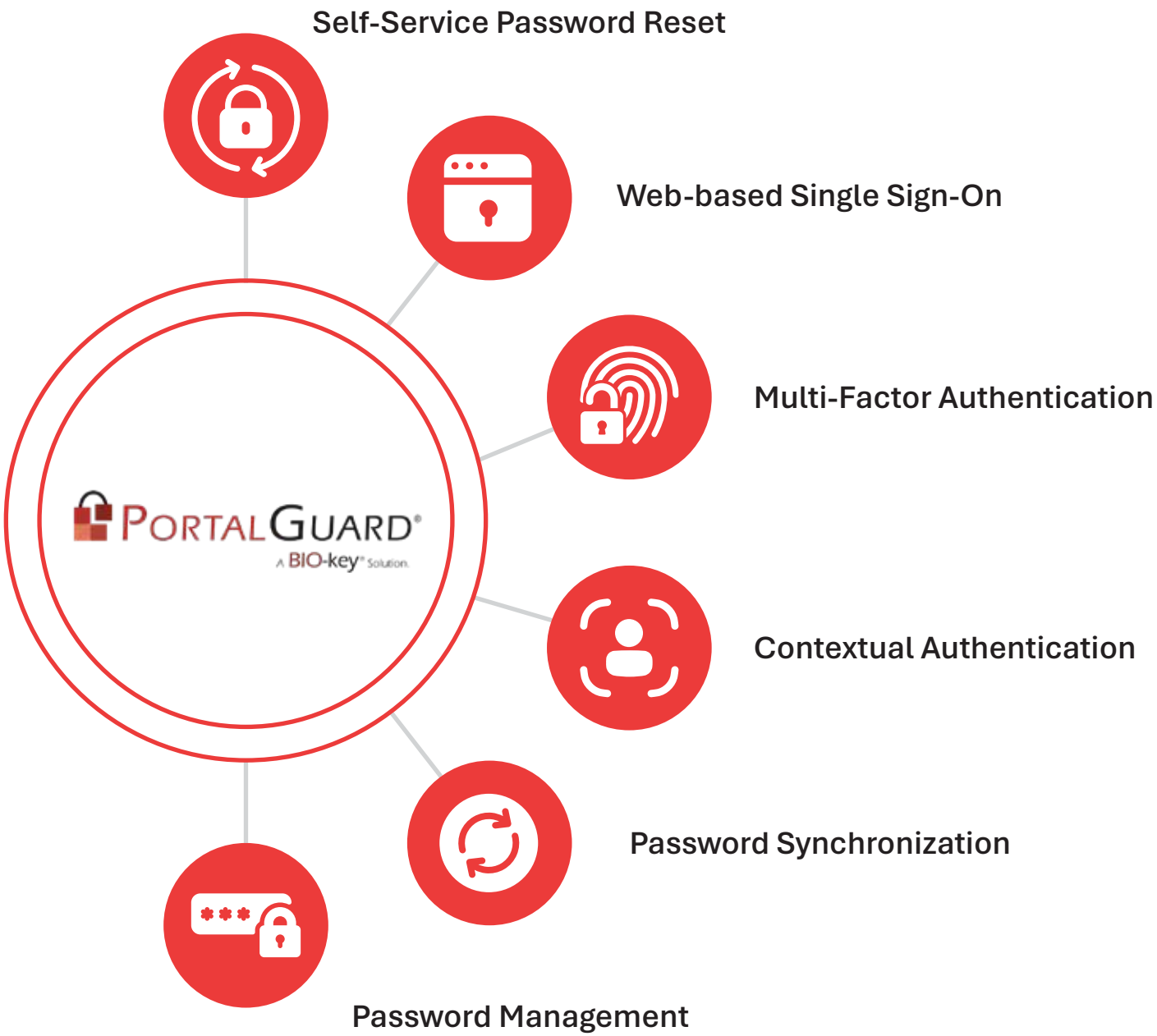
NOTE: There is no SAML Request in this scenario. The SAML Response is sent unsolicited to the target SP.



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 that 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:



System Requirements

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

Active Directory Requirements

- PortalGuard supports Active Directory domain and forest functional levels of 2016 or greater.
- PortalGuard requires SSL LDAP (port 636) to be enabled.
- It is recommended that you organize your users and computer accounts into dedicated organizational units in Active Directory.

PortalGuard On-premises

Server Requirements

The installation requires a single server where the PortalGuard software is installed. This server will house the PortalGuard software, SQL server, and WEB-key server (optional) roles.

The following requirements are needed:

- BIO-key recommends installing all PortalGuard server components on a single physical or virtual server.
- PortalGuard Server must have Windows Server Standard 2016 or newer installed and fully patched.
- It is recommended that the PortalGuard server have Internet access for the software installation process. Doing so makes the software licensing process more straightforward by avoiding the manual authorization process.
- The PortalGuard solution can only be installed on a member server; it is not recommended that it be installed on a domain controller.
- The PortalGuard member server must meet the following hardware requirements: 64-bit dual processor, 16GB of RAM, and 100GB of hard disk space.

SQL Server

- BIO-key recommends SQL Express for all POV environments. SQL Express must be installed on the PortalGuard server before the scheduled installation date.
- It is recommended that SQL Express be configured to limit the amount of RAM available to the SQL service. Failure to configure SQL properly can result in performance problems in the environment.

For more information, see this Microsoft article: https://docs.microsoft.com/en-us/sql/database-engine/configure-windows/server-memory-server-configuration-options?view=sql-server-2017.

- SQL Management Studio is required in the POV environment to facilitate solution management and configuration. Please make sure SQL Management is installed and accessible before the installation date.

See this article for more information:
https://docs.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms?view=sql-server-2017

- SQL Express must have Mixed Mode authentication enabled. Additionally, SQL must listen to port 1433 to function with the software. These options are not configured by default in SQL Express. See the install procedure in the software download package for more information on properly configuring SQL.

See this article for more details:
https://docs.microsoft.com/en-us/sql/database-engine/configure-windows/change-server-authentication-mode?view=sql-server-2017.

PortalGuard IDaaS

PG Connect:

- Windows Server 2016 or later
- 64-bit OS only
- Windows 10 or later
- .NET Framework 4.7.2 or later installed
- 8GB of memory (16GB for 500K+ users)
- 30GB of free disk space
- Direct network connectivity to 1 or more Domain Controllers over Port 636/tcp

A single AD service account for searching and updating AD:

- The account can be a standard "Domain User"
- The account must be delegated these permissions in AD's "Delegate Control" wizard
- create, delete, and manage user accounts
- Reset user passwords and force password change at the next logon

DC Connect:

- Windows Server 2016 or later
- 64-bit OS only
- 8GB of memory
- 10MB of free disk space
- Visual C++ Redistributable 2015-2022, 64-bit
- .NET Framework 4.7.2. or later installed

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 not to 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 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

WS-Federation

WS-Federation is based on web services standards such as XML, SOAP, and HTTP, making it more adaptable to various application architectures. It operates on a trust model, where a user authenticates with an identity provider (IdP) and receives a security token containing identity claims. This token is presented to relying parties (RPs) for accessing protected resources. On the other hand, SAML SSO focuses on web browser-based SSO and uses XML-based assertions. It employs an assertion-based model, where the IdP issues a SAML assertion containing the user's identity information, which is then sent to the service provider (SP) for validation and authorization.

Central Authentication Service (CAS)

Central Authentication Service (CAS) operates on a web-based architecture and leverages a centralized server to handle authentication requests from multiple applications. It uses a ticket-based mechanism, where a user authenticates with the CAS server and receives a ticket that represents their authenticated session. This ticket can be presented to various service providers (SPs) to gain access without the need for repeated authentication.

Unlike SAML, which relies on XML-based assertions and operates on a trust model, CAS focuses on a centralized authentication model with ticket exchanges. CAS is typically used in web applications and supports a wide range of authentication mechanisms, including username/password, LDAP, and more, making it flexible and adaptable to various authentication systems. Additionally, while SAML SSO is a standardized protocol for browser-based SSO across different platforms and domains, CAS offers a more lightweight and customizable approach for web-centric SSO implementations.

OAUTH

OAuth (Open Authorization) is primarily designed for authorization rather than authentication, allowing users to grant access to their protected resources without sharing their credentials. It enables users to delegate access to a client application by obtaining an access token from an authorization server. This access token is then presented to the resource server, which verifies it and grants access to the requested resources.

OpenID Connect

OpenID Connect (OIDC) builds upon the OAuth 2.0 framework and provides a standardized way to authenticate users and obtain their identity information. OIDC utilizes JSON Web Tokens (JWTs) to securely transmit identity assertions between the identity provider (IdP) and the service provider (SP). It enables users to authenticate with the IdP and receive an ID token that contains user information and authentication details. The ID token can be verified by the SP to authenticate the user and authorize access to protected resources. Unlike SAML SSO, which relies on XML-based assertions and is primarily designed for web browser-based SSO, OIDC is more lightweight and suitable for modern web and mobile applications.

Shibboleth SSO

Shibboleth is an open-source software system that provides SSO capabilities and identity federation using the SAML protocol. It enables organizations to establish trust and securely share user identity information across multiple applications and domains. Shibboleth acts as an identity provider (IdP) and relies on SAML assertions to authenticate users and assert their identity to service providers (SPs). It offers features like attribute-based access control, privacy protection, and fine-grained authorization policies. While Shibboleth is based on SAML, it differs from traditional SAML SSO implementations in that it provides a more comprehensive framework for identity federation and attribute sharing. Shibboleth supports a variety of authentication methods, including username/password, X.509 certificates, and integrated Windows authentication. It also offers integration options with existing identity management systems and provides libraries and plugins for easy deployment.