1. **Introduction to Ping Access**

* Ping Access is used to protect sites, APIs, and other resources using rules and other authentication criteria. It allows both internal and external users to access applications securely.
* Ping Access is one of the alternatives to the Web Access Management Systems.

1. **Ping Access V/s Ping Federate**

* Ping Federate takes care of authentication whereas Ping Access takes care of authorization.
* When an application is configured with SAML, we do have access to whole application but Ping Federate Cannot differentiate the access.

1. **Why Ping Access?**

* Some Applications that don’t support SAML or may need granular access control. For that purpose, Ping Access is required.
* Ping Access can be deployed as a reverse proxy for all traffic to the application. After authentication by PingFederate, Ping Access can provide session data to the application through HTTP headers, JWT tokens or proprietary token injection.
* When combined with PingFederate, Ping Access enhances security by applying resource policies (URL) and transaction-specific policies with a wider range of contextual data used for authorization.
* Ping Access can validate authentication tokens by communicating with PingFederate in predetermined time intervals.

**Note:** First, user needs to be authenticated using Ping Federate. So, authentication requests is sent to Ping Federate and Ping Federate sends the authentication response. Virtual hosts will generate a URL for which application configuration to be triggered.

1. **Why Ping Access is a Policy Engine?**

Policy Engine is an enforcing mechanism that allows to enforce certain rules. Hence, Use of Ping Access is to enforce certain set of rules to provide access to resources for some set of users. That’s why Ping Access acts as Policy Engine.

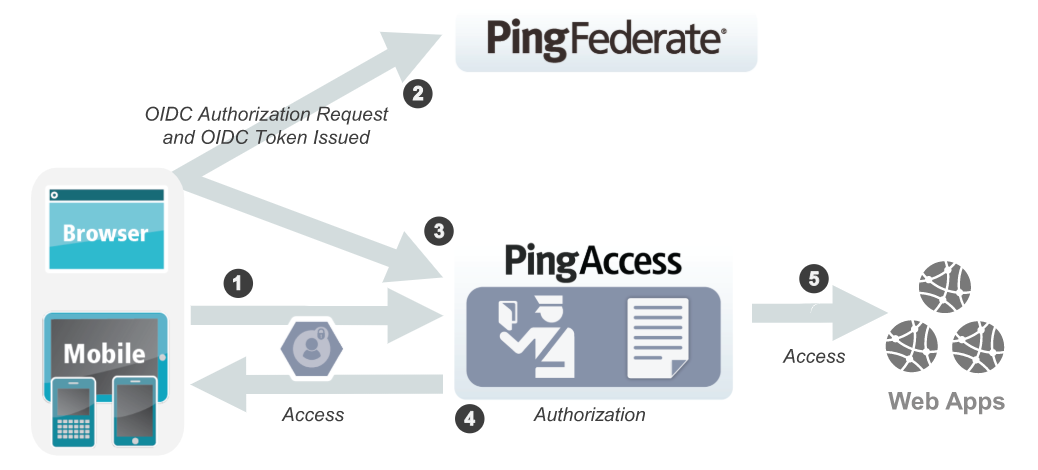
1. **Need for OIDC in Ping Access**

* SAML allows to protect your application but it does not allow to have granular level of control.
* OIDC allows to authenticate user then authorization occur using OAuth.
* OIDC works in integration with OAuth

1. **Terminologies in Ping Access**
2. Virtual Hosts: Virtual Hosts is the receiving end which receives the request in Ping Access. For user point of view, it is the URL for which application need to be triggered.
3. Web session: Web Sessions allows Ping Access to communicate to Ping Federate

* In web session There is a client id which allows to map Ping Access to Ping Federate

1. Token Provider: In Ping Federate, token provider provides the OIDC tokens
2. Site: It points to an application server. Site configuration sends the request outwards from Ping Access. Ping Access, Ping Federate and the application is running on diff servers. So that’s why site is configured which is used to send the user to network URL which has some servers.
3. **How Ping Access Works?**



* When a user requests a Web resource from Ping Access, Ping Access inspects the request for a PA Token.
* If the PA Token is missing, Ping Access redirects the user to an OpenID Connect Provider (OP) for authentication.

Note: OAuth must be configured.

* The OP follows the appropriate authentication process, evaluates domain-level policies, and issues an OpenID Connect (OIDC) ID Token to Ping Access.
* Ping Access validates the ID Token and issues a PA Token and sends it to the browser in a cookie during a redirect to the original target resource. Upon gaining access to the resource, Ping Access evaluates application and resource-level policies and optionally audits the request.
* Note: Ping Access can perform [Token Mediation](https://documentation.pingidentity.com/pingaccess/pa40/pa_c_Token_Mediation.html) by exchanging the PA Token for the appropriate security token from the PingFederate STS or from a cache (if token mediation occurred recently).
* Ping Access forwards the request to the target site.
* Ping Access processes the response from the site to the browser (step not shown).

1. **Gateway V/s Agent Based Deployment**

Ping Access can be deployed using Agents, as a Gateway (or reverse proxy), or using a combination of both. Before deciding on a deployment, it is important to understand the pros and cons of each deployment scenario and determine how they impact your strategy.

**8.1 Gateway**

Pros:

* Fewer number of deployed components that require maintenance
* Independent of target application platform
* No impact on web/app server processing and performance
* Able to work with existing security token types (such as creating 3rd party WAM tokens)

Cons:

* Requires networking changes
* Requires strategy for securing direct access to backend web/app servers (network routing or service level authentication)
* Depending on the application, may require content / request/response rewriting
* Another layer that requires HA/DR planning

**8.2 Agents**

Pros:

* No networking or server level authentication changes required
* Tight integration with web server handling requests
* Scales with application

Cons:

* High cost of ownership when many agent instances deployed, although should be upgradable/patchable independently of Ping Access (policy) server
* Policy evaluation is cached; although it is periodically flushed/re-evaluated (for new sessions, updates to session token, etc.) it isn't quite is "real time" as proxy
* Tight dependency on web server version & platform

Request comes to PING ACCESS and PING ACCESS forwards the request to validate/authenticate the user to PING FEDERATE. PING fedr authenticates the user using OAuth and returns the response to PING ACCESS

If user tries to access one of the resources under resourecs tab then it is checked whether user has legitimate access to that requested resource. When the user is successfully checked then it goes to site url

It is done using web policy

1. Access the url : authentication is configured
2. Authorization : verify that user has valid access to the resource

then user is granted resources using Ping Access.

Virtual hosts: url for which application to be triggered.

Resources : url are provided to access different Ping Accessrts of an application.

Step 2 is Oauth: takes Ping Accessrt of authorization

Step 1 is Oidc helps in authentication

Gateway Based Authentication: User request to Ping Access and Ping Access forwards to Ping Federate.

There are 2 ends in PING ACCESS.

1. Receiving end is VH
2. Sending end is Site

Virtual Hosts: In GBA there has to be a receiving end which receives the Request which is VH in PING ACCESS.

For end users it is just a URL but is a receiving end for PING ACCESS.

Web session: WS allows to Ping Access to communicate to Ping Federate

In web session There is a client id which allows to map Ping Access to Ping Federate

In Ping Federate token provider provides the oidc tokens

Site: it points to an server. Site confn sents the request outwards from Ping Access. site is a sending end. PING ACCESS and Ping Federate running on diff servers. N the application is running on different server. So that why site is configured used to sent the user to network url which has some servers.

There are multiple URL on which app servers are running. Load Balancer determines to select the server. LB redirects the traffic to one of the servers of APP.

So there is one backend URL which consists of multiple URL of servers. And this backend url has a network LB which distributes the servers.

Instead of providing each and every URL of all the servers in PING ACCESS what we do is provide a single backend URL which further consists of servers URL.

How Ping Access communicates with PingFederate?

Agent Based Communication: end user request goes to app server. Within that Ping Accessrticular app server there exits an additional entity called agent which determines whether user have access to app or not.

Inside Ping Access there would be option to select either agent or site (GBA). For each Agent there would be agent id which helps PING ACCESS to identify the agent.