|  |  |  |
| --- | --- | --- |
| Date | Description | Responsible Party |
| 11/30/21 | Initial | Tauk Chang |
|  |  |  |

**Design Patterns for Modern Access Management - Strategic Viewpoint**

**Summary**

The purpose of this document is to provide a simple strategy for the one most important step in securing applications today - that of requiring applications to use MFA.

In general, the current best practice for modern access management for applications is to enable the application to use MFA. To accomplish this, we use an authorization server (Okta) that has MFA enabled or is capable of MFA. Otherwise, we use an Access Gateway (Okta Access Gateway) in between these legacy apps and Okta.

Thus, all applications need to authenticate against an authorization server that is MFA - enabled (Okta). The preferred protocol to be used is OIDC (and pkce).

|  |  |
| --- | --- |
| Goal | Secure Applications with MFA |
| Solution | Migrate applications to use OIDC against an MFA enabled Authorization Server (Okta),  Otherwise use an Access Gateway (Okta Access Gateway) to perform the authentication. |

**I. Two Types of Apps**

We have two types of applications (clients) : OIDC applications vs "Legacy Applications."

1. OIDC Apps (OIDC clients, or able to be migrated to be OIDC clients).

2. Legacy Applications that use protocols such as LDAP, IWA, Kerberos, Login Forms, Headers Auth (Siteminder.)

**A. OIDC Apps**

* For OIDC Apps webapps, we install OIDC libraries into the app (examples are: Okta Auth SDK, IBM Auth SDK, Spring, etc..) to authenticate against an Auth Server, such as Okta.
* For APIs, we put in an API gateway in between the calling application and the Auth Server. The API gateway will perform OIDC authentication against the Auth Server on behalf of the calling application.
* For Mobile apps, we use OIDC / PKCE.
* For SAAS apps, we use OIDC "Federation." In Okta terms, this is considered "social authentication." If the SaaS app does not support OIDC, we can use SAML SSO.

**B. Legacy Apps (non OIDC Apps)**

For applications that cannot be migrated for various reasons, we will use a gateway approach. For Okta, we use the Okta Access Gateway. The legacy app will be repointed to the OAG, which will perform authentication against the Auth Server (Okta instance). Ping offers a simply access gateway product. For Azure Ad, there is no such product. In this case we can use the Big IP F5 APM module, which has similar functionality as OAG. A legacy app will point to the F5 APM which authenticates using SAML SSO against Azure AD.

(Why not put MFA elsewhere? Enabling MFA within an application will require that separate MFA software be used installed on each application.)

(Which vendors can we use? There are many: Ping, Okta, WSO2, etc. .)

**III. Review with Graphics**

A. In the most abstract fashion we an Authorization Server that is used by Applications, which are OIDC Clients.



B. This can be expanded. These OIDC clients are : Web Apps, Mobile Apps (PKCE), the API Gateway, and Federated Apps



C. For legacy apps, we will use the Okta Access Gateway (which is also an OIDC Client.)

