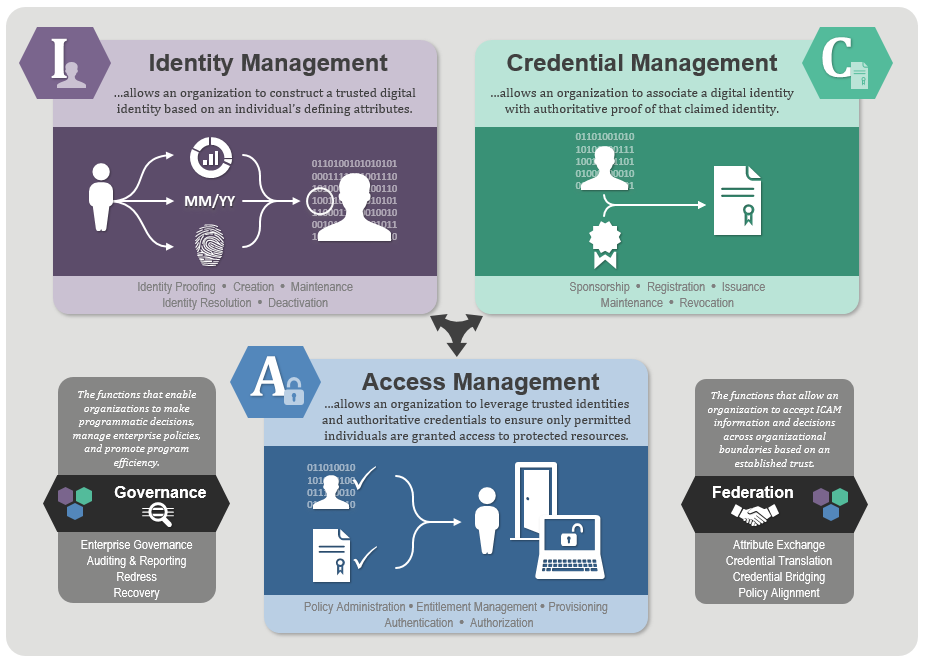
# Identity and Access Management

Identity and Access Management Overview  
The purpose of Identity and Access Management within any organization is to properly construct a trusted digital identity of an individual (whether it is an employee or an external customer) based on individual’s defining attributes such as online credentials, physical finger print, eye retina scan, or personally identifiable information such as birth day, social security number or mother’s maiden name.

The diagram below shows how Identity and Access Management fits within the overall digital security architecture. As noted in the diagram, the functions of Identity and Access Management include:

* **Identity Management (IdM)** – the process of identifying, authenticating and authorizing an individual or group of individuals on an application, system or comprehensive IT environment. IdM provides a process to vet users (identity proofing) and manage the lifecycle of digital identities providing VA applications with a level of assurance that users are who they claim to be.
* **Credential Management (CM)** – the storage and management of end user credentials. CM allows organizations such as VA to associate a digital identity with authoritative proof of that claimed identity.
* **Access Management (AM)** - the process of identifying, tracking, controlling and managing authorized or specified users' access to a system, application or any IT instance that provides capabilities.



Within an API Management Platform, Identity and Access Management platform(s) acts as an “External Identity Provider” used for Identity Management, Credential Management and Access Management. An API Platform does not act itself as an Identity and Access Management (IAM) platform unless it has built-in IAM components. Typically, however, the IAM function is outsourced to a dedicated Identity and Access Management platform, hence it is treated as an “External” Identity Provider for the API Management Platform.

Examples of the best-of-breed Identity Management platforms include Oracle Access Manager, Ping Identity, Okta, Microsoft Active Directory, CA SiteMinder and IBM IAM.

When reviewing the capabilities of any API Management Platform, it is important to consider whether a given API Gateway has an ability to externalize IAM functionality and what standards are supported for that externalization. For example, a consideration should be given to whether a given API Gateway supports authentication via an external identity provider using SAML.

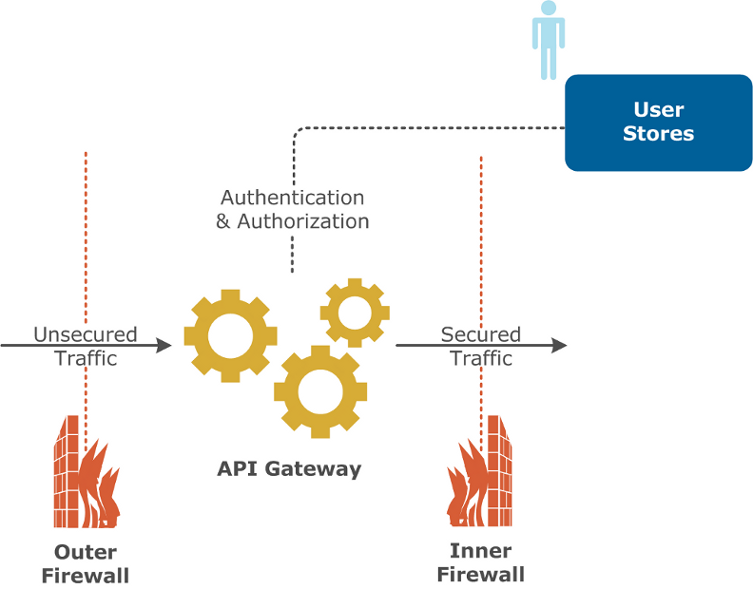
API Gateway Integration with Identity and Access Management

An enterprise API Gateway should be able to support a wide variety of Identity and Access Management user stores. Core user management stores should include LDAP as well as Microsoft Active Directory, where the Identity and Access Management platform would provision and maintain end user identities. The API Gateway should also be able to integrate with a number of Access Control products such as CA SiteMinder and Oracle Access Manager using industry standards (e.g. SAML, OAuth). Identity and Access Management user stores contain some of the most valuable information in an organization.

The diagram below shows how an unsecure traffic hits the API Gateway on inbound API requests, to be protected and secured by the API Gateway and Identity and Access Management User Stores integration. An API Gateway will receive all unsecure traffic requests for a given API, validate the identity of the requestor, and pass the authorized and secured traffic downstream to the APIs. The process of validating and authenticating an identity of the API requestor requires an API Gateway to interact with the Identity and Access Management platform to authenticate and authorize the identity or to validate an existing secure session or token.

For example, if an API request already contains a SAML assertion, Open ID Connect profile, or JWT or OAuth Bearer token, then the function of the API Gateway, at this point, is to request the External Identity Provider to validate the token, this may include making a request to the Identity and Access Management platform for such token validation. Certain security flows such as SAML 2.0 or JWT contain signed assertions (in the case of SAML) or digitally signed tokens (i.e., with JWT). In the case of a digitally signed SAML assertion, the API Gateway can only proxy the SAML assertion request from the Service Provider (SP) to Identity Provider (IdP) for validation. In the case of digitally signed JSON Web Token (JWT), the API Gateway can validate the digital signature via a custom security policy or plugin, provided the Gateway has access to the public certificate for signature validation. If a token is unsigned, however, like in the case of OAuth 2.0 Bearer token, the API Gateway has to simply request the Identity and Access Management platform to determine the validity of the token (based on token expiration, client id/secret, and originating caller address) before allowing the API request to be passed through the gateway.

In the case when the API requestor hasn’t yet been authenticated and authorized by the Identity and Access Management platform, the function of the API Gateway is to request the External Identity Provider to authenticate, authorize and establish a secure session for the requestor, before passing the original API request downstream to the API Management platform.

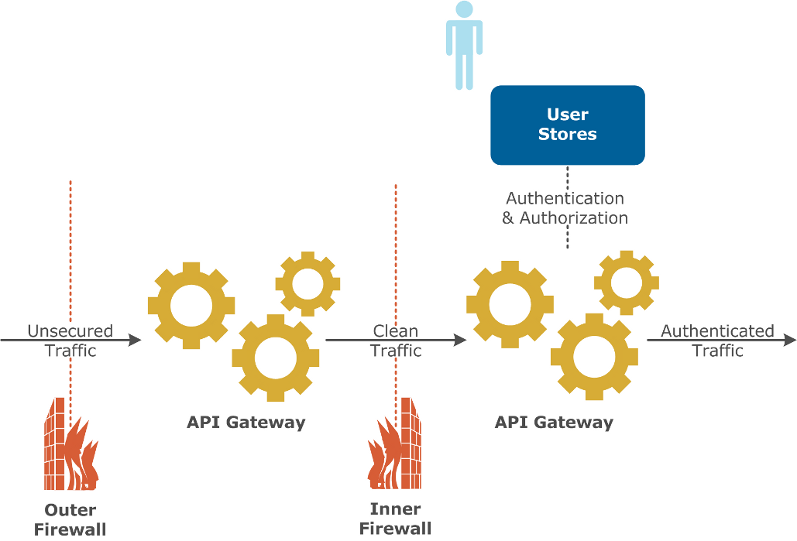


In both scenarios, an unsecure request for an API gets validated by the API Gateway before the request is passed downstream.

A more secure solution would have two API Gateway placements:

* One API Gateway in the DMZ to act as a Web Application Firewall (WAF) to protect internal resources from DDoS, JSON, XML or SQL Injection attacks
* The second API Gateway is placed inside the internal network zone protected by the Inner Firewall. The purpose of that second API Gateway is to interact with the Identity Management user stores and to request identity validation and authorization.

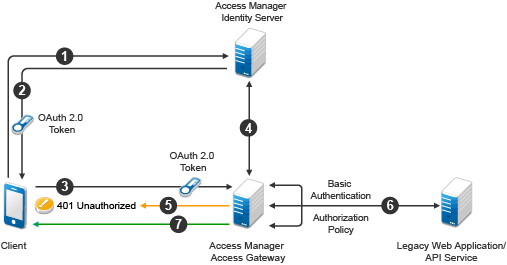
The diagram below depicts this two-API Gateway deployment which is a more secure and scalable solution than having only single API Gateway placed in the DMZ:



#### Identity and Access Management Flow

An API Gateway is the gatekeeper of incoming API traffic to ensure that all requests get validated with the External Identity Provider. The below diagram shows an example of how the identity of a requestor gets verified, a security token issued by the Identity and Access Management Platform, an API request with the token get passed through the API Gateway and finally gets validated and passed downstream to the actual API.

An example OAuth 2.0 flow depicts the purpose of each of the IAM components and how the API Gateway interacts with these components to secure the API invocations by the client.



**Step 1** – A Client authenticates via IAM Access Manager platform by providing end user credentials or some other identity required attributes.

**Step 2** – Once IAM Access Manager validates the identity using its connection to the User Store (e.g., LDAP store), it issues an OAuth 2.0 token in response to the Client.

**Step 3** – The Client then makes an API request passing the OAuth 2.0 Bearer token in the request, as part of the Authorization header.

**Step 4** – The Access Gateway (i.e., API Gateway) requests the IAM Access Manager to validate this token.

**Step 5** – If the token or assertion is expired or invalid, an API Gateway throws back an error response back to the Client (for example, HTTP 401 Unauthorized in the case of a REST/JSON API invocation, which can also be a specific SOAP Fault on security failure for SOAP-based APIs).

**Step 6** – If, however, the token or assertion is validated successfully, the API Gateway validates Authorization policy to ensure that an authenticated user has proper roles and permissions to access the API. Once Authorization policy is validated, the API Gateway passes the request (which is now considered a secured request) downstream to the API Service.

Typically, API Gateway tasks covered in the above steps are implemented via out-of-the-box or custom security policies that perform authentication validation and authorization policy verification, before considering the API request secure. There is a number of default security policies that are provided out-of-the-box with Mulesoft, Apigee, Azure, AWS, Kong, 3Scale or other API platforms.

API Gateway security capabilities cover the following:

* Out-of-the-box pre-built, reusable and configurable security policies which can be easily added to one or more API
* Support for open standards security protocols such as OAuth 2.0, SAML 2.0, Open ID Connect, and JWT
* Enabled security logging and monitoring
* Certification for secure management of PII, PHI, and PCI sensitive end user data