# Cloudflare Access

Transparent Authentication for Web Applications

Oliver Yu (oli@cloudflare.com)
John Graham-Cumming (jgc@cloudflare.com)

## Abstract:

Cloudflare Access is an application on the Cloudflare Edge Network that enforces authentication and authorization for protected resources. Since Cloudflare proxies all traffic to these resources, we can intercept and inject information into the flow to create a layer of authentication transparently, without major integration efforts on the part of the the underlying applications. The power and convenience of Cloudflare Access is amplified when more properties are protected, since Access can provide shared identity across all the applications. We present the architecture and overall flow of the operations for Cloudflare Access and how it balances concerns around security, flexibility, and scalable performance, using a lightweight model, similar to OpenID Connect/OAuth 2.0 and a plugin architecture that delegates authentication to 3rd party systems, like other OAuth, SAML, etc Identity providers.

## Outline:

- Introduction
- System Architecture
- Theory of Operation
    - Key Distribution
    - Configuration of applications and policies
    - Initial Resource Request
    - 3rd Party Authentication
    - Authorized Resource Requests
- Conclusion

## Audience:

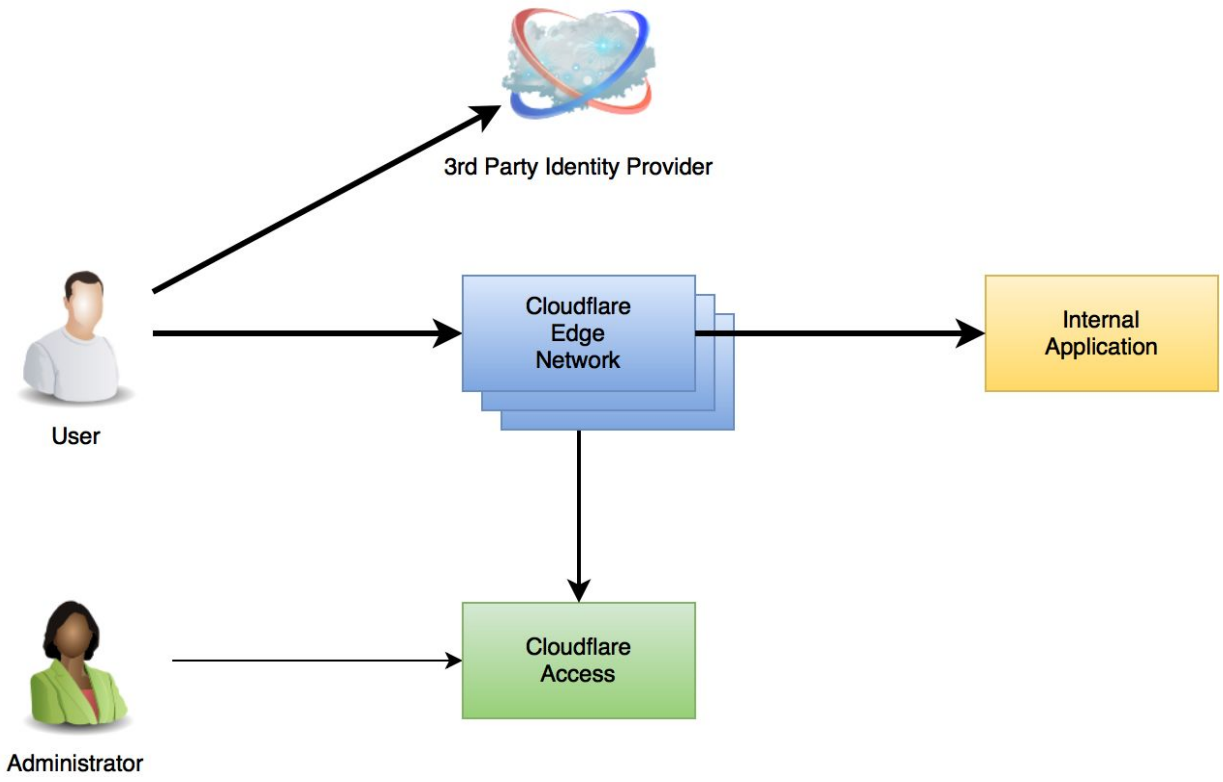
- General Audience, assuming only familiarity of OAuth 2.0/OpenID Connect

# Introduction

As companies move their internal applications into the Cloud and employees grow more increasingly mobile and bring unmanaged devices to work, the traditional VPN (walled garden) model of corporate networks is failing to address the demands of modern workflows. This leads to a world where companies want access to all of a company's internal applications on the open internet. Each application often implements its own simplistic access controls, but this leads to large gaps in maintenance, as authorization for users is distributed across many applications. In addition, these access controls are typically coarse grained and difficult to administer. Alternatively, integrating each of these applications to a company's identity infrastructure is non-trivial and requires developer maintenance for each application involved. With Cloudflare Access, we are changing the way enterprises bring applications to the internet, by offering a secure authenticated gateway in the cloud in front of application resources, like a virtual VPN-less VPN. Employees can use single sign on to login to all of their company's resources. Login once, and navigate seamlessly across tools and internal sites from any device, anywhere.

In order to accomplish this, Cloudflare Access leverages its global network in over 120 data centers to proxy traffic to the internal applications. It uses a subset of the OpenID Connect /OAuth 2.0 flows to secure traffic to the application. The actual authentication and identity information is supplied through a pluggable system of 3rd party integrations that connects to various common Identity Providers through OAuth, SAML, etc. In this interaction, Cloudflare Access acts as an application or resource server to the 3rd Party Identity Provider, while acts as an Identity Provider for all the applications.

## System Architecture




Cloudflare Access consists of software that runs within the Cloudflare Edge Network and the Cloudflare Access service itself. The Cloudflare Edge Network handles the bulk of the traffic and interactions either routes traffic to the Cloudflare Access service itself to handle authorization flows, or forwards the traffic back to the Internal Application once the identity has been established. The Access service also handles all administrative functions. This separation allows the system to scale to very large amounts of traffic with low latency, since much of the traffic can flow through networks closest to the end user, without having to consult centralized resources. The Cloudflare Access service itself exposes a single authoritative service and API that masters all the configuration data and coordinates the distribution of the configuration to the rest of the Edge Network.

## Theory of Operation

Since Cloudflare Access controls the traffic to each individual application as well as the authorization service, we can keep the protection mechanism simple. We leverage a subset of the interactions in OpenID Connect (based on OAuth 2.0). Specifically, we use the implicit grant

flow, with JSON Web Tokens as the access tokens for the applications. From here, we extend the interaction to support a couple of different concerns:

1. We split the role of request/token verification from the process of authorization to leverage our Edge Network infrastructure for performance and scalability.
2. To actually authenticate the user, we have a pluggable system of connectors to 3rd Party Identity Providers. We believe many organizations will already have their own user management systems that they will want to integrate, rather than have to maintain another system of record.
3. We also maintain a access token to a Single Sign On session on an authentication domain, separately from each application's access token. This allow a user to sign into multiple applications through the authentication domain.

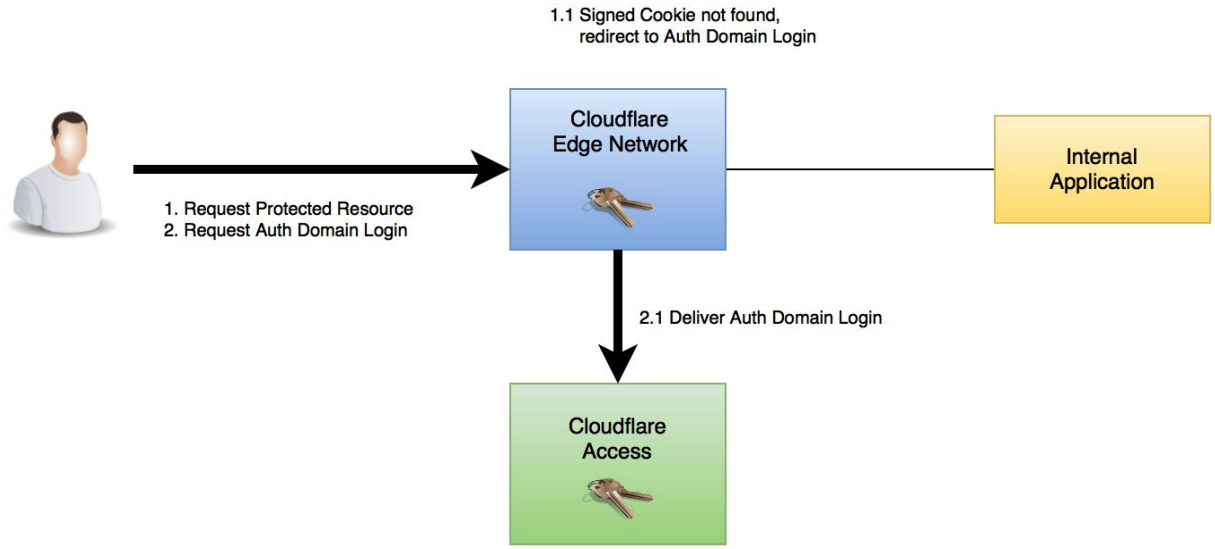
### Key Distribution

Prior to handling any requests, the Cloudflare Access service distributes signing keys from to the Edge Network. We automatically rotate overlapping keys, so that we can handle delays in replicating out to the global network.

### Configuration of applications and policies

A company's administrator then configures an Single Sign On Authorization Domain and connections to 3rd party Identity Providers. The Single Sign On Authorization Domain acts as the application to the 3rd party Identity Providers and we simplify the integration to the 3rd parties through generated instructions and configurations. For example, for OAuth2 connections, we configure the Client ID, Client Secret, and Redirect URL. This is also where the administrator can define URL mappings to various internal applications and access policies for users to all the applications. For example, administrative portion of an site can be a separate application from the main site, so that different classes of users can be granted access. The access policies can take into account the user identity, attributes, and group memberships to deny or grant access to each application.

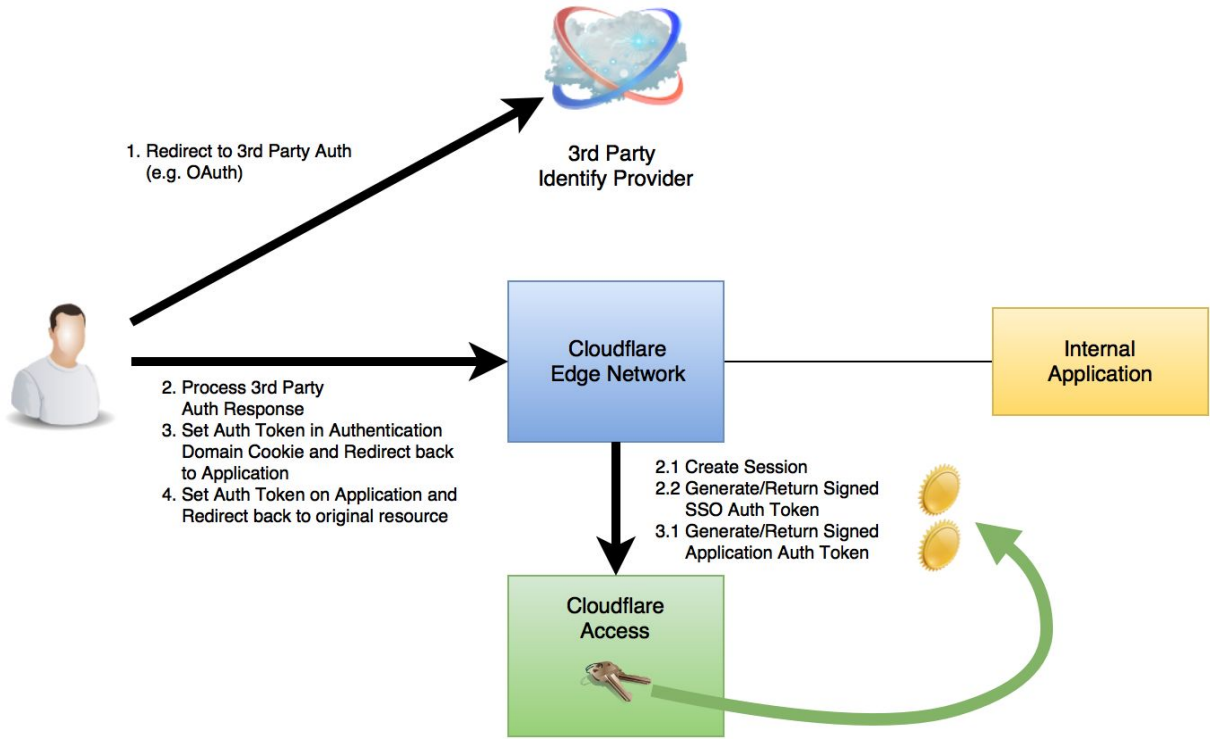
### 1. Initial Resource Request




For unauthenticated traffic, the request goes through the following flow:

1. The request comes into the system to the Cloudflare Edge Network.
    1.1. The request is checked for a signed JSON Web Token in a cookie and redirects to an Authorization Domain Login Page upon failure.
2. The redirected request comes back to the Cloudflare Edge Network at the Authorization Domain
    2.1. The request is routed to the Cloudflare Access service to deliver the login page, which allows a user to select a configured 3rd Party Identity Provider

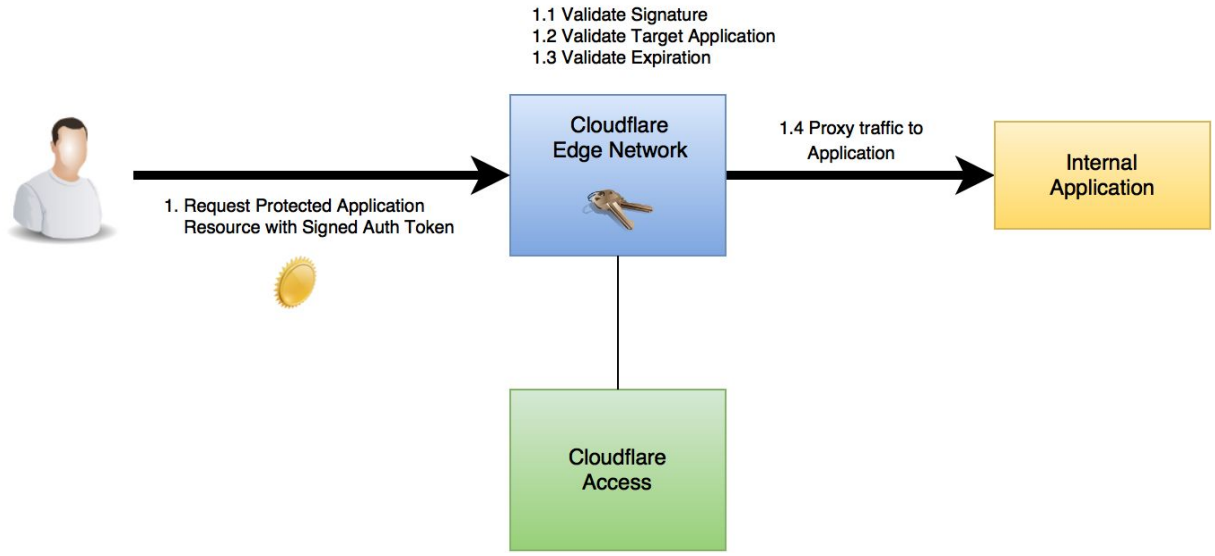
**2. 3rd Party Authentication**




On the Authentication domain, the user logs into a shared Single Sign On session. We authenticate the user by delegating to a Identity Provider. From the login page, after the user chooses the 3rd party Identity Provider:

1. The user is redirected to the Identity Providers, encoded with a return URL to continue the flow within Cloudflare Access.
2. Upon a successful user login, Access confirms the identity and authority of the user and optionally queries for additional information, like user attributes and group membership.
    2.1. A Single Sign On login session is created and stores the authentication information session for later use. Using the login session, Access cross checks this information against the pertinent access policies and makes a final Access/Deny determination.
    2.2. Upon success, we link the application to the session and generate a JSON Web Token for the Authorization Domain, singed with the shared keys, and send it back as a cookie, while redirecting back to the original application domain.
3. At the application domain,
    3.1. Another JSON Web Token for the application is generated and signed with the shared keys and send it back as a cookie on the application's domain, while returning a redirect to the original resource.
4. The user's browser now save the final JSON Web Token that authorizes access to the specified application.

**3. Authorized Resource Access**




Now that the client has the appropriate authentication token, the user can return to the originally requested URL:

1. The request with the JSON Web Token in the cookies is received by the Edge Network
    1.1. The signature on the token is verified with the key that was previously distributed.
    1.2. The target application is verified against the URL path.
    1.3. The token's expiration date is verified
    1.4. All checks pass, so we can forward the traffic to the internal application.

All subsequent traffic to the internal application is similarly secured.

## Conclusion

Cloudflare leverages the interactions that were already pioneered in OAuth and JSON Web Tokens in conjunction with its global Edge Network to provide a scalable, performant, and transparent authentication mechanism for web resources. By marrying this to a pluggable system for delegating authentication to 3rd party identity providers, the result is a flexible system for effortlessly integrating fine-grained authentication to a cohesive set of protected applications.