Reveal BI

Architecture and Security

Abstract

This document describes architecture and security for Reveal BI Server, Reveal BI Applications and Reveal BI Embedded.

Table of Contents

Reveal Architecture and Security

Reveal Cloud (SaaS) Server

Reveal Cloud is an online service that allows you to create BI (Business Intelligence) dashboards and visualizations by pulling data from different data sources. These dashboards can later be shared with other users (inside or outside your company).

This document describes the Reveal Cloud architecture. It's also focused on security and describes how users are authenticated, how data is retrieved from data sources and what information is stored in the cloud platform, including how authentication is handled and stored.

Reveal Cloud Architecture

The following diagram shows the architecture of Reveal Cloud Server:

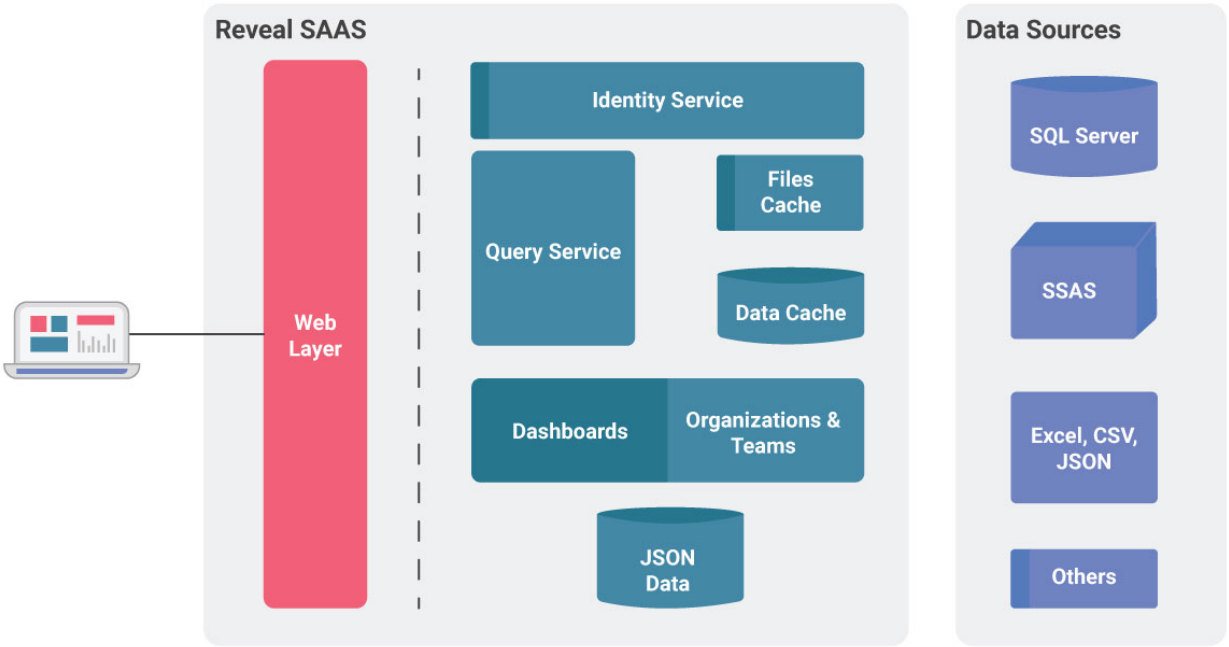

Figure 1 - Reveal Cloud Architecture Diagram

For clarity, the diagram shows a single container and a single instance for each service, but the architecture is actually based on Microservices, which means that at a given moment there might be multiple instances of any of the services, based on the current system load and configuration.

Regarding the services, Reveal uses the **Identity Service** for authenticating users, as it will be described later it supports authentication against multiple providers like Google or Microsoft Office 365.
Dashboards, Teams and Organizations are the services that handle information stored in a JSON database. These services provide storage for dashboards documents as well as membership, permissions and sharing for teams and organizations.
Query Service is the core data transformation component, used to retrieve data from data sources, cache it and transform it according to the visualizations defined in dashboards.
As we'll see later, data and files are cached to improve performance and to implement some of the data transformation features (like joining data form multiple data sources through the Data Blending feature).

Multi-tenancy

The server has the ability to separate data and services based on tenants. A given Organization, for example, might be configured to use a separate storage area, which means all data (Teams, Dashboards and even cached data) will be stored on a dedicated storage and even services at runtime might be isolated. This ensures that performance for that Organization will not be affected by the load of the rest of the system.

Authentication

Reveal Cloud authentication is performed using the OpenID Connect (OIDC) standard. The Identity Service component displayed in the previous diagram provides secure authentication against multiple authentication sources: Google, Office 365 and Infragistics accounts.

Authentication Flow

Reveal authentication follows the standard flow for OIDC authentication as you can see in the following diagram:

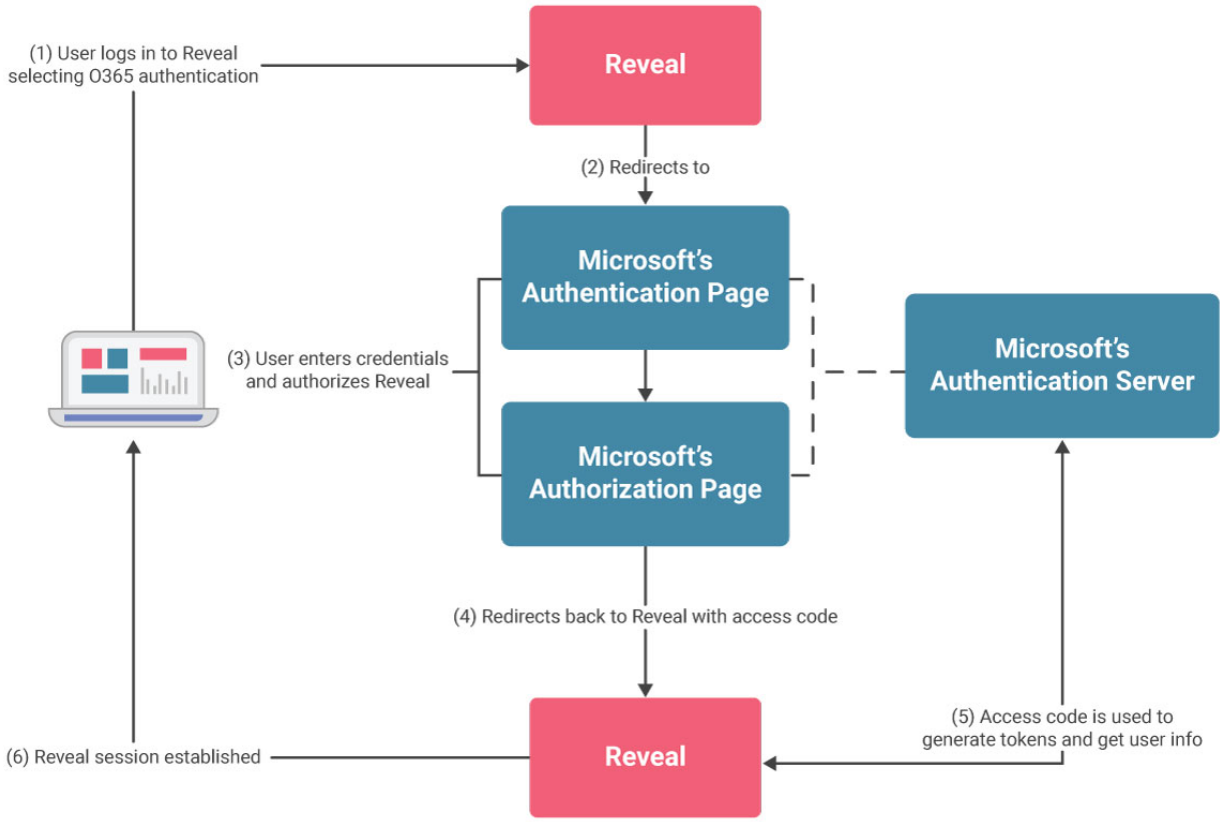

Figure 2 - Sample Authentication Flow Using O365

It's important to note Reveal never has access to the user's credentials as they are entered in a page provided by the authentication provider (Google, Microsoft or Infragistics), the authentication provider then redirects the user to Reveal with an access code (step number 4). Reveal then uses this code to get user information (such as e-mail address that is used to identify the user in Reveal) and tokens that will be used to access some other services (like Google Drive for Google accounts or SharePoint and OneDrive for Microsoft O365 accounts).

Data Retrieval

Reveal supports multiple data sources, such as:

- Relational databases (MS SQL Server, MySQL, and others)
- Data files (CSV, Excel, and JSON from Google Drive, Dropbox, etc.)
- Cloud services (Salesforce, Dynamics CRM, and more)

Reveal takes advantage of the processing capabilities from each provider, for example when getting data from SQL Server filters are applied in the SQL query used to retrieve data, but in the case of Excel files data is filtered as part of Reveal data transformations.

For providers like SSAS (Microsoft Analysis Services) that provides a rich querying and aggregation language most of the data processing is performed in the data source, and only some transformations like calculated fields are performed as Reveal data transformations.

Data Access Flow

The next diagram shows how data is processed for a sample request (a CSV file downloaded from a file provider like Dropbox).

As you can see data flows through a processing pipeline, this pipeline is configured according to the data source in use (CSV in this case) and the transformations applied to it (filtering, aggregation, etc.).

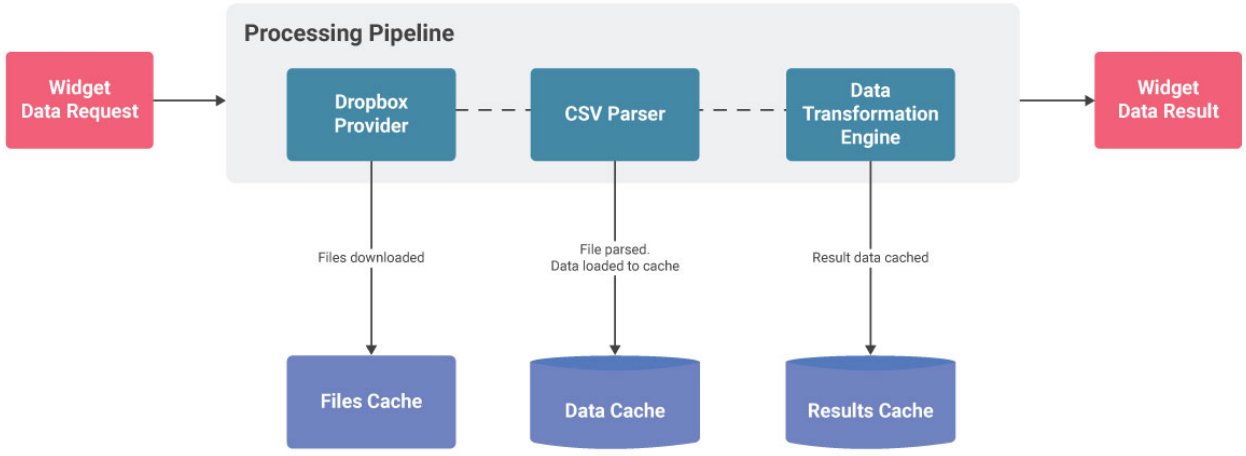

Figure 3 - Sample Data Flow for a CSV File from Dropbox

Data Access Restrictions

As a SaaS solution, Reveal Cloud Server is running on a cloud platform and thus might not have access to data sources located inside your company, like a database server.
You can still access data on those servers using the client applications (Windows, iOS and Android platforms). When trying to open dashboards using “in-house” data in the Web version you’ll see a connection error because the server cannot access data behind your corporate firewall.

In the future, Reveal will include a data agent service that can be installed inside your corporate network and will provide access to your data even when using the cloud platform. In the meantime, if you need to open dashboards in the Web version using data from relational databases (or other in-house data sources) you’ll need to publish them to be accessed from the Internet.

Data Sources Authentication

Each data source provider requires a different authentication method. Reveal supports four types of authentication:

- Anonymous
- User/password
- NTLM: user, password and domain
- OAuth v2

For those providers requiring user/password authentication (like a Web Resource using Basic/NTLM authentication, etc.) credentials are stored in a secure storage, used only in the cloud platform and never sent to client applications.

OAuth Data Sources

For those providers supporting OAuth (like Google Drive, Dropbox, OneDrive and others) Reveal will redirect the user to the authentication page provided by the data provider requesting permissions to access its data.
Once the authentication is complete, Reveal will store the authentication tokens in order to access the data on behalf of the user across all supported platforms.
Please note for OAuth providers Reveal never has access to the credentials entered by the user. Credentials are entered only in the page served by the provider (Google for example) and Reveal receives only a token, that will be securely stored in the cloud platform. This token might be revoked by the user in case the access from Reveal needs to be disabled.

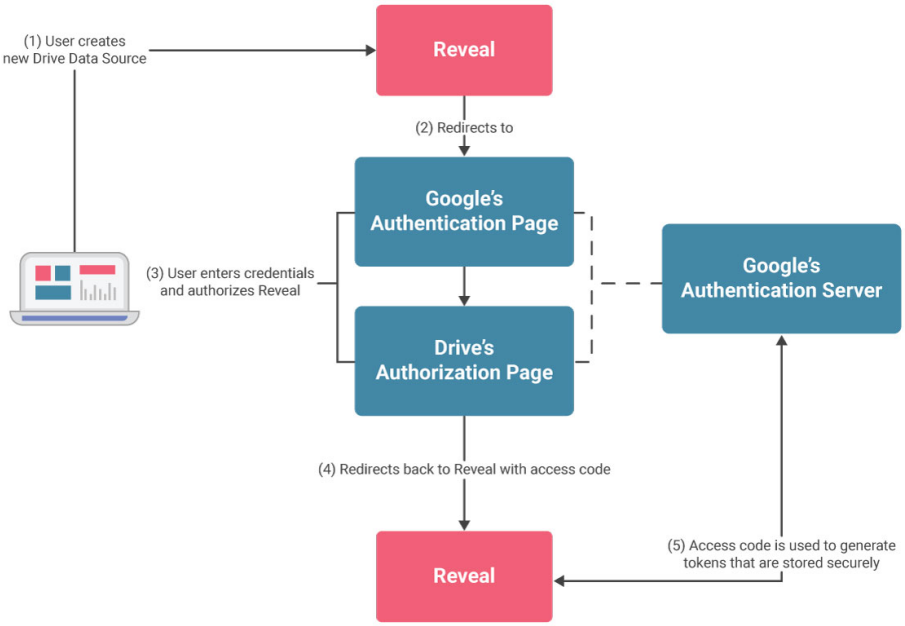

Figure 4 - Sample OAuth Authentication Flow for Google Drive

Data Caching

For performance reasons, and to support some of the advanced data transformation features, data retrieved from external data sources will be temporarily stored securely in our cloud platform. This applies to both data retrieved from databases or online services and files, such as CSV, Excel, and others, or downloaded from content providers including Google Drive, Dropbox, and more.

This data is automatically cleaned up based on expiration settings configured in the dashboard using this data.

Dashboards Storage

Dashboards are JSON documents describing where to get the data from, how to transform it and what visualization to use.
No sensitive data like credentials is stored in dashboards, only data identifying the data source (like URL, host name, etc.), metadata information (like table or field names) and filters (like the list of selected countries) is stored in this JSON document.

Reveal Client Applications: iOS, Android and Windows

In addition to using Reveal within a web browser you can also install native apps for the most popular platforms: Windows, iOS and Android.
Authentication on these apps is performed using the Reveal Cloud platform, with the same OIDC authentication protocol.
Dashboards, Teams and Organizations are also retrieved from the Reveal Cloud platform, granting the user access to the same dashboards used in the Web platform.

Data is directly accessed from data sources and not retrieved from the cloud platform. This means you can access data in your local network, even when that data is not accessible to Reveal Cloud.
If you're on the go and need to open dashboards accessing data from your corporate network, you'll need to use a VPN solution.
As data is directly accessed from these native apps, there's no need for a server component to be installed in advance for you to use Reveal on your device, you just need to install the app on your device and log in using the same credentials you use to access Reveal Web.

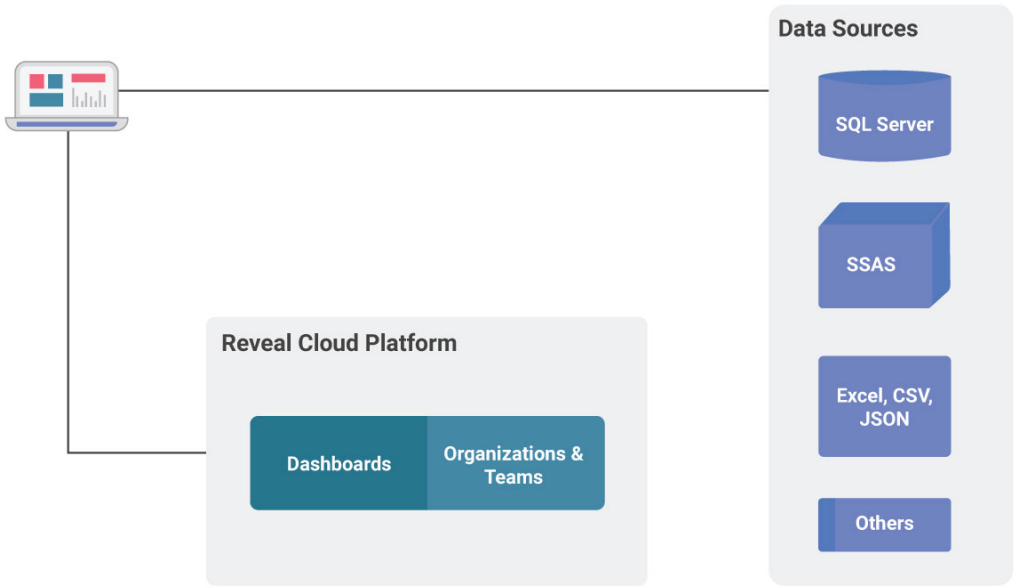

Figure 5 - Reveal Client Applications Architecture

Data Security

To improve performance, Reveal stores cached data locally in the device in order to minimize the number of queries sent to the server or database.
Credentials to access data sources are also stored in the device as they are requested only when the data source is configured or used for the first time. This means that sensitive information is stored in the device.
Reveal, therefore, takes advantage of the security capabilities available on each device.

iOS

In iOS devices, the passwords are stored in the device's Keychain. As long as a passcode is set on the device, cached data and the local database is stored using iOS Data Protection, which means all files will be encrypted as soon as the device is locked. This feature is provided by iOS itself and ensures the best performance.

Desktop and Android

In Desktop and Android, all passwords are encrypted before being stored.
Cached data stored in local databases is also encrypted.
Cache files are not encrypted as there is no feature similar to iOS' "Data Protection." For these platforms, we recommend using MDM platforms, which will help keep data secure.

Reveal Embedded – Architecture

Reveal Embedded allows developers to embed Reveal into their applications, and dashboards can be displayed and even modified by end users.
Reveal SDK can be used to integrate Reveal into applications developed in multiple platforms and technologies: Web, Windows WPF and Windows Forms, iOS and Android.

The containing app can use Reveal SDK to:

- Provide in-memory data to dashboards. If data is already loaded in the containing app there's no need to store it before opening the dashboard, Reveal can use the built-in In-Memory data provider to use that data as the input for the dashboard.
- Configure the dashboard before it gets rendered, for example changing the name of the database or table to use to get the data based on the current user.
- Change dashboard filters before the dashboard gets rendered or even while it's visible. The containing app can use this feature to synchronize filters or selections in the app with the data visualized in the dashboard.
- Get notified when a data point is selected in the dashboard (like a bar in a chart is clicked), for example to display additional information or trigger an action in the app like navigating to a page with more details.
- And much more features you can read about in the Reveal Embedded Dev Guide.

Reveal SDK for Native Applications (Windows, iOS and Android)

When Reveal is embedded in an application (Windows WPF, Windows Forms, iOS or Android), the Reveal SDK is provided as a library or framework that is integrated into the app (the integration steps are different for each platform, for more details check the Reveal Embedded Dev Guide).
The containing app creates a RevealView object configured with the dashboard to render, this view is displayed in the containing app and then a set of callbacks can be used to customize how the dashboard is rendered and what data is used.

RevealView component provides rendering and data transformation capabilities (automatically through the Reveal Engine), but it does not handle the storage of dashboards or credentials. The binary contents (.rdash file) that contains the definition of the dashboard must be provided by the containing app. This allows the container app to handle how dashboards are used and shared by end users (for example they can be downloaded from an internal server, bundled as resources in the app's binary, stored in file system, etc.).

Regarding credentials, when getting data from databases or other data sources requiring authentication usually the containing app already handles these credentials by loading them from configuration files or storing them in a secure storage. So Reveal delegates the storage and handling of these credentials to the container. The app might decide to return internal credentials or to prompt the user for them, if needed.

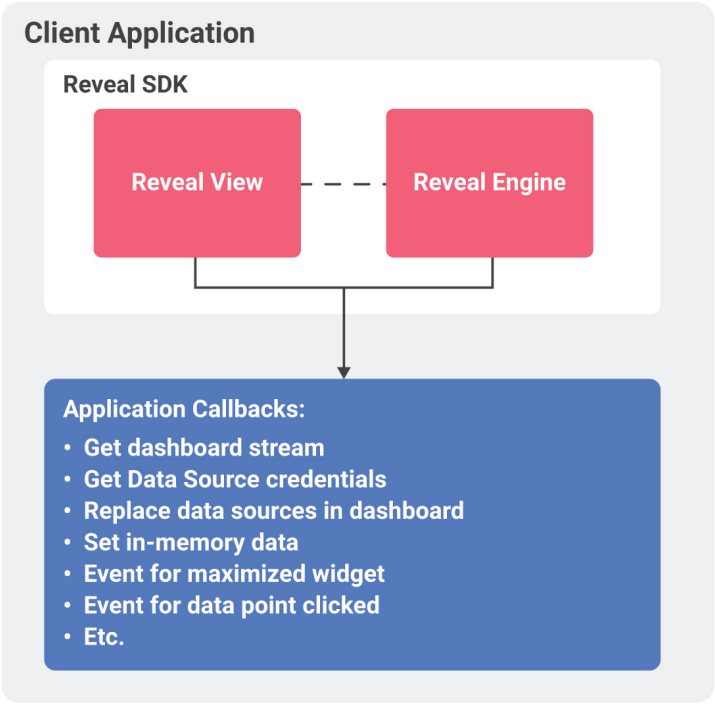

Figure 6 - Reveal SDK for Native Applications

Reveal SDK for Web Applications

When embedding Reveal into web applications, the architecture is a little bit more complex as two components are involved:

- Reveal Client SDK: a set of JavaScript libraries and CSS files that needs to be integrated into the web application. The frameworks supported today are: jQuery, Angular and React.
- Reveal Server SDK: the server-side component to be integrated into the server application, currently this is an ASP.NET Core application using .NET Runtime v4.6.1 or later. In the near future a library using .NET Core will be released.

In the following diagram you can see what's the architecture for a web application embedding Reveal Web SDK.

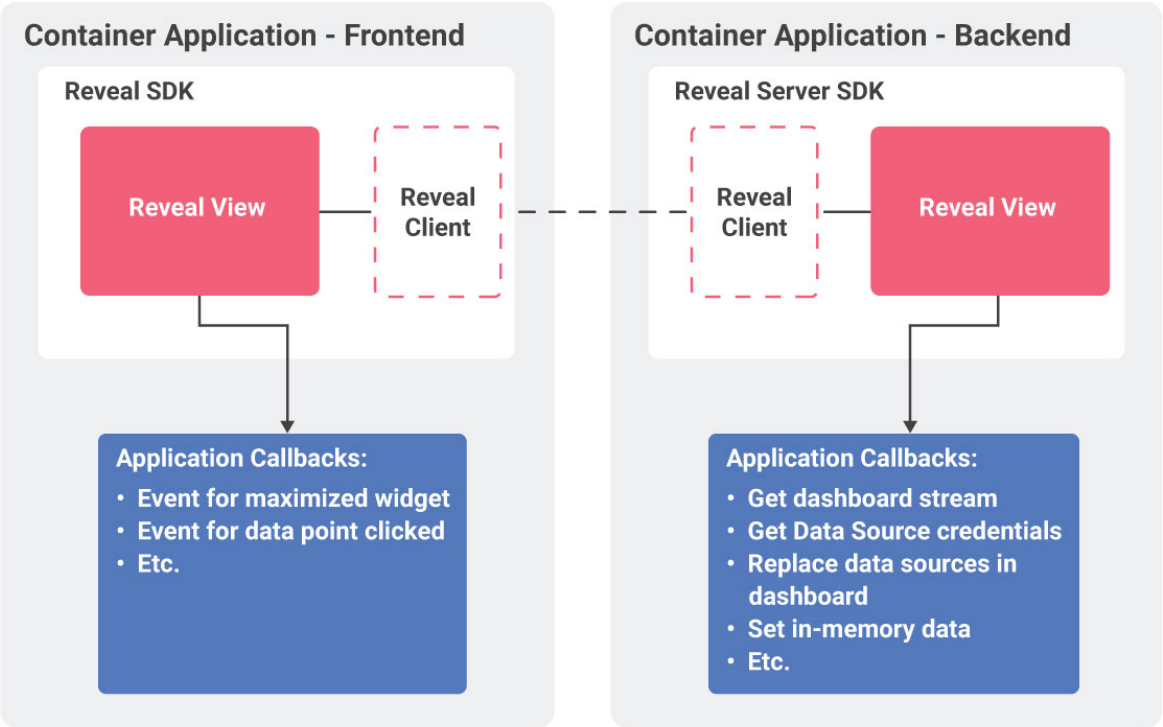

Figure 7 - Reveal SDK for Web Applications

As you can see it works pretty much the same way as it works for native apps with the difference that some of the callbacks are invoked in the client side (like the event sent when a data point is clicked) and some others server side (like the callback to load the dashboard or to provide in-memory data).

Deployment

The SDK server-side component is a set of ASP.NET Core libraries (supporting both .NET Framework and .NET Core), this allows the deployment into both on-premise or cloud applications.
For cloud applications, it can be deployed to multiple cloud containers like AWS, Azure, etc. and multiple platforms like Windows or Linux.

Reveal SDK can also be easily integrated into multi-tenant applications where multiple instances of the SDK component might be running at the same time, each of them configured with its own resources such as cache storage, credentials management, and dashboard processing.