White Paper

Security

Basics, Architecture, Cloud & On-Premises Hosting

Open as App GmbH

www.openasapp.com

Content

Introduction

This document revisits the basic concepts of Open as App, introduces the architecture of the overall system, and elaborates on security concerns. This Whitepaper contains information on

- App Rendering
- App Creation Process
- App Creation Process with External Cloud Storage Providers
- App Creation Process with On-Premises Data
- Authentication Concepts

App Rendering

Before being able to discuss the architecture of the overall system and related security issues, one needs to revisit the most basic concept of Open as App: the dynamic rendering of spreadsheet-based apps on mobile devices.

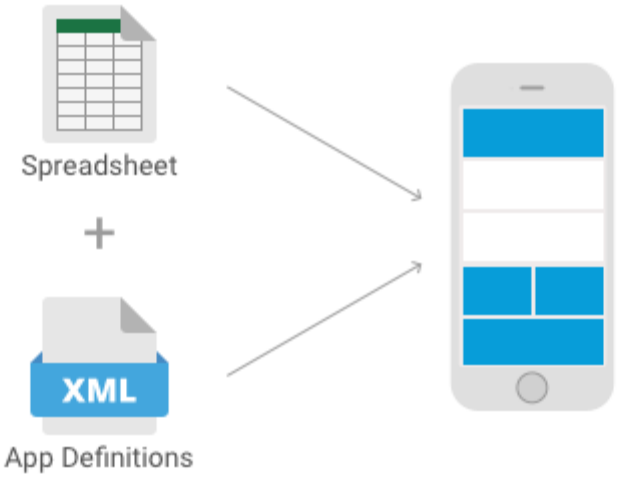

Figure 1: An app is rendered from the spreadsheet and the app definitions locally on the mobile device.

Open as App dynamically renders spreadsheet-based apps on the mobile device on-demand when the user starts an app. Two resources are required for this process: 1) a local copy of the original spreadsheet, and 2) an app-specific *so-called* app definition (see Figure 1). The app definition contains descriptive data of the

app (name, description, icon, etc.) and provides the mobile device with instructions on how to visualize the data from and interact with the spreadsheet. The app definition does not contain any sensible data, it merely describes the user interface and contains references to segments of the spreadsheet; see Table 1 for an exemplary app definition record.

Special Open as App architecture

Any data and logic are directly provided by the spreadsheet itself. This provides a clear separation of layout, data, and logic. Both, the spreadsheet and the app definition, are synchronized to the mobile device and are combined on-demand when the user starts an app.

The special Open as App architecture not only enables offline access but also allows to independently update either of both resources. Any synchronized data is securely stored in encrypted storage on the mobile device, protected by the user's credentials and the Microsoft Data Protection API (DPAPI)[1]. The app is compatible with mobile device management systems like MobileIron and BlackBerry Dynamics[2].

Id	fb864b33-c138-4b16-b582-e6bf25a65895
Name	BMI Calculator
Description	Body Mass Index Calculator
SpreadsheetUrl	https://data.openasapp.net/fb864b33-c138-4b16-b582-e6bf25a65895.xlsx
Layout	...<item caption="Body Height" inputtype="input" contenttype="integer" address="B1">...
Owner	apphub@openasapp.net
PublishMode (internal, not synced)	private
AccessControlList (internal, not synced)	[j.doe@openasapp.net, ...]

Table 1: Exemplary app definition used for rendering a spreadsheet-based app.

App Creation Process

The overall Open as App system consists of three main components:

1) the web-based Open as App Portal for creating, maintaining and distributing apps,
2) the Open as App Client for consuming apps on mobile devices, and
3) the Open as App *Cloud services* wiring both platforms together.

The Open as App *Cloud services* can further be sub-divided into three sub-components:

3a) the *app creation service*,
3b) the *app distribution service*, and
3c) the Open as App *cloud storage*.

Figure 2 illustrates the responsibilities of the system's main components using the example of the three most basic use cases that the typical actor roles, the *app distributor* and the *app consumer will* encounter. Apps are built and published via Open as App Portal and consumed via App Client. All three use cases involve the Open as App Cloud services for creating, storing, and distributing the apps.

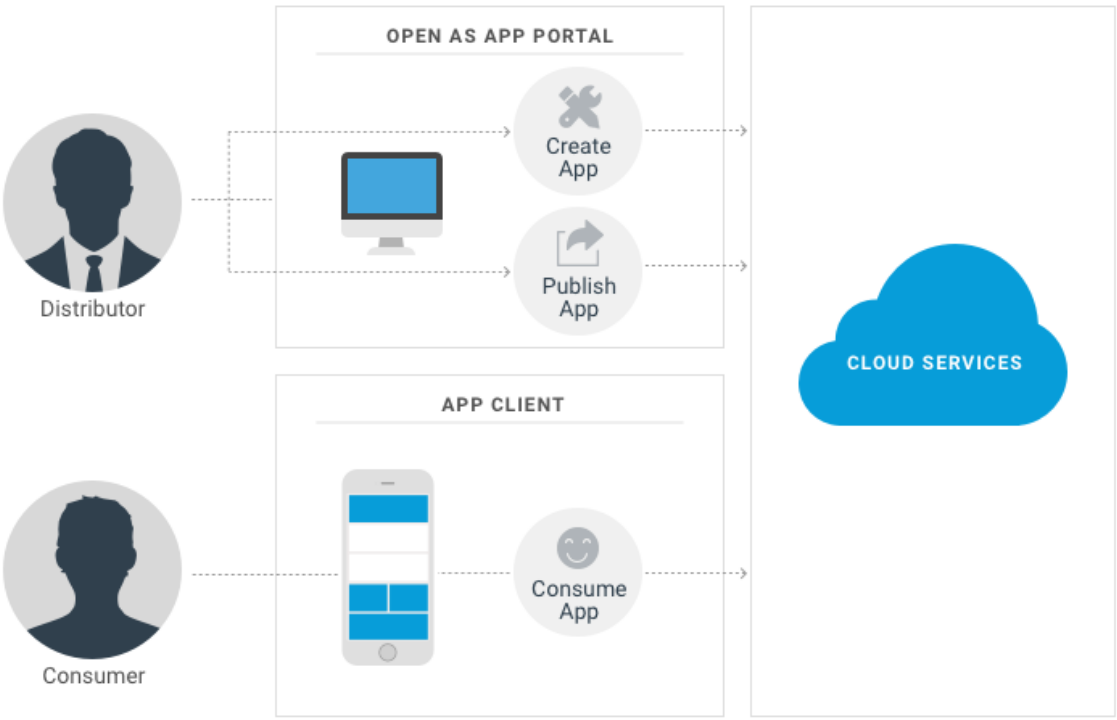

Figure 2: Responsibilities of the Open as App Portal, the App Client, and the Cloud services

Figure 3 provides an in-depth illustration of the workflow of creating and consuming a spreadsheet-based app.

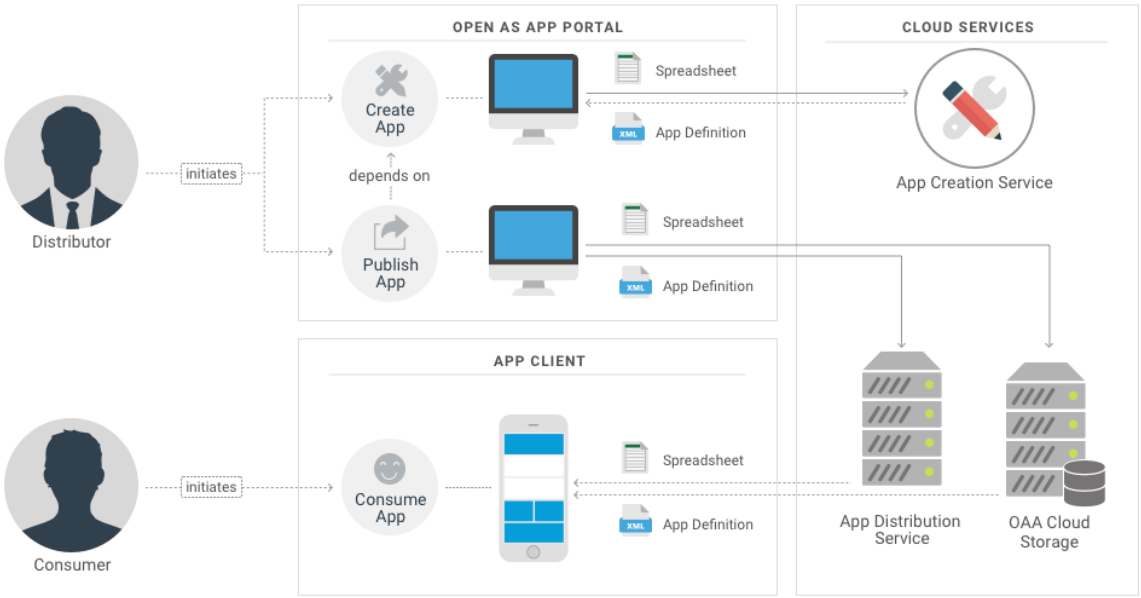

Figure 3: Workflow for creating and consuming apps based on spreadsheets from a user's local device.

For creating an app, the user sends a spreadsheet to the *App Creation Service* where it is processed by Open as App's App Creation Technology (patent-pending). The user negotiates the desired app layout with the *app creation service* via *the Open as App Portal* in a semi-automatic guided process.

VPN with auto-expiring memory

It is important to note that the *app creation service* is an isolated service in the Open as App cloud which securely processes the data in-memory. Intermediate results from this service are temporarily and anonymously stored in a secured non-persisting behind-VPN in-memory Redis[3] cache, auto-expiring after a short period of user inactivity and only visible to the active user (secured by a random identifier token only known to the user's active browser session).

No Open as App service and especially no third party is able to access the files provided to the app creation service without an explicit request from the user itself. Furthermore, the files are only used for the app creation process. At no time, data from these files is stored in any persisting location, neither directly nor indirectly. Any processing is solely done in-memory. Neither the file itself nor any derived data is ever stored in the cloud. Except, of course, one decides to host the spreadsheet with Open as App in the publishing process described in the following paragraph rather than with third-party cloud storage providers or on-premises.

Still, Open as App offers the possibility to design apps by hand and to skip all interaction with the *app creation service* for an application in maximum security environments. However, this means passing on the patent-pending semi-automatic app creation system of Open as App.

After the user has decided on the final app layout, he or she can finally upload and publish their app. The spreadsheet is uploaded to the Open as App *cloud storage* (technically, the spreadsheet is retrieved from the *app creation service*'s in-memory cache with the file's random identifier token only known to the user's active browser session rather than being uploaded a second time). The *app definition* including the URL of the spreadsheet is published to the *app distribution service*. When creating an app with a file from the user's local device, the spreadsheet is always stored in the Open as App cloud, however, this step is optional. The spreadsheet may also be hosted with external cloud storage providers (as Dropbox, OneDrive, or Google Drive) or even with on-premises servers. See the following sections *App Creation Process with External Cloud Storage Providers* and *App Creation Process with On-Premises Data* for more on this topic.

After the app has been published, it can be consumed by authorized Open as App users (see *Team, User, Groups, and Guests* for more on the sharing capabilities of Open as App). For this purpose, the Open as App Client retrieves the *app definition* from the *app distribution service* and a copy of the spreadsheet from the *cloud storage*.

> ### Access Control
>
> Both, the *app definition* as well as the spreadsheet, are secured by the Open as App authentication and resource sharing system and only accessible by people explicitly authorized by the apps' administrators (see following section *Authentication Concepts* for more on this).

App definition and spreadsheet are afterward combined for rendering the app on-demand on the mobile device as already described in the previous section *App Rendering*. This separates layout, data, and logic and enables the system to retrieve the spreadsheets from arbitrary data sources like external cloud storage providers (as Dropbox, OneDrive, or Google Drive) or even on-premises servers, allowing Open as App to be used without having to store sensitive data in the cloud.

This topic is elaborated in the following sections *App Creation Process with External Cloud Storage Providers* and *App Creation Process with On-Premises Data*.

App Creation Process with external Cloud Storage Providers

As discussed in the previous section, users are offered the possibility to host spreadsheets for their apps with Open as App's own cloud storage. However, Open as App also integrates with a variety of third-party cloud storage providers (including Dropbox, OneDrive, and Google Drive), allowing users to create apps based on files from their personal cloud storage from supported external providers.

Figure 4 illustrates the workflow of creating and consuming apps with spreadsheets from external cloud storage providers. Due to the modular structure of the Open as App system, this workflow is very similar to the workflow discussed in the previous section. Basically, it only differs in the way the *Open as App Portal* and the *Open as App Client* access the spreadsheet.

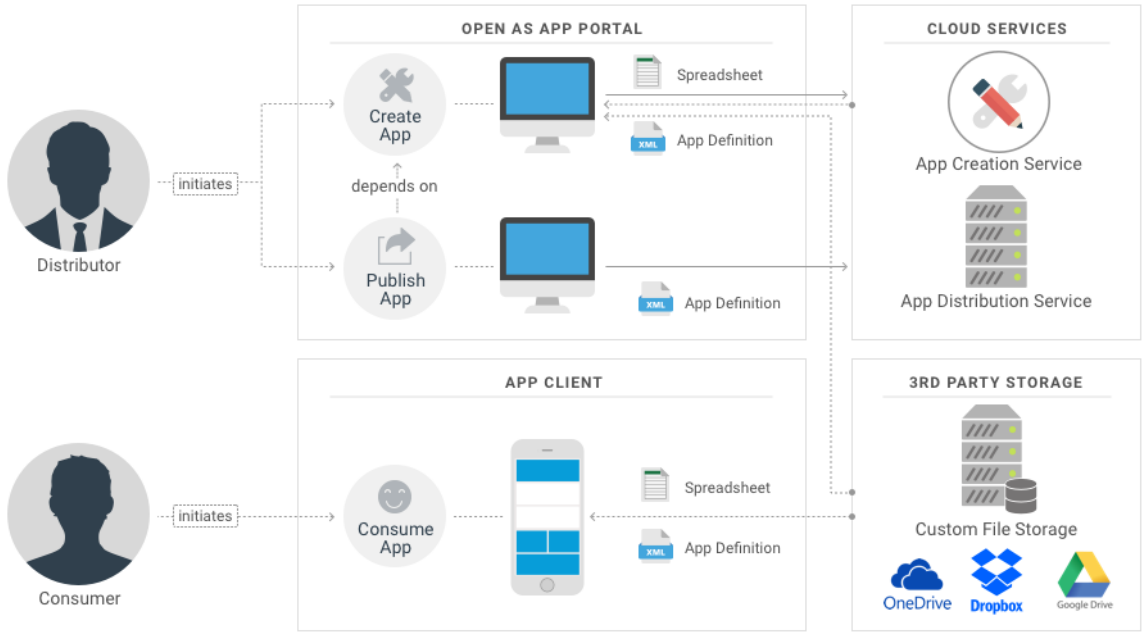

Figure 4: Workflow for creating and consuming apps based on spreadsheets from third-party storage providers

The app creation process starts with the *Open as App Portal* retrieving the spreadsheet to create the app from. The user selects the desired cloud storage provider, authenticates to the platform (usually using OAuth[4]), and selects the file to use. Afterward, the management portal downloads the file into local memory and the app creation process proceeds as if the file was from the user's local device (especially the interaction with the *app creation service* stays the same) until the user is about to publish the app. When publishing the app, the spreadsheet is

not uploaded to the Open as App *cloud storage*. Instead, the *app definition* is directly linked to the spreadsheet's original location since the file is already available online.

The process of consuming an app essentially stays the same as if the app was hosted with the Open as App *cloud storage*. It only differs in the way the *Open as App Client* retrieves the spreadsheet. It starts with the *Open as App Client* retrieving the app definition from the *app distribution service* and downloading the spreadsheet. Instead of retrieving the spreadsheet from the Open as App *cloud storage*, however, the *Open as App Client* directly retrieves it from the external cloud storage provider. It locates the file, authenticates the user with the external platform if required (usually using OAuth5) and downloads it into its encrypted local storage. Afterward, both resources are combined for rendering the app on-demand as usual.

App Creation Process with On-Premises Data

Just like Open as App can be used with external cloud storage providers (as discussed in the previous section App Creation Process with External Cloud Storage Providers), Open as App can also be used with on-premises servers (like SharePoint on-premises installations). Besides the location of the server, this approach does not differ from the one described in the previous section. Please refer to the previous section *App Creation Process with External Cloud Storage Providers* for details on this approach.

> ## Maximum Data Security Approach
>
> Spreadsheets can be hosted on-premise. Solely the *app definition* - not containing any sensible data - is still hosted in the cloud with the *app distribution service*. *The Open as App Portal* as well as *the Open as App Client* directly retrieve the spreadsheet from their original location and do not require any interaction with a cloud service for doing so. Therefore, spreadsheets may be hosted on-premises just like they can be hosted with external cloud storage providers.

Since both systems run on the user's local device (the *Open as App Client* is a native mobile app and the *Open as App Portal* is a pure JavaScript application running in the user's local browser), it is even possible to run apps based on spreadsheets securely stored with on-premises servers behind private VPNs – the server

is not required to be accessible via public IP. Open as App provides maximum data security with this approach - app distributors have most fine-grained control of their apps since they have full control of their data. It is technically impossible to run apps if their spreadsheets cannot be accessed. Therefore, withdrawing access to a spreadsheet automatically withdraws access to their apps.

Open as App integrates with any on-premises servers offering files via the HTTP protocol. On-premises files can be secured using basic authentication, NTLM or OAuth.

Authentication Concepts

- User register with Open as App by providing their email address and a password of their choice. Passwords are combined with a salt of at least 64-bit and stored as hashes derived with PBKDF2[5] using at least 10,000 iterations as recommended by NIST[6].
- Users are required to confirm the possession of the email address they registered with before being able to use all features of Open as App. Users can start testing and creating apps without confirmation but are required to confirm their email address before being able to accept any invitations to prevent illegitimate access to shared apps.
- Authentication is based on short-lived HMAC-SHA256 signed JWT[7] tokens. Mobile devices use revocable refresh tokens to get permanent access.
- Companies can bring their own identity management. Open as App supports integration of a company's Active Directory and the like using OpenID Connect and SAML2 protocols.
- Email confirmation links are secured with non-expiring HMAC-SHA256 signed JWT tokens.
- Invitation links are secured with non-expiring HMAC-SHA256 signed JWT tokens.
- Password reset links are secured with short-lived (expiration after 24h) HMAC-SHA256 signed JWT tokens.

Client-side security

In order to support offline usage, we save your Excel file locally on the mobile device. The following section describes our solution to ensure best security for your data inside the native mobile app.

On first app start we create a 256 Bit Key using the cryptographic .NET RandomNumberGenerator[8]. This key is stored in the iOS Keychain[9] and Android Keystore[10].

The key is used to encrypt and decrypt an SQLite database with 256-bit AES encryption using SQLCipher[11], which uses peer reviewed crypto OpenSSL algorithms. Your Excel file is never persisted outside of the encrypted database.

On top of that you can integrate our app with MDM solutions like MobileIron or BlackBerry Dynamics Mobile Security Platform to keep more control over devices that are using Open as App. In general, we support MDM solutions that are AppConfig.org compliant[12].

Open as App also provides a remote wipe feature that you can trigger in case a device is stolen or lost. The server sends a push notification to the client and the next time the mobile device gets a connection to the server it synchronizes and removes apps were access rights have been revoked.

Another possibility to further enhance the client-side security is to encrypt your Excel file with a password[13]. In this case the user must insert the password when opening the app inside of the Open as App container. The file is never stored without the password lock and as described above it is also saved within our encrypted database.

Regarding network communication on iOS we use NSURLSession for server communication to ensure maximum security and compliance with MDM software.

Further reading and sources

[1] Microsoft Data Protection API: https://msdn.microsoft.com/en-us/library/ms995355.aspx

[2] MobileIron: https://www.mobileiron.com, BlackBerry Dynamics: https://en.blackberry.com/enterprise/blackberry-dynamics

[3] Redis: http://redis.io

[4] OAuth: http://oauth.net

[5] PBKDF2: http://www.ietf.org/rfc/rfc2898.txt

[6] NIST Recommendation for Password-Based Key Derivation: http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-132.pdf

[7] JWT: https://jwt.io

[8] RandomNumberGenerator: https://msdn.microsoft.com/de-de/library/system.security.cryptography.randomnumbergenerator(v=vs.110).aspx

[9] Keychain: https://developer.apple.com/documentation/security/keychain_services

[10] Keystore: https://developer.android.com/training/articles/keystore

[11] SQLCipher: https://www.zetetic.net/sqlcipher/

[12] AppConfig.org: https://www.appconfig.org/

[13] Excel Encryption: https://docs.microsoft.com/en-us/DeployOffice/security/cryptography-and-encryption-in-office

Open as App

Open as App is the first #nocode platform empowering everyone to create and share apps automatically, including logic, charts, and calculations.

We enable individuals and companies of all sizes to

- save time and budget
- protect their know-how
- secure and manage the way, data are shared
- provide great customer experience and innovation
- increase their resources for digitization with citizen developers

And the best, Open as App is free to try.

Would you like to use Open as App as an Enterprise platform?

Go to www.openasapp.com/enterprise and book your demo today!

Open as App GmbH

Create & share great apps based on your data in Excel, Google Sheets or databases. Automatically. Instantly. #nocode. Secure. Any platform.

www.openasapp.com

contact@openasapp.com