IDENTITY AND DATA SECURITY IN K2 APPIT FOR SHAREPOINT

Originally published: July 6th, 2016
Updated February 11th, 2016 with K2 Mobile for Android information
Updated February 10th, 2017 with information on token changes, handling, and storage

Table of Contents

INTRODUCTION

K2 Appit® for SharePoint applications act as a hub of information in the enterprise. In this role Appit handles data from many different systems which is exposed to users through Forms, Reports and consumed in Workflows. The way Appit authenticates users and communicates with line-of-business (LOB) systems is secure and reliable. This document describes some technical aspects of identity and data flow in the context of K2 Appit for SharePoint applications.

This document describes how Appit:

- Protects your data, allowing only your users access to your company's information
- Integrates securely with Azure AD for authenticating users
- Uses standard protocols, such as OAuth, for communicating with LOB systems
- Makes use of the Office 365 framework for requesting administrative consent
- Employs secure technologies for data encryption and does not store or cache business data

UNDERSTANDING SMARTOBJECTS

It is important to understand the central role that SmartObjects play in K2 Appit for SharePoint and the K2 platform. In K2-based applications, SmartObjects are the primary data access mechanism that allows K2 to interact with LOB systems.

A single SmartObject can contain data from multiple LOB systems, each with its own authentication and/or authorization mechanism. In this case K2 provides an abstraction layer with SmartObjects that allows the K2 user to provide a logical and complete data entity which is more aligned to how users think about their data than the actual LOB storage and authentication and authorization mechanism. This allows users and workflow designers to more easily use data in their application elements. Diagram 1 illustrates the foundational role that SmartObjects play.

One of those application elements, SmartForms, is built entirely on SmartObjects. This link between SmartForms and SmartObjects emphasizes the need to understand authentication modes and authorization options for SmartObject service instances.

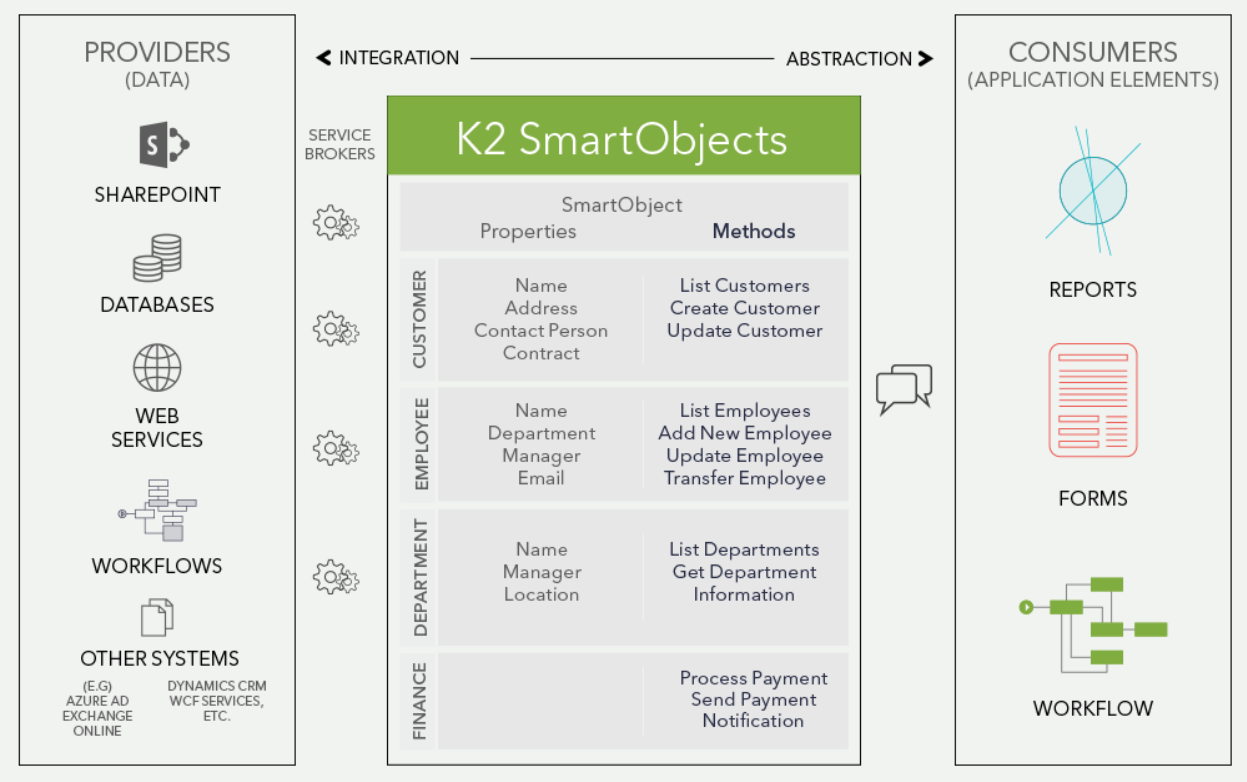


Diagram 1: SmartObject LOB Data Abstraction – SmartObjects allow a K2 administrator to abstract LOB data into logical entities which are then used in application elements such as forms, workflows and reports.

IDENTITY

Most communication from K2 to LOB (line-of-business) systems happens via the SmartObject Service layer, which is the common point of integration and custom extension in the K2 platform. This layer supports many different types of authentication modes, including OAuth, Single Sign-On (SSO), Static and Integrated. Depending on the specific service and the LOB system, the SmartObject Service layer can use whichever mode is best suited for communicating with the LOB system in the most secure manner.

In Diagram 2 a SmartForms user is authenticated to the K2 server, either in the browser or from a mobile device. In the case of the browser the user is authenticated via Azure AD. The user's claims-based identity is then used by the K2 server to communicate with the LOB system. This authorization mechanism is determined by how the SmartObject service instance is configured to communicate with the LOB system. In the case of SharePoint it uses OAuth but it can be any number of mechanisms depending on what the LOB supports and how the service instance is configured.

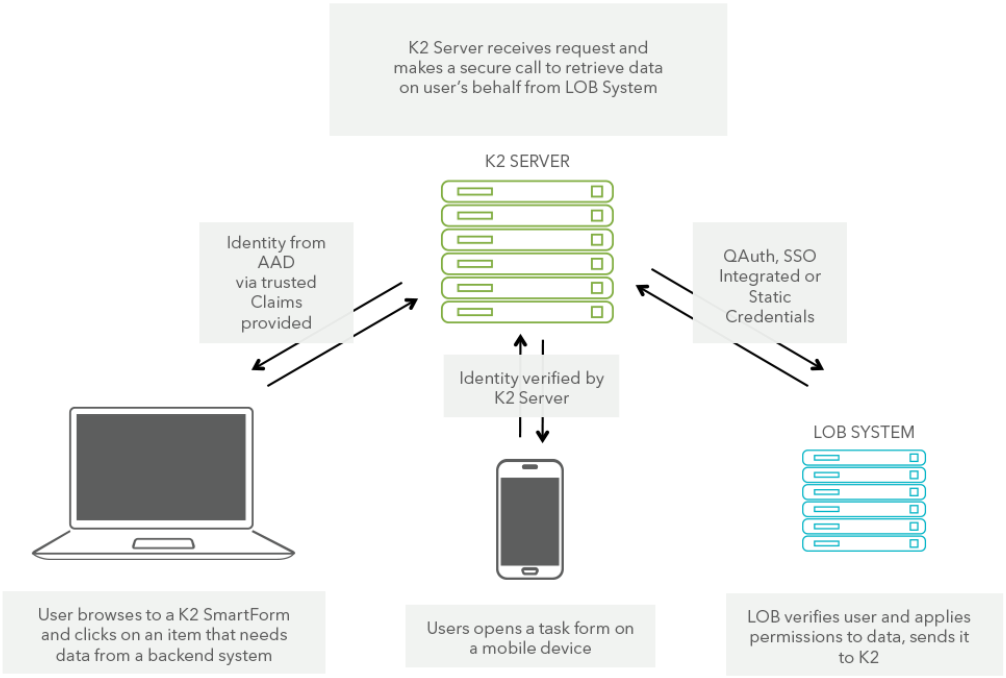


Diagram 2: Authentication and Authorization – The authentication when the user opens a SmartForm from any device is determined by how the SmartForms site is configured, which is typically via Claims. The K2 Server verifies the K2 Mobile user first and then follows the configured authentication path for that user's identity. The authorization to the LOB system is determined by the SmartObject service instance.

Another way of looking at this, from the K2 Server to the LOB system, is that the SmartObject services layer controls how the communication happens to the LOB system. In Diagram 2 you can see that the service instances can be configured in a number of different ways, depending on what is most appropriate for the LOB system. SharePoint 2013 and SharePoint Online use OAuth for authorization, which is the most secure and allows Appit to impersonate the user without needing to store their credentials.

When a SmartForm is opened in the browser or in the K2 Mobile app, the SmartForms site authenticates the user with the K2 Server. The K2 Server then retrieves the form definition from the K2 database, populates the form using AJAX (Asynchronous Javascript and XML) calls to the SmartObject API which in turn calls whatever LOB system is required based on the service instance configuration. In this way, setting up your SmartObjects with the proper authentication mechanism automatically ensures that your forms are retrieving and writing data from and to the LOB system as required by your scenarios.

Looking at this is from the user's identity, when they open a SmartForm in SharePoint, Appit has already been configured to trust the Azure AD (AAD) provider. SharePoint hands the user's identity over to Appit. Once verified directly with AAD, that user's identity can then flow all the way through to the LOB system when the service instance is configured for OAuth, SSO or

Windows. The user is authenticated via an OAuth token or a cached credential via SSO. When static credentials are used, a single identity is used for all communication with the LOB system and the user's identity cannot flow through to the LOB system.

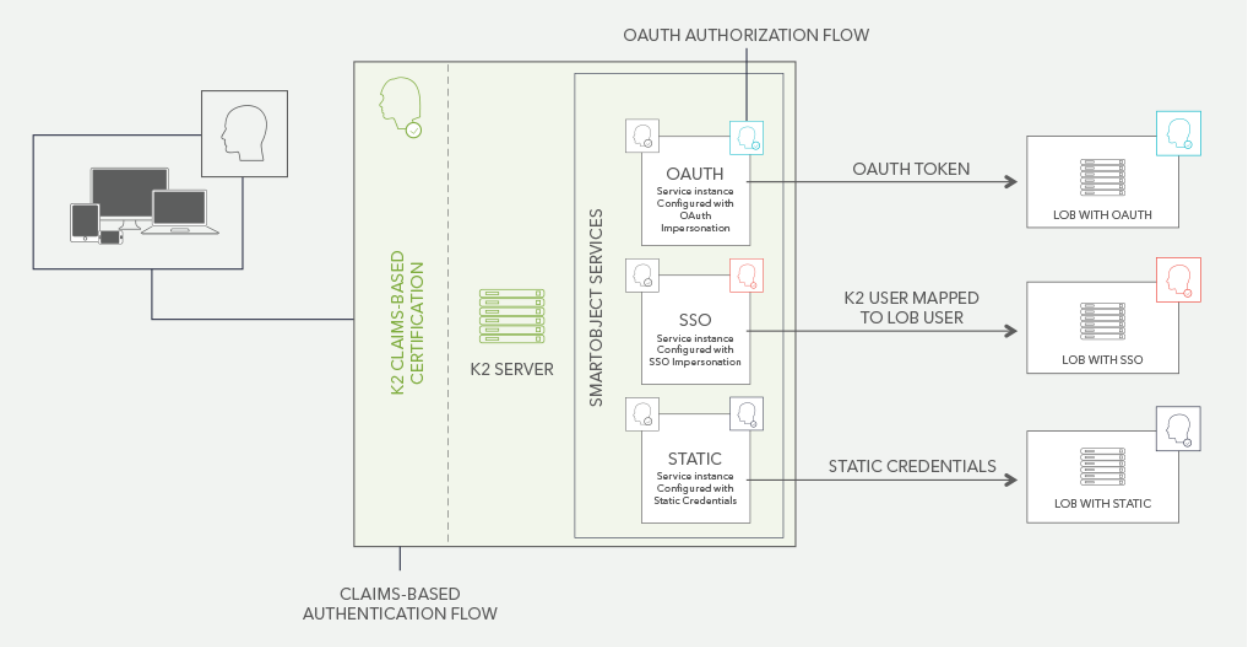


Diagram 3: User Identity Flow – The user opens a SmartForm in SharePoint. SharePoint passes the claims token on to K2. K2 trusts AAD and cracks the claim to resolve the user. The K2 server then, according to how the service instance is configured, either impersonates the user or uses stored SSO credentials to connect to the LOB system. Or, in the case of static credentials, uses a single account and the user information cannot flow through to the LOB system.

When the user comes into Appit they are authenticated via claims. The following diagram illustrates the claims-based authentication flow.

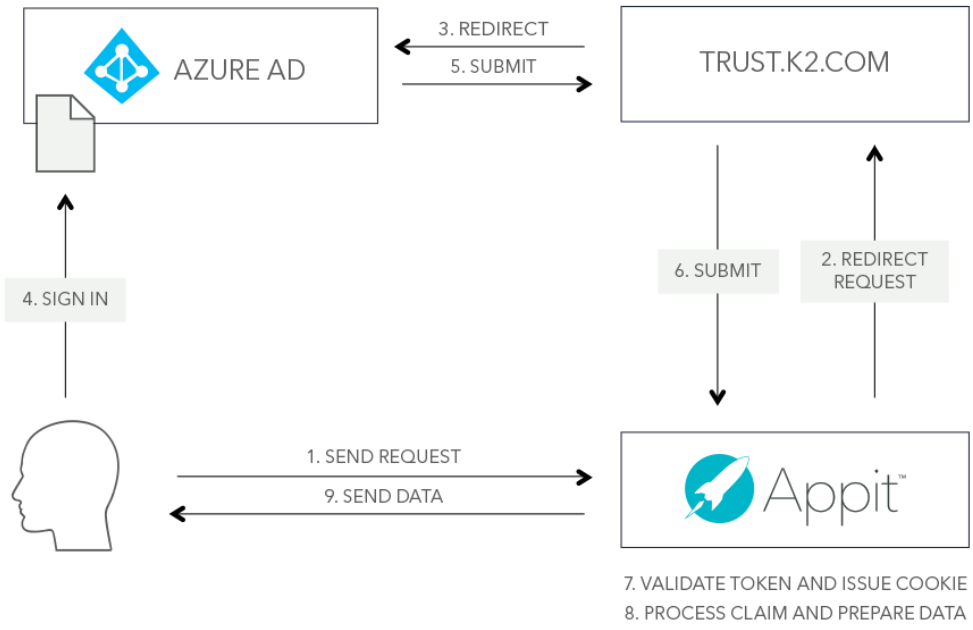


Diagram 4: Claims-based Authentication Flow – The claims-authenticated user browses to a SmartForm in SharePoint or opens a SmartForm in the K2 Mobile app. The request is sent to K2 Appit which redirects the request to trust.k2.com, which forwards the request to AAD. AAD (Office 365) prompts the user to sign in (only when they do not have an active session) and a token is generated by the issuer, aka the STS, which is submitted to trust.k2.com and forwarded to K2 Appit. K2 Appit validates the token, issues a cookie, processes the claim and finally prepares and sends the data to the browser or mobile client.

Claims-based Authentication is all about determining who the user is, and is an incoming *authentication* flow to Appit. Once the server knows who you are, sending that information to other systems so that they can *authorize* Appit to do something on those systems on your behalf is where OAuth comes into play. The OAuth flow happens in a similar way via trust.k2.com, Azure AD and K2 Appit for SharePoint.

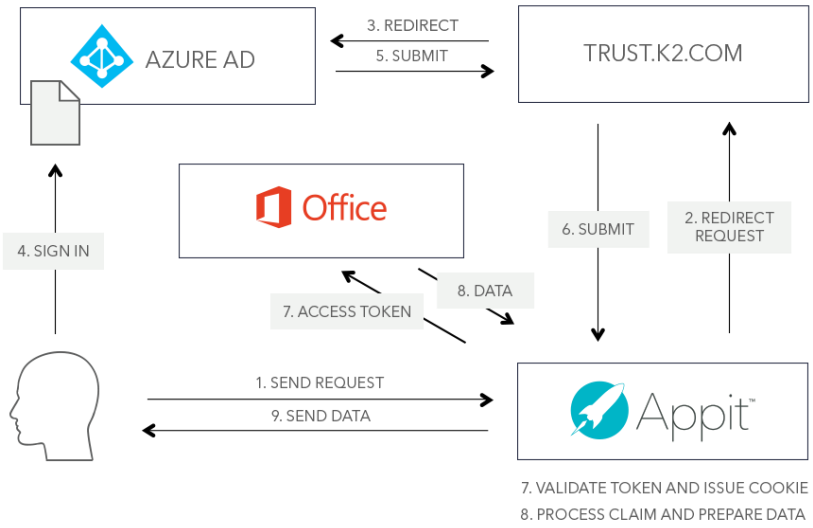


Diagram 5: OAuth Authorization Flow – The user's identity is redirected from K2 Appit for SharePoint through trust.k2.com and then to Azure AD. The OAuth flow to SharePoint/Office 365 follows the path above to get an authorization code and the resulting access token is used to query and update data with SharePoint/Office 365. With Common Consent the user is not prompted to trust the K2 for SharePoint app so they may not see the prompt in #4. However, if they've not logged in to SharePoint/Office 365 in 24 hours they'll need to re-authenticate and will see a login prompt.

ABOUT TRUST.K2.COM

In Diagram 4 you'll notice that trust.k2.com is used as an intermediary between the K2 server and AAD's STS (ACS). This is because Appit must maintain an internet-accessible Single Sign-On (SSO) service to handle claims from multiple K2 Appit for SharePoint tenancies based on unique Office 365 realms. It is a Relying Party Security Token Service (RP-STS) used to broker authentication requests between Appit and the AAD STS. In this capacity, trust.k2.com serves as a generic authentication and authorization redirector for the entire K2 platform. Validation on trust.k2.com between an Appit instance and an Office 365 realm is linked at the time of provisioning, and all future communication between a specific Appit tenancy and a unique Office 365 is validated based on the claims identity that is received from AAD.

A requirement of all requests to and from Azure is that it occurs over an encrypted, SSL connection. This encryption also occurs from the Appit server to trust.k2.com and can only be decrypted by trust.k2.com. A different certificate on trust.k2.com is used to sign claims requests received from AAD which are then validated by the Appit server based on the thumbprint of trust.k2.com. In an OAuth flow, the client ID and reply URL are used to map a specific user's token to a registered AAD tenant ID which is established at the time of provisioning. A token that does not contain a valid realm ID for the given AAD tenant is blocked and cannot continue. If the realm ID matches based on what has been registered in

trust.k2.com, the authorization code is passed to the Appit server which then retrieves the access token. With this access token the Appit server can then act on the user's behalf via the service instance which is configured for OAuth.

When the thumbprint of trust.k2.com is changed due to a certificate renewal, the thumbprint is automatically updated in the claims mapping of the Appit tenancy through metadata exchange between the Appit server and trust.k2.com.

SMARTOBJECT SERVICES AUTHENTICATION AND TRANSPORT MODES

K2 Appit for SharePoint provides several standard Service Types which are used to expose certain LOB technologies as SmartObjects. These Service Types support different authentication and transport modes. It is important to use the appropriate authentication mode for the system you wish to integrate with, to maintain data security.

The table below describes the currently available Service Types in K2 Appit for SharePoint along with the available authentication modes and the default transport mode used by these services.

Table 1: Transport Protocol Matrix

Service Name	Description	Authentication Mode	Transport Mode
SharePoint	Provides K2 access to SharePoint site collections that have been integrated using the K2 Appit for SharePoint app	OAuth	REST (SSL)
SQL	Provides read/write access from K2 to SQL tables, views and stored procedures.	Static, Impersonation, Service Account[1]	TDS (over SSL)
Exchange	Enables K2 to allow several Exchange administrator tasks from SmartObjects and SmartWizards.	Static, Service Account, SSO	SSL
Dynamics CRM	Provides access to CRM data and is used by K2 CRM SmartWizards for workflow integration with CRM.	Static, Service Account, SSO	SSL
Web Services	Provides access to data provided by web services to K2.	Static, Service Account, SSO or OAuth[1]	SSL or unencrypted[2]
SmartBox (SQL Azure)	Provides access to store and retrieve data in SmartBox (stored in the SQL Azure instance associated with K2).	Service Account	TDS (over SSL)
Azure Active Directory	Provides access to Azure Active Directory (AAD) for user information.	OAuth	REST (SSL)
DocuSign	Provides access to DocuSign for signing documents via K2 SmartWizards.	Static	REST (SSL)

[1] The authentication mode is determined when you create the service instance.

[2] The transport mode depends on the service and your choice of configuration.

ENCRYPTED TRANSPORT

All data transport between the browser and the K2 SmartForms server utilizes, by default, Secure Sockets Layer (SSL). The resulting internal data flow between the SmartForms server and the internally-hosted SmartObject and Workflow servers are communicated through local system internal transport channels. This internal server-to-server subsystem transport channel is not encrypted as it is entirely isolated via internal local server connections.

When the K2 platform is integrated with external LOB systems through the SmartObject broker layer, all data flow is, by default, configured to utilize SSL (for the REST-based brokers) or TDS protocol over TLS (for the SQL Azure broker).

When using Microsoft Azure Active Directory (AAD) for authentication, AAD requires that all communication, including integration communication such as between Appit and AAD, happens over a secure channel.

Please refer to **Table 1: Transport Protocol Matrix** for details on transport security and encryption.

IDENTITY USED IN WORKFLOW SERVER STEPS

When the user context is not available, such as during the execution of workflow server events, the identity of the Appit server is used to communicate with other systems. This contrasts the flow of identities and LOB system impersonation when a user is interacting with a LOB system through a SmartObject, because in a workflow server step, there is no context of a currently-connected user. With Workflow Server steps, the Appit server is the only identity that is in context when a Workflow Server event is executing, and this identity is used to connect to the LOB system.

For example, when a SharePoint document is moved or copied during workflow execution, the identity associated with the destination document is the SharePoint identity associated with the Appit server. This means, in the case of SharePoint 2013 and SharePoint Online, that the Appit server must have sufficient rights to create and update documents and document metadata on the target library.

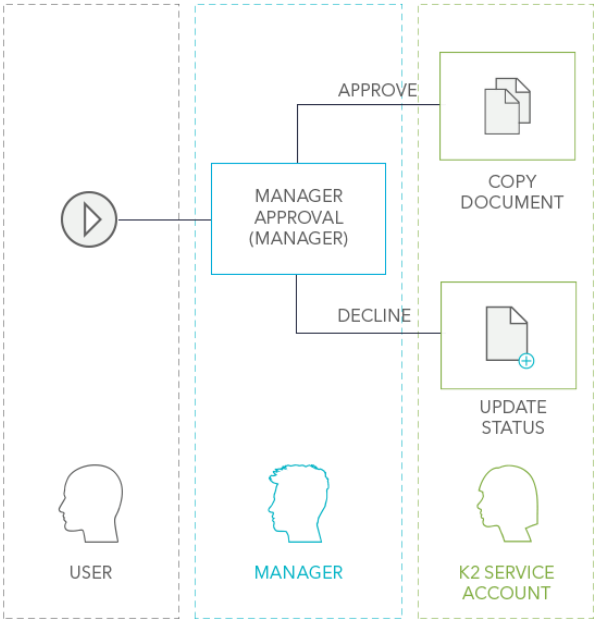


Diagram 6: Workflow Engine Execution – When the user starts a workflow, the user is authenticated with K2 and their permission to start a workflow is verified and recorded. In the Manager Approval step, the user's manager is authenticated and assigned rights to action the task. The action is recorded in their name. Either action, Approve or Decline, then happens as the Appit Server since these are server events. On Approval the document is copied to another library. On Decline the document's status is updated. Both actions occur using the identity of the Appit Server because they are server events and there is no user context. In the case of SharePoint, the Appit server's OAuth token is used by SharePoint to authorize Appit.

DATA AND CREDENTIAL CACHING

The K2 platform architecture does not synchronize or cache business data from LOB systems and is instead designed to securely retrieve or update data in real time from external LOB systems.

Although data flows directly from the external LOB system through the K2 platform to the end-user's browser and is never permanently stored in Appit, the K2 platform does make use of SQL Server Common Table Expressions (CTEs) for internal operations such as SmartObject disparate data joins and normalization. A CTE is similar to a derived table in that it is not stored as an object and lasts only for the duration of the query

Furthermore, the K2 platform does not attempt to replicate access control over LOB system data. The data access control is enforced by the LOB system through flowing the identity of the user making the request, from the browser through to the LOB system and allowing the LOB system to enforce in place access control and trim data accordingly.

Please refer to the Identity flow and caching matrix for details on identity types supported and flowed through by the K2 platform.

INTERNAL STORAGE OF BUSINESS APPLICATION DATA

The primary application data access mechanism in Appit is SmartObjects, which means that business data is never stored by Appit but rather retrieved by Appit on-demand. In most cases, the business application data would reside in a particular LOB system such as SharePoint, SQL Server, Azure Active Directory or some other data storage location. However, there are some situations where K2 may store auxiliary business data internally.

Note that communication and data transfer between the K2 application service and the K2 SQL Azure database is encrypted with the Encrypt parameter provided by *SQLConnection* objects.

SMARTBOX-BASED SMARTOBJECTS

K2 provides a mechanism for exposing business data as SmartObjects when there is no existing system of record that provides the data. This storage area is a collection of K2-managed SQL Database tables in the K2 application database, known as *SmartBox*. When building SmartObjects, the application designer may choose to create SmartBox-based SmartObjects, which means that any business data stored in these SmartObjects will be stored in the K2 database. By default, Appit does not encrypt this stored data and the data stored in SmartBox tables is accessed and exposed just like any other SmartObject.

Once a SmartBox SmartObject is created, it is possible to use K2 administration tools to set method-level security on that SmartObject to specify which users may execute which methods on that SmartObject, for example controlling which users may execute a Delete method vs. which users may execute the Get List method for a specific SmartBox SmartObject. By default, the authentication mode of the SmartBox Service Instance is configured to use the K2 Service Account, but user credentials are taken into account to maintain the SmartObject method-level security settings, so effectively SmartBox uses Impersonation to authorize access to SmartBox data.

If you do not wish to store business data using K2 SmartBox, you can use another data storage mechanism such as SQL tables in a different database or a SharePoint List to store your business data instead, expose the data through the standard K2 SmartObject mechanism and use the LOB system's security features to restrict access to the data.

PDF SnapShots

When using the Save as PDF function to save a snapshot of a form, the generated PDF files are saved to a SmartBox-based SmartObject called "PDF File". This could be considered as business data storage, since the generated PDF snapshot could contain business data that was displayed on the targeted form at the point that the PDF was generated. These PDF files are stored

indefinitely by default, but it is possible to delete the stored PDF files from this database by executing the delete method of the PDF File SmartObject.

WORKFLOW DATA

Appit stores native workflow reporting data internally in the K2 databases. This data includes workflow statistical data such as timestamps and workflow auditing data such as audit trails. Normally, this workflow reporting data does not include business application data, unless gusiness data is captured in something like the Workflow Folio property during workflow execution.

Normally, business application data is abstracted from the workflow and the workflow only stores record ID's that are used to retrieve data on-demand from another system. It is possible for workflow designers to define workflow-level Data Fields and XML Fields to store business application data on the workflow level. In these cases, Appit will store this business data as part of the workflow data in the K2 database.

The workflow data stored by Appit can be protected by setting workflow-level permissions through Appit administration tools. These tools allow administrators to restrict the users that may report on workflows (and therefore view business data stored in the workflow).

Note: If business data was included in something like an email that was sent as part of a workflow, the data in the email should be considered uncontrolled since the email may be forwarded by the recipient, outside of the control of Appit or any other system.

DATA LOSS AND PREVENTION

K2 Appit for SharePoint is a highly-available service that is built on SQL Azure. In terms of data loss and prevention (DLP), the SQL data is auto-replicated to a different geo-location. This depends on the features available to your specific Azure subscription. If auto-replication is not available, Appit backs up all data on a periodic basis.

Appit also stores all K2-specific configuration settings so that environments can be rebuilt in the event of a disaster.

THE K2 APPIT FOR SHAREPOINT AND K2 FOR OFFICE 365 APPS

There are two K2 apps that allow Appit to integrate with Microsoft SharePoint and Office 365 technologies: *K2 for Office 365* and *K2 Appit for SharePoint*. Use the information in this section to learn more about these Apps and the permissions required to install and use them.

This topic is divided into two sections: SharePoint Online and SharePoint 2013 On-Prem. This is because the app requirements are different depending on which version of SharePoint you are integrating with. SharePoint Online requires the *K2 for Office 365* and *K2 Appit for SharePoint* apps, while SharePoint 2013 on-prem requires only the *K2 Appit for SharePoint* app.

K2 APPIT FOR SHAREPOINT ONLINE

Because SharePoint Online is part of the Office 365 offering, K2 Appit for SharePoint Online requires two K2 apps: *K2 for Office 365* and *K2 Appit for SharePoint*. This section details the requirements for both apps.

To upload the K2 Appit for SharePoint app to the SharePoint App Catalog you must have the following permission:

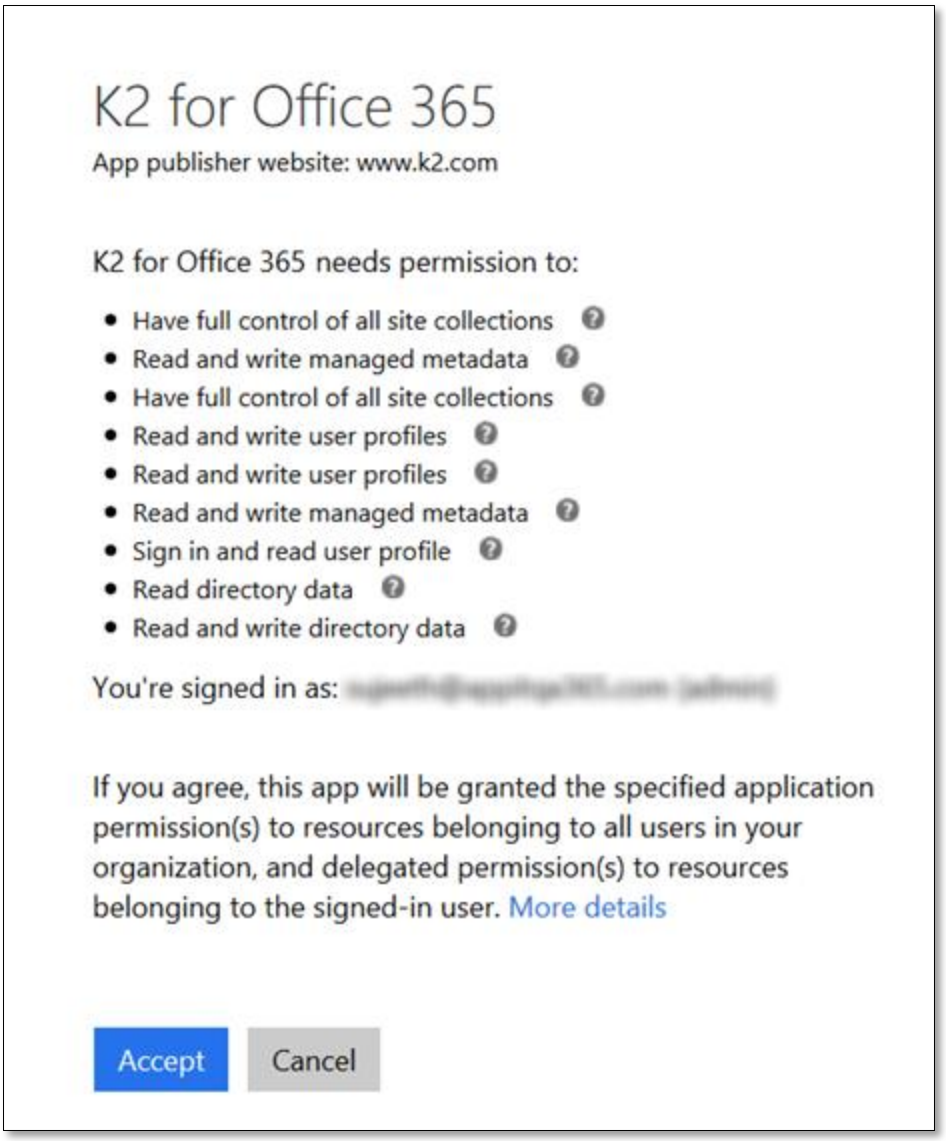
- **SharePoint Online Tenant Administrator.** This is because SharePoint requires this permission for provider-hosted apps such as K2 for SharePoint. For more information see Authorization and authentication of SharePoint Add-ins (MSDN)

Application Scope and Permission Requests

The K2 Appit for SharePoint online solution requires the installation and configuration of the **K2 for Office 365** and **K2 Appit for SharePoint**.

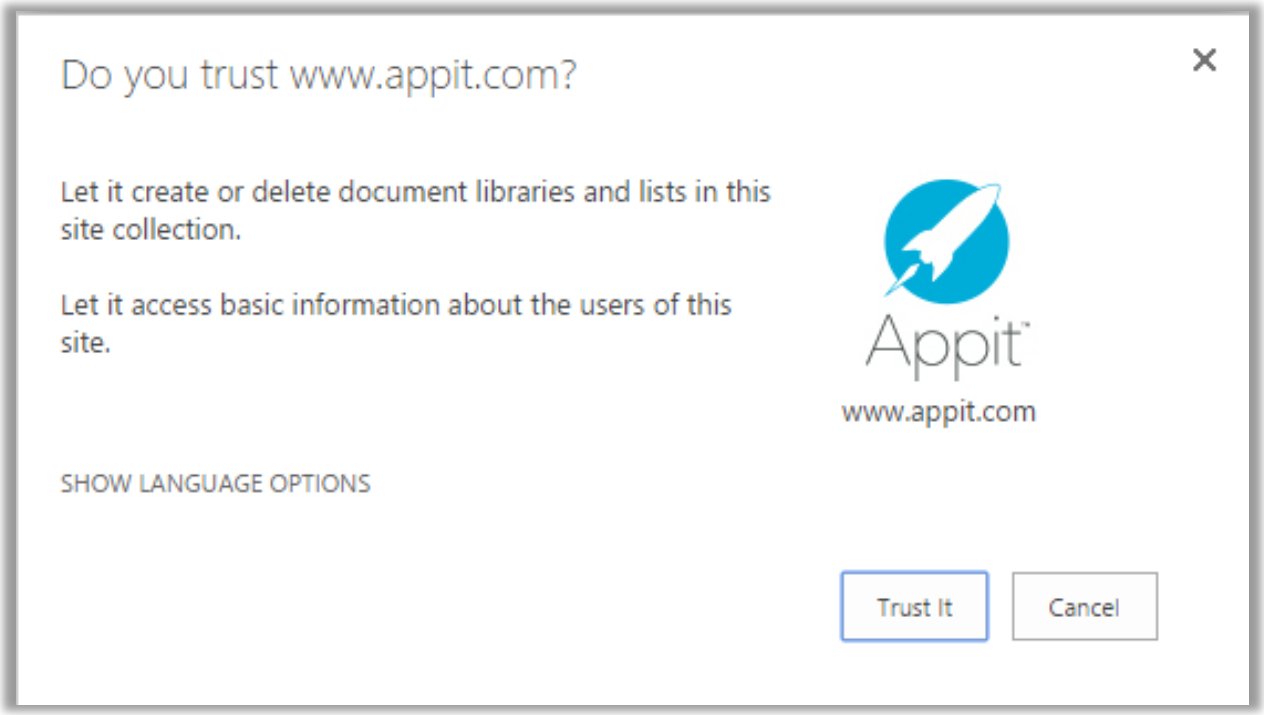
The **K2 for Office 365** application requests the following permission scopes

- **Have full control of all site collections**: Allow the application to have full control of all site collections on behalf of the signed-in user
- **Read and write managed metadata**: Allows the app to read, create, update, and delete managed metadata and to read basic site info on behalf of the signed-in user.
- **Read and write user profiles**: Allows the app to read and update user profile information.
- **Read and write directory data**: Allow the application to read and write data in your organization's directory, such as users and groups.
- **Sign in and read users' profiles**: Allow users to sign in to the application with their organizational accounts and let the application read the profiles of signed-in users, such as their email address and contact information.
- **Read directory data & Read and write directory data**: Allows the app to read and write information to Azure Active Directory. Unless you use the Azure AD wizards in workflows, the write portion of this permission is not used by K2.



The **K2 Appit for SharePoint** app requests the following permission scopes to register remote event receivers on lists and libraries.

- **Let it access basic information about the users of this site.** This is the base scope required for all applications.
- **Let it create or delete document libraries and lists in this site collection**. This maps to Scope="http://sharepoint/content/sitecollection" and Right="Manage"



Installation, Configuration and Use of K2 Appit for SharePoint

The **K2 for Office 365** application requires the following permissions to install and configure the application on each **tenancy**.

- The minimum permissions required to grant permissions to the application
  - Tenant: **Global Administrator**
  - Beginning with Appit 1.5 Update 3 (Feb 2017), an AppOnly OAuth token is stored and used by the K2 Service when user context is not available, such as execution of server-side events in workflows. For more information see the Outbound Authorization and OAuth in K2 whitepaper in the whitepaper series available at http://help.k2.com/news/authwhitepapers
  - The OAuth AppOnly token is valid indefinitely and is used only by the K2 service account to get access tokens on behalf of users. This token request happens using a connection to trust.k2.com to protect the K2 application client secret. While trust.k2.com brokers the token exchange, it does not store the access or refresh tokens, which are only stored by the K2 server.
  - User context is available for tasks such as starting or actioning a workflow, executing a SmartObject or rendering a SmartForm.
  - The UPN property must be configured and populated for all users in the tenant that use Appit.

The **K2 Appit for SharePoint** application requires the permissions detailed in Table 3 to install, configure and use the application on each **site**.

Table 3: Permissions Required for SharePoint Online Installation

Action	Permissions Required	Notes
Add, Grant Permissions or Update the application	• Site Collection: **Site Collection Administrator** • App Catalog: **Read**	• The users and groups configured as **End Users** when provisioning the App Catalog site are granted **Read** permissions. • The user can have **Read** rights to the App Catalog site or the Application file directly.
Remove the application	Web: **Full Control**	The default **Owners** group typically has **Full Control** permission
Configure the application via the K2 Registration Wizard	Web: **Full Control**	
Start a Workflow, View a Workflow or access K2 Application User Pages, including Learn, Get Help, End User License, Reports, SmartObjects and the K2 Designer	• Web: **Read** • K2 Solution Participants: **Read**	• The default **Visitors** group typically has **Read** permissions. • The default **Owners** and **Members** groups typically have **K2 Solution Participants** rights. • On deployment of the workflow, the **K2 Solution Participants** are added to the process **Start** and **View** rights
Create and Edit Data, Forms, Workflows and Reports	• Web: **Edit** • K2 Solutions Designers: **Edit**	• The default **Members** group typically has **Edit** permissions. • The default **Owners** and **Members** groups typically have **K2 Solution Designers** rights.
Run the Registration Wizard, Uninstall, Configure K2 Permissions and Synchronize Groups	• Web: **Design**	• The default **Owners** group typically has **Full Control** permissions which includes **Design** permissions.

For more information, see **Add apps for SharePoint to a SharePoint 2013 site**
https://technet.microsoft.com/en-us/library/fp161231.aspx

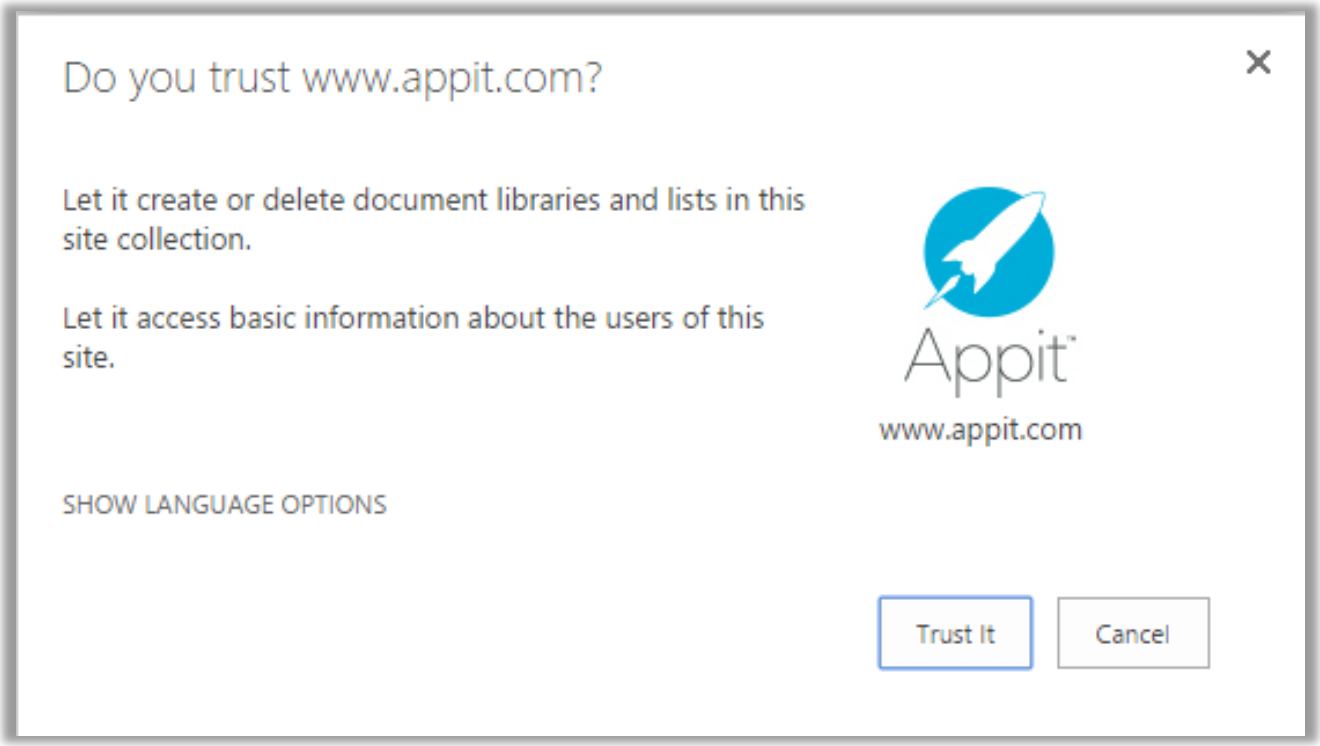
K2 FOR SHAREPOINT 2013 ON-PREM HYBRID

K2 Appit for SharePoint on-prem is not supported unless the SharePoint on-prem local Active Directory is synced with an AAD instance. The recommendation if you have both on-prem and online SharePoint environments, you should use the same AAD instance.

Note: Appit connects to an on-prem SharePoint server using a high-trust certificate from Appit and does not go through trust.k2.com or use the Common Consent flow.

The K2 Appit for SharePoint application requests the following permission scopes:

- **Let it have full control of this site collection**: This maps to Scope="http://sharepoint/content/sitecollection" and Right="FullControl"
- **Let it access basic information about the users of this site**: This is the base scope required for all applications.
- **Let it share its permissions with other users**: This maps to AllowAppOnlyPolicy=true



Installation, Configuration and Use of K2 for SharePoint

The K2 Appit for SharePoint application requires the following permissions to install and configure the application on ***each site***. Users also require permissions to access and perform various K2 actions within SharePoint.

Table 2: Permissions Required for SharePoint On-Prem Installation

Action	Permissions Required	Notes
Add, Grant Permissions or Update the application	• Site Collection: **Site Collection Administrator** • App Catalog: **Read**	• The users and groups configured as **End Users** when provisioning the App Catalog site are granted **Read** permissions. • The user can have **Read** rights to the App Catalog site or the Application file directly.
Remove the application	Web: **Full Control**	The default **Owners** group typically has **Full Control** permission
Configure the application via the K2 Registration Wizard	Web: **Full Control**	The default **Owners** group typically has **Full Control** permission
Start a Workflow, View a Workflow or access K2 Application User Pages, including Learn, Get Help, End User License, Reports, SmartObjects and the K2 Designer	• Web: **Read** • K2 Solution Participants: **Read**	• The default **Visitors** group typically has **Read** permissions. • The default **Owners** and **Members** groups typically have **K2 Solution Participants** rights. • On deployment of the workflow, the **K2 Solution Participants** are added to the process **Start** and **View** rights.
Create and Edit Data, Forms, Workflows and Reports	• Web: **Edit** • K2 Solutions Designers: **Edit**	• The default **Members** group typically has **Edit** permissions. • The default **Owners** and **Members** groups typically have **K2 Solution Designers** rights.
Run the Registration Wizard, Uninstall, Configure K2 Permissions and Synchronize Groups	• Web: **Design**	The default **Owners** group typically has **Full Control** permissions which includes **Design** permissions.

For more information, see **Add apps for SharePoint to a SharePoint 2013 site**
https://technet.microsoft.com/en-us/library/fp161231.aspx

COMMON CONSENT

Common Consent allows K2 to reduce the number of trust prompts that it may need to show when integrating Appit with other sites in a site collection or, more broadly, across the tenancy or farm. This represents the first in what is likely to be a trend to extend OAuth more into the enterprise arena from its personal, 1:1 beginnings.

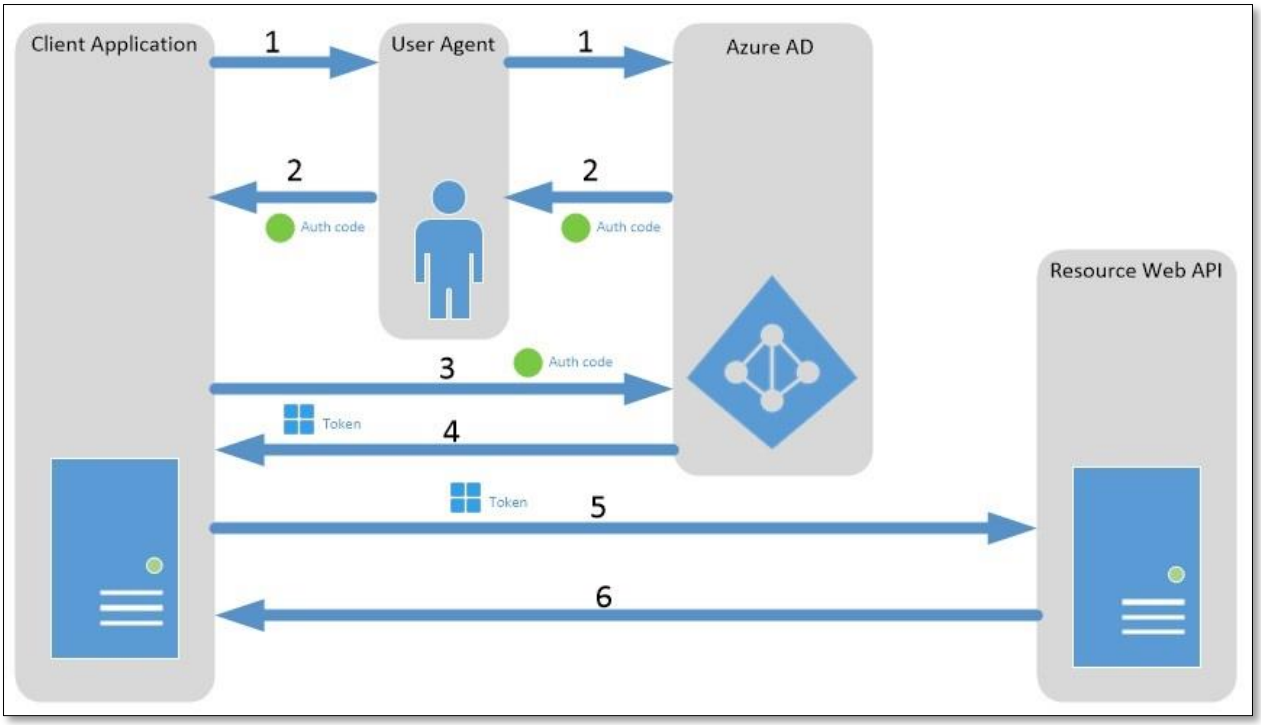


Diagram 7: Common Consent flow. Image from Authorization Code Grant Flow (https://msdn.microsoft.com/en-us/library/azure/dn645542.aspx)

A Common Consent token flow looks like the following:

(1) The client application makes the request for a resource. The user agent passes along this request to Azure AD.
(2) Azure AD verifies the user and passes back an authorization code to the client application
(3) The client application sends the auth code directly to AAD
(4) AAD passes back an access token
(5) The client application uses the access token to query the online resource (such as SharePoint, Exchange, Lync, etc.)
(6) The resource sends back the requested data

The importance of Common Consent here, and the flow, is that the trust has been promoted up to Azure AD instead of sitting at the SharePoint, Exchange or K2 Web API level, so a common trust consent applies to all properties managed by Azure AD.

MOBILE APPS & OFFLINE DATA STORAGE

K2 Mobile Apps communicate with an Appit environment through a URL-based K2 endpoint. This endpoint is typically encrypted via SSL/HTTPS with authentication provided by Azure AD. The connection from the device to the Appit server can take place over any valid network connection, whether that's a WiFi access point, VPN or via cellular data service. All communication with the Appit server is encrypted regardless of the connection type.

The K2 Web API lets the app know that it supports AAD authentication which causes the app to prompt the user to authenticate against https://login.microsoft.net/common. This uses an OpenID Connect authentication flow. Once K2 Mobile has the token, the app sends it to the K2 Web API service which then verifies that the token is valid for the client's Realm ID based on the claims mappings for the Appit instance.

The session state and cookies for SmartForms are handled in the same way as they are handled in a browser.

IOS

Data for K2 Mobile is stored in a SQL Lite database and credentials are encrypted using the keychain.

When using offline forms on an iDevice, the local data is stored using Apple's Complete Protection (NSFileProtectionComplete) class key, which means the data is inaccessible when the device is locked, until the user unlocks the device again. Note: K2 Mobile 1.2.1 or later is required for this protection.

Important: On devices that have been jailbroken, data is not secure. It is recommended not to install K2 Mobile on jailbroken devices.

ANDROID

Authorization tokens used by K2 Mobile are exchanged like the iOS version of K2 Mobile. For SmartForms it works the same as in a browser, which relies on Federated Authentication with cookies. For web service calls using OAuth, the validity of the token is determined by the token issuer (Azure AD/ACD), which follows all the standard functionality of expirations and revocations. The OAuth token is encrypted and uses Microsoft classes to perform OAuth token storage and retrieval. The Android app does not honor the Application Timeout setting at this time.

Passwords are stored and encrypted on the device using a SHA-256 encryption algorithm. When using OAuth as the authentication mode, there is no password present, just the OAuth token. If the phone is backed up the passwords remain encrypted.

K2 Mobile files stored on the device are encrypted using the encryption APIs provided in the Java.Security package, as the Android does not provide native file-level encryption.