Fabric - Get data SharePoint from Fabric

SYSM-354 - Jira project doesn't exist or you don't have permission to view it.

- Data retrieval strategy from SharePoint:
  - Files, Lists, custom applications, OneLake File Explorer
  - security, best practices

Version	Date	Description	Contributor
V0.1	30 Mar 2026	Initial document	COLOMBANI Théo
V0.2	07 Apr 2026	Added to the wiki	COLOMBANI Théo
V0.3	15 Apr 2026	Added Shortcut specific guidelines (section 4) Added authentification part to be set up (section 5)	COLOMBANI Théo

---

1. Axis — Load into Lakehouse *Files*

1.1 OneLake Shortcut (SharePoint / OneDrive)

Description
Logical link exposing SharePoint folders in OneLake without data duplication.

Functioning

- Shortcut points to a SharePoint folder (folder-level only)
- Data remains in SharePoint and is accessed virtually
- Accessible across Fabric workloads

Key capabilities

- Data virtualization (no physical copy)
- Automatic synchronization with source changes
- Unified access through OneLake

Advantages

- No pipelines or ETL required
- No data duplication
- Fast implementation
- Unified access layer

Limitations (decision drivers)

- Folder-level granularity only
- Performance dependent on SharePoint (latency, throttling)
- No control over ingestion (no filtering, no incremental logic)
- Runtime dependency on source availability
- Not suitable when strong data isolation or historization is required

1.2 Custom ingestion — API (Notebook or Pipeline) Files

Description
Extraction via Microsoft Graph or SharePoint REST API and storage in Lakehouse Files.

Execution models

- Notebook (Spark / Python)
- Data Pipeline:
    - Web Activity (REST calls)
    - Copy Activity with API source

Functioning

- API calls to retrieve files or metadata
- Data written into OneLake Files

Key capabilities

- Supports full SharePoint surface (files, folders, metadata)
- Custom ingestion logic (filtering, incremental, structuring)
- Can be orchestrated via pipelines

Advantages

- Full flexibility on ingestion logic
- Ability to implement incremental loads (delta, watermark) at bronze layer
- Can handle complex folder structures and edge cases
- Works even when no native connector exists

Limitations (decision drivers)

- Requires handling:
    - authentication (OAuth / Service Principal)
    - pagination (@odata.nextLink)
    - API rate limits / throttling
- More complex error handling and retry logic
- Development and maintenance effort
- Pipeline Web Activity is stateless (no built-in transformation)
- Copy Activity / Web Activity require manual schema handling

---

2. Axis — Load into Lakehouse *Tables*

2.1 Shortcut with transformation Delta Tables

Description
Use of SharePoint shortcut with transformation to project files into Delta tables.

Functioning

- Shortcut exposes files
- Transformation step converts them into structured tables
- Tables remain synchronized with source

Key capabilities

- Automatic file-to-table conversion
- Continuous synchronization
- Direct consumption in SQL / BI

Advantages

- No pipeline required
- Direct analytical usability
- Integrated with OneLake

Limitations (decision drivers)

- Strong dependency on source file structure and quality
- Limited transformation capabilities compared to ETL
- Limited control over schema evolution
- Debugging and lineage less explicit than pipeline-based ingestion

2.2 Mirroring (SharePoint Lists)

Description
Replication of SharePoint Lists into OneLake as Delta tables.

Functioning

- Connection to SharePoint list
- Continuous replication into Fabric tables
- Automatic synchronization

Key capabilities

- Near real-time data replication
- Native Delta format
- No ETL required

Advantages

- Continuous synchronization
- Simplified ingestion architecture
- Direct usability for analytics

Limitations (decision drivers)

- Limited to structured data (lists only)
- Limited transformation capabilities during ingestion
- Dependency on mirroring feature availability and scope
- Limited control over ingestion logic (filters, enrichment)
- Schema evolution handled automatically but with limited customization

2.3 Custom ingestion — API (Notebook or Pipeline) Tables

Description
API-based extraction with transformation and direct load into Delta tables.

Same comments from Section 1.2 Custom ingestion — API (Notebook or Pipeline) Files

---

3. Considerations

API usage (Notebook vs Pipeline)

Notebook

- Better suited for:
    - complex transformations
    - large data processing
    - advanced logic (joins, enrichment)

Pipeline (Web / Copy Activity)

- Better suited for:
    - orchestration
    - simple ingestion patterns
    - metadata-driven ingestion

Security

- Authentication methods:
    - Organizational account
    - Workspace identity

Service principal recommended

- API-based approaches require:
    - token management
    - permission configuration (e.g. Sites.Read.All)

---

4. MATRIX(s)

Synthesis

Data type	Load target	Options
Files	Files	Shortcut / API (Notebook or Pipeline)
Files	Tables	Shortcut + transformation / API (Notebook or Pipeline)
SharePoint Lists	Tables	Mirroring / API (Notebook or Pipeline)

Criteria

Criteria	Shortcut (Files)	Shortcut + Transform (Tables)	Mirroring (Lists)	API via Notebook	API via Pipeline (Web / Copy)
Data movement	No copy (virtual access)	No copy (virtual + projection)	Physical copy (replication)	Physical copy	Physical copy
Latency / freshness	Near real-time (source-driven)	Near real-time	Near real-time sync (incremental)	Depends on orchestration	Depends on orchestration
Transformation capabilities	None	Limited	Limited	Full (Spark / code)	Limited (mapping / chaining)
Incremental / CDC logic	Not supported	Limited / implicit	Built-in incremental sync	Fully customizable	Manual implementation required
Handling complex structures	Limited (folder-based only)	Limited	Not applicable (structured only)	Strong capability	Moderate (complex via chaining)
Control over ingestion logic	None	Low	Low	Full	Medium
Operational complexity	Very low	Low	Low	High	Medium
Dependency on source availability	High	High	Low	Low (after ingestion)	Low (after ingestion)
Schema control / evolution	None	Limited	Limited	Full control	Medium control
Cost (compute / storage)	Low	Low	Free	Higher (compute + dev)	Medium (pipeline runs)

Supported data types	Files only	Files (JSON, CSV, PARQUET, EXCEL) (structured)	SharePoint Lists only	All (files + lists)	All (files + lists via API)

4. Technical solutions (recommended - SharePoint Shortcuts)

- P1 : SharePoint Shortcuts,
  - Triggers on OneLake Events ? **-> Trigger Events not working for shortcuts.**
  - Directly to Silver Tables Lakehouse with auto transform in delta (see Référence) **-> newly working for .xlsx to delta table**
  - or to files zone Lakehouse (csv shortcut) then transformation to silver tables
- Prerequisites : folder hierarchy for files & Service Principal (or Workspace Identity). One shortcut = one folder
- See also Limitations : https://learn.microsoft.com/en-us/fabric/onelake/create-onedrive-sharepoint-shortcut#limitations

Limitations

The following limitations apply to SharePoint shortcuts:

- OneLake doesn't support shortcuts to personal **or OnPremise** SharePoint sites. Shortcuts can only connect to enterprise SharePoint sites **and OneDrive for Business.**
- Based on Azure ACS retirement, Service Principal authentication will not work for SharePoint tenants created after Nov 1st, 2024.
- SharePoint and OneDrive Shortcuts are supported only at folder level and not at file levev

---

5. Security & Authentication Guide

SharePoint Microsoft Fabric (Shortcuts)

1. Authentication Overview

SharePoint / OneDrive shortcuts in Fabric support three authentication methods:

- Organizational account
- Workspace Identity
- Service Principal

This guide focuses on the two **recommended enterprise patterns**:

- Workspace identity : To use workspace identity authentication for OneDrive or SharePoint shortcuts, you need to grant your workspace identity access to the OneDrive or SharePoint site.
- Service Principal :To use service principal authentication, register an application in Microsoft Entra ID and create a client secret. Then, grant the service principal access to your SharePoint site using Microsoft Graph. The service principal needs at least **read** permission on the SharePoint site

---

2. Summary — Workspace Identity vs Service Principal

Create a OneDrive or SharePoint shortcut

Fabric SharePoint Shortcut - potentials issues

Criteria	Workspace Identity	Service Principal
Definition	Fabric-managed identity (auto-created service principal)	Entra ID application identity
Credential management	Fully managed (no secrets)	Requires secret or certificate
Setup complexity	Low	Medium
Governance control	Limited to Fabric scope	Full control via Entra ID
SharePoint authorization	Requires explicit site access	Requires explicit site access
Security risk	Low (no credential exposure)	Medium (secret lifecycle)

Cross-platform usage	Limited	Strong (usable across services)
Lifecycle	Tied to workspace	Independent lifecycle
Recommended usage	Simplicity / low ops	Enterprise governance / control

---

3. Workspace Identity

3.1 Description

A Workspace Identity is a **Fabric-managed service principal automatically created and maintained by the platform**.

It allows Fabric to authenticate to external systems (including SharePoint) **without managing credentials**.

3.2 Configuration Steps (Shortcut context)

- Configure workspace identity authentication
- Authenticate with workspace identity

1. Create a Workspace Identity in Fabric
   - Workspace settings Workspace Identity
   - Requires admin role
2. Retrieve the identity in Entra ID
   - Same name as workspace
   - Copy Application ID
3. Grant access to SharePoint site
   - Add the identity to the site permissions
   - Minimum: read access
4. Create the Shortcut
   - Select **Workspace Identity** as authentication method

3.3 Best Practices

- Use Workspace Identity when supported by the connector
- Restrict access to specific SharePoint sites (least privilege)
- Limit who can manage the workspace identity (admin role only)
- Monitor identity usage via Entra audit logs

3.4 Limitations (important for design)

- Not supported in all connectors or scenarios
- Not compatible with cross-tenant access
- Lifecycle tied to workspace (deletion = identity loss)
- Limited governance outside Fabric
- Must be excluded from certain Conditional Access policies to function properly

---

4. Service Principal

4.1 Description

A Service Principal is a **non-interactive identity registered in Microsoft Entra ID**, used for application-to-application authentication.

It provides **full control over permissions and lifecycle**, making it suitable for enterprise scenarios.

4.2 Configuration Steps (Shortcut context)

Securely connect Microsoft Fabric to SharePoint

1. Create an App Registration in Entra ID

    - Generate:
        - Client ID
        - Client Secret or Certificate
2. Assign API permissions
    - SharePoint / Graph permissions
    - Prefer **Sites.Selected**
3. Grant access to SharePoint site
    - Explicitly authorize the Service Principal
    - Required in addition to API permissions
4. Configure authentication in Fabric
    - Select **Service Principal** in Shortcut
    - Provide credentials

4.3 Best Practices

- Use **Sites.Selected** instead of tenant-wide permissions
- Prefer **certificate-based authentication** over client secrets
- Store credentials in **Azure Key Vault**
- Rotate secrets regularly
- Use dedicated Service Principals per environment (Dev / Prod)

4.4 Limitations (important for design)

- Requires credential lifecycle management
- Higher setup complexity
- Risk of misconfiguration (permissions or secrets)
- Requires dual configuration:
    - Entra ID permissions
    - SharePoint site-level authorization
- SharePoint has its own authorization layer API permission alone is not sufficient

---

5. Key Design Considerations

5.1 Authentication vs Authorization

- Authentication = identity (Workspace Identity / Service Principal)
- Authorization = access granted in SharePoint

Both must be configured correctly.

5.2 Choosing between Workspace Identity and Service Principal

Key decision drivers:

- Need for **centralized governance** Service Principal
- Need for **low operational overhead** Workspace Identity
- Need for **cross-platform reuse** Service Principal
- Need for **simplified setup** Workspace Identity