iMaintenance Deployment & Setup Guide

Connected Worker Solutions

Title and Copyright

Copyright and **Terms of Use** page for **iMaintenance Installation**.

Installation Guide for **iMaintenance**, a *Connected Office Worker Solution*.

Release Version: *2507*

Release Date: *03 October 2025*

Published Date: *03 October 2025*

Document Version: *2.0*

Primary Author: Innovapptive Inc.

Preface

Understand audience, know related documents and products and conventions followed in this document.

Intended Audience

This guide is intended for **SAP Basis/IT platform engineers, integration architects, and security admins** who own everything **outside the product UI**—from provisioning and data seeding to SAP connectivity and API security.

Document Conventions

Table 0-1 Conventions followed in the document

Convention	Meaning
boldface	Indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Indicates book titles, emphasis, or placeholder variables for which you supply values.
`monospace`	Indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter

Related Products & Solutions

- Work Order Management
- Inventory and Warehouse Management
- Analytics and Dashboards

Contact Innovapptive

For information on Innovapptive products, visit the Innovapptive's Support Portal at http://helpdesk.innovapptive.com. The updates to this document are published on this support portal. Check this website periodically for updated documentation.

For additional information about this document, send an email to documentation@innovapptive.com.

Contents

1. Introduction

The **iMaintenance Deployment & Setup Guide** gets iMaintenance running safely in your enterprise landscape.

This guide is written for **SAP Basis/IT platform engineers, integration architects, and security admins** who own everything **outside the product UI**—from provisioning and data seeding to SAP connectivity and API security.

What you will find in this guide

- **Application Environment Setup**: Set prerequisites, seed core JSONs to MongoDB, start SAP inbound services, and verify collections/statuses for a clean first run.
- **SAP Roles & Authorizations**: Map iMaintenance roles, import transports via STMS, and update required authorization objects.
- **API Exposure on SAP BTP**: Create API providers/proxies, connect on-prem via Cloud Connector, and apply policy-based controls in SAP API Management.
- **Edge Security**: Enforce IP allowlisting at the infrastructure edge and at the API gateway; define AccessControl policies to restrict access.
- **Auth Hardening**: Configure OAuth 2.0 and implement CSRF token acquisition/ propagation using AssignMessage and ServiceCallout policies.
- **Ops & Troubleshooting**: Fix metadata issues, clean caches, and validate OData service readiness.

2. Foundation Setup for iMaintenance Deployment

Before iMaintenance can connect to SAP and expose APIs, the core environment must be initialized. This setup includes seeding MongoDB with configuration data, enabling SAP inbound services, mapping dependencies, and starting the microservices that power the application.

Note:
Run these steps in sequence. Skipping or misordering tasks can result in broken integrations, failed API calls, or missing configurations. By the end of this phase, you will have a clean, fully initialized environment ready for SAP connectivity and API exposure.

What You Will Do in This Phase:

- Verify Prerequisites *(on page 9)*: confirm tools, credentials, and seed data sources
- Seed MongoDB with Core Data *(on page 10)*: load JSON configurations and validate collections
- Insert Seed Data *(on page)*:
- Insert SAP Data into MongoDB *(on page 12)*: run inbound services and confirm integration statuses
- Map Dependent Fields *(on page 13)*: align Plant and Asset references in MongoDB
- Clear Existing MongoDB Collections (Optional) *(on page 10)*: reset environment when reinitializing or switching systems
- Configure Application Microservices *(on page 14)*: start the functional services powering iMaintenance
- Set Up Cron Jobs *(on page 15)*: enable background sync and asset refresh tasks

2.1. Verify Prerequisites

Before initializing the environment, ensure the following prerequisites are in place. Missing even one will cause deployment failures later.

- **SAP Connectivity** → Verify that the required SAP APIs are active and reachable from your network.
- **Seed Data Access** → Confirm access to the latest JSON seed files from the *devdb* or provided Innovapptive repositories.
- **System Tools Installed** → Ensure the following tools are available on the host where setup will run:
    - Node.js and NPM
    - Access to MongoDB Atlas
    - Access to AWS S3 console
    - Access to project repositories (RACE backend, seed data folders)
- At the end of this step, you can confidently source seed data and connect to both SAP and MongoDB without dependency issues.

2.2. Clear Existing MongoDB Collections (Optional)

When switching SAP environments or reinitializing the system, you may need to clear existing MongoDB collections. This provides a clean start for reseeding and prevents conflicts with outdated data.

1. Navigate to the reset script: `RACE/backend/SapInbound/src/utilities/deleteCollections.js`
2. Run the script: `node deleteCollection.js`
3. Ensure the `nosql` object in the script contains valid DB credentials before execution.

2.3. Seed MongoDB with Core Data

MongoDB must be initialized with seed data so that iMaintenance has its baseline configuration — roles, forms, dashboards, preferences, and integration templates. Without this step, the application will launch with missing or incomplete data.

To seed MangoDB with core data:

1. Source Latest JSON Files
    - Location: `RACE/backend/Seed Data/JSON`
    - Retrieve the latest configuration, permission, integration, and form-related JSONs from *devdb*.
    - If configurations are not coming from SAP, include the default configuration for `attachments` and ensure **Plant** is added manually in the DB.
2. Verify Required JSON Files
    - Integrations
    - SAP Inbound
    - AlertString
    - Configurations
    - Dashboard
    - Language (Usecase – Language)
    - MasterBo (Reports)
    - MasterForm, MasterFormPage, MasterFormPageSection, MasterFormPageSectionField, MasterFormPageSectionFieldLogic
    - MobileAppPreference
    - Permission
    - Report
    - RiskMatrix
    - Role
    - User
    - Widget, WidgetConfiguration
3. Confirm Storage Locations

- **AWS S3 Bucket:** `race2.0resources`
- **Drive Link:** `RACE2.0 SEED DATA` (Google Drive folder)

4. Update MongoDB Connection Configuration
   - File Path: `RACE/backend/Seed Data/connection/mongo.js`
   - Modify the `nosql` object with the appropriate values:
     - Database
     - Host (only if switching MongoDB clusters)
     - Username
     - Password
5. Run Seed Script and wait for the process to complete.

```
cd RACE/backend/Seed Data

node index.js
```

2.4. Insert SAP Data into MongoDB

This step ingests master and transactional data from SAP into MongoDB. It enables iMaintenance to work with real SAP objects instead of placeholders. By the end, inbound services will be active, and collections updated with the correct status.

To insert SAP Data into MongoDB

1. Update SAP MongoDB Connection
   - File Path: `RACE/backend/SapInbound/src/direct_adaptor/lib/mongo.js`
   - Modify the `nosql` object with correct values (Database, Host, Username, Password).
2. Start SAP Inbound Microservice

   `cd RACE/backend npm run inbound`

   Monitor terminal logs to confirm data ingestion is running.
3. Monitor MongoDB Atlas

- Navigate to the **Integrations** collection.
- Check the `status` field:
  - `0` → Not Active
  - `1` → Active
  - `3` → Running
  - `4` → Failed
- Wait until logs confirm **completion**.

2.5. Map Dependent Fields

Some fields, such as **Plant**, require mapping in MongoDB to ensure correct alignment between SAP data and iMaintenance. This step uses the seed mapping script to establish those dependencies.This topic explains how to map dependent fields, such as Plant, in MongoDB by running the seed mapping script located at `RACE/backend/SapInbound/src/integration-v2/startSeed.js`. It also outlines final steps, including stopping the SAP Inbound container and updating the Integrations collection in MongoDB Atlas when switching SAP environments.

To map dependent fields:

1. Run Seed Mapping Script
   - File Path: `RACE/backend/SapInbound/src/integration-v2/startSeed.js`
   - `node startSeed.js`

     This maps dependent fields (e.g., Plant) correctly in MongoDB.
2. Finalize Mapping
   - Stop the SAP Inbound microservice after the script completes.
   - If switching SAP environments, update the **Integrations** collection in MongoDB Atlas to reflect the new environment.

2.6. Configure Application Microservices

iMaintenance relies on multiple micro-services to handle authentication, integration, asset data, forms, reports, and AI-driven tasks. These services must be deployed and running for the application to function end-to-end.

Key Micro-services and Their Purpose

Micro-service	Purpose
ai-chatbot-service	Digital Work Instruction
ai-localisation-service	Language Translation
asset360-service	Asset 360's Dashboard & Data Prep
alert-api-service	Alerts & Push Notifications
auth-api-service	Authentication
authored-form-api-service	Field Configuration Forms
erp-master-data-api-service	General Dropdowns
erp-outbound-consumer-service	Post Transaction Data to SAP
erp-outbound-producer-service	Listen for Transaction Events
integration-manager-api-service	(Details not specified)
master-configuration-api-service	Plant/Asset/Functional Location Master Data
mobile-app-preference-api-service	Global/Transactional Preferences
my-forms-api-service	Form Templates
operator-rounds-api-service	Observations, Issues, Actions
report-api-service	Dashboard Reports
sap-inbound-api-service	Initial SAP Data Ingestion
settings-api-service	General Settings
storage-api-service	Attachment Handling
user-service-api-service	Users, Roles, Permissions, Groups
workorder-api-service	MCC (Work Order Management)
platform-dev → auth-service	oAuth (Google/Microsoft) Authentication

Micro-service	Purpose
AI Issue Detection Service	Autonomous Issue Detection
AI WorkOrder	Autonomous Work Order Creation

Deployment Notes

- Each microservice is typically packaged as a **Node.js service** (often containerized).
- They should be started and monitored individually or through your orchestration platform.
- Validate that each service is running before proceeding to cron jobs and higher-level configurations.

2.7. Set Up Cron Jobs

Scheduled jobs automate recurring tasks in iMaintenance, such as syncing incremental updates and refreshing asset dashboards. These must be configured to ensure the system stays current without manual intervention.

Cron Jobs

- **delta-cronjob** → Synchronizes incremental updates from SAP.
- **asset360-cronjob** → Refreshes Asset360 dashboard data.

Supporting Components

- **Messaging Queue** → `rabbitmq-service` handles asynchronous communication.
- **Distributed Caching** → `redis-cluster` maintains high-speed cache for frequently accessed data.
- **CWP Portal** → `race-2-client` connects the cron jobs and microservices with the portal.

Deployment Notes

- Use standard Linux `crontab` to schedule these jobs at required intervals.
- Validate execution logs after the first run to confirm jobs are firing correctly.
- Ensure RabbitMQ and Redis are up and running before enabling cron jobs.

3. Configure SAP iMaintenance Roles and Authorizations

Before iMaintenance can integrate with SAP, specific roles and authorizations must be configured. These roles control access to master data, transactional posting, and integration endpoints.

Authorization objects ensure that only permitted users and services can perform critical actions such as creating work orders, posting goods movements, or reading plant data.

> **Note:**
>
> Missing or incomplete authorizations will result in failed API calls, restricted access to master data, or posting errors in SAP. Proper configuration at this stage ensures secure, reliable communication between iMaintenance and SAP.

What you will do this is step:

3.1. Configure SAP iMaintenance Roles

Innovapptive provides **pre-defined SAP iMaintenance roles** as transport requests. These must be imported into your SAP systems (ECC, S/4HANA, or NetWeaver Gateway) based on your deployment landscape.

Manual role creation is only required if transports are not available.

To configure SAP iMaintenance roles:

1. Import the pre-delivered transport requests into the target SAP system (ECC, S/4HANA, or Gateway) using **STMS**.
2. In transaction **SU01**, assign the imported roles to the appropriate technical or end users.
    - Adjust roles if customer-specific modifications are needed.
    - Ensure assignments align with the user's job functions.
3. If transports are not available, manually create the roles using the **transaction codes and authorization objects** listed in this guide.

3.2. Define SAP iMaintenance Authorization Requirements

Access to iMaintenance requires specific SAP authorizations. These are bundled within the Innovapptive-provided roles but may also need to be tailored to match your organization's functional scope and security policies.

To assign authorizations:

1. In transaction **SU01**, assign either the **pre-delivered Innovapptive roles** or your customer-specific roles to the appropriate users.
2. Ensure access is consistent across **Development, Quality, Pre-Production, and Production** systems.
3. Review and adjust authorizations as required by your **internal security and compliance policies**.

3.3. Use Pre-Defined SAP Roles for iMaintenance

Innovapptive provides specific roles for ECC, S/4HANA, and Gateway systems. These roles include all necessary authorizations to perform standard iMaintenance tasks.

Table 3-1 Roles for SAP Systems (ECC/S/4HANA)

Role Name	Description	Transactions	Authorization Objects
ZINV_IMAINT_ECC_END_USER	Innovapptive - iMaintenance - End User - Ecc Authorizations	CAT2, CATS_APPR_LITE, CS03, CV01N, CV02N, CV03N, IA03, IA07, IB03, IE01, IE02, IE03, IH01, IH03, IK01,	S_RFC, S_RFCACL

Table 3-1 Roles for SAP Systems (ECC/S/4HANA) (continued)

Role Name	Description	Transactions	Authorization Objects
		IK02, IK03, IK11, IK13, IW21, IW22, IW23, IW31, IW32, IW33, IW34, IW41, IW42, IW45, MIGO	

Table 3-2 Role for SAP Systems (S/4HANA/Gateway)

Role Name	Description	Authorizations
ZINV_IMAINT_NWG_END_USER	Innovapptive - iMaintenance - End User - Gateway Authorizations	S_RFC, S_RFCACL, S_SERVICE, S_TABU_DIS, S_USER_GRP

You can use the provided roles as is or copy them to match your internal naming conventions before generating them for use.

3.4. Deploy SAP iMaintenance Roles Using Transports

Innovapptive provides pre-delivered transport requests containing the required iMaintenance roles. Importing these transports ensures consistent role configuration across ECC, S/4HANA, or Gateway systems.

To deploy roles using transports:

1. Prepare the transport files
    - Extract the archive (`.zip` or `.rar`) received from Innovapptive.
    - Locate the files:
        - **Co-files** (start with K9*) → copy to `/usr/sap/trans/cofiles`
        - **Data files** (start with R9*) → copy to `/usr/sap/trans/data`
2. Import the roles using STMS

a. Log in to the target SAP system (ECC, S/4HANA, or Gateway).

b. Open transaction **STMS_Import**.

c. In the Import Queue, choose **Extras > Other Requests > Add**.

d. Enter the transport number and confirm.

e. Select the transport request and click the **Truck** icon to start the import.

f. Enter the **target client number**.

g. Select the following options:

- *Leave Transport Request in Queue for Later Import*
- *Ignore Invalid Component Version*

h. Confirm to begin the import process.

3. Verify the imported roles
   - After import, confirm the roles are available in **PFCG**.
   - In **SU01**, assign the roles to the iMaintenance technical user or other required users.
4. Update S_SERVICE authorizations

a. Open transaction **SE16/SE16N/SE11**.

b. Enter the table name **USOBHASH**.

c. Use the following search parameters:

Object Type	Object Name
IWSG or IWSV	`/INVICL/ICWP_INTEGRATION_SUITE_SRV_0001`

d. From the results, copy the **hashed service names** (30-character alphanumeric values).

e. Add these values to the user's **S_SERVICE** authorization object (field: SRV_NAME).

3.5. Handle Transport Dependencies (If Any)

In some cases, the role transports provided for iMaintenance may have **dependencies on base packages**. To avoid import errors, transports must be applied in the correct order.

- Check the transport documentation for dependency information.
- Import base transports (e.g., Innovapptive packages) before importing dependent role transports.

Troubleshoot Transport Import Issues

If issues occur during transport import, capturing logs and screenshots helps Innovapptive support resolve them quickly.

1. Take screenshots of any error messages in STMS.
2. Save the **transport import log** from the Import Queue.
3. Share the transport numbers, screenshots, and logs with your designated Innovapptive SAP Basis contact.

4. Secure API Access with IP Whitelisting

Secure connectivity for Innovapptive-hosted applications is enforced by restricting inbound traffic to trusted client IPs and permitting outbound traffic only to approved Innovapptive domains. This ensures that only authorized systems can exchange data with iMaintenance while preventing unauthorized access.

Responsibilities at a Glance

Layer	Responsibility
WAF (Ingress)	Managed by Innovapptive using AWS WAF to allow access only from whitelisted IPs
Firewall (Egress)	Managed by the customer to permit outbound access to specific Innovapptive domains

To maintain security and seamless communication between customer environments and Innovapptive's cloud services:

- **Ingress filtering (AWS WAF):** Only specific client IPs are allowed to access Innovapptive services.
- **Egress filtering (Customer Firewall):** Outbound traffic is restricted to the domains required for iMaintenance functionality.

This dual-layer approach prevents unauthorized access while ensuring stable, authorized connectivity across all environments.

4.1. IP Allowlisting Through AWS WAF

Innovapptive manages inbound access to iMaintenance services through AWS Web Application Firewall (WAF). Only customer-approved IPs are allowed, while all others are blocked

Prerequisites

- Enable AWS WAF on the relevant AWS resource.
- Provide a list of static public IPs or CIDRs.

1. Create an IP Set
    a. Navigate to the AWS WAF Console → *IP sets*.
    b. Click **Create IP set**.
    c. Enter a name (e.g., `WhitelistedIPs`).
    d. Select the applicable Region (Global or Regional).
    e. Add the customer-provided IP addresses or CIDRs.
    f. Click **Create**.
2. Create or Edit a Web ACL
    a. In the WAF Console, go to *Web ACLs*.
    b. Either create a new Web ACL or select an existing one.
    c. Attach the Web ACL to the required AWS resource (ALB, CloudFront, etc.).
3. Add an Allow Rule
    a. Within the Web ACL, go to *Rules* → **Add rule**.
    b. Rule name: `AllowOnlyWhitelistedIPs`.
    c. Statement: Use the previously created IP set.
    d. Action: **Allow**
4. Add a Block Rule
    a. Rule name: `BlockAllOthers`.
    b. Action: **Block**.
    c. Alternatively, set the Web ACL's default action to Block.
5. Deploy and Verify
    a. Ensure the Allow rule is listed **above** the Block rule in priority order.
    b. Save and deploy the Web ACL.
    c. Test access from an allowed IP (succeeds) and a non-allowed IP (blocked).

4.2. Domain Allowlisting at Customer Firewall

Customers must configure their network firewalls to allow outbound access from corporate networks to specific Innovapptive domains. This ensures iMaintenance services can connect seamlessly to required cloud endpoints.

Required Domains for Outbound Access

Table 4-1 File Access (S3 Resources)

Description	Domain
Innovapptive file access	innoresources.s3.amazonaws.com

Table 4-2 Authentication (Azure Active Directory)

Description	Domain
Azure AD login	login.microsoftonline.com
Legacy Azure login	login.windows.net
Microsoft auth CDN	aadcdn.msauth.net
Microsoft branding assets	aadcdn.msauthimages.net
Microsoft token service	aadcdn.msftauth.net

Table 4-3 DynamoDB Table Synchronization (AppSync)

Description	Domain
AWS AppSync end-point	appsync-api.us-east-1.amazonaws.com
AppSync WebSocket sync	hyufofsc35b67j4grrbfmjuqv4.appsync-realtime-api.us-east-1.amazonaws.com

Table 4-4 MongoDB Realm Sync

Description	Domain
MongoDB Realm WebSocket	https://ws.realm.mongodb.com
MongoDB Data API	https://data.mongodb-api.com
MongoDB Realm wildcard	*.realm.mongodb.com
MongoDB API wildcard	*.mongodb-api.com

Configuration Notes

- Outbound traffic to the above domains must be allowed on TCP port 443 (HTTPS).
- Bypass SSL inspection for these domains to prevent certificate errors or service disruptions.
- Ensure DNS resolution is not restricted for these domains.
- If your firewall or proxy does not support wildcards, the wildcard entries may need to be expanded into individual domain rules.

> **Note:**
> This list may be updated based on service expansion or region-specific deployments.

Summary of Responsibilities and Actions

Component	Allowlisting Type	Owner	Action Required
AWS WAF	IP-based (Ingress)	Inno-vapptive	Customer must share static public IPs or CI-DRs
Customer Firewall	Domain-based (Egress)	Customer	Allow outbound access to specified do-mains on TCP 443

4.3. Verify Secure API Access and Maintain

After configuring IP allowlisting, it's critical to confirm that only trusted systems can reach iMaintenance services and to keep the rules updated as environments evolve. Verification ensures the setup is working as intended, while regular maintenance keeps access aligned with changing business and IT requirements.

To verify:

1. From an **allowed client IP**, call an iMaintenance API → request should succeed.
2. From a **non-allowed IP**, repeat the call → request should be blocked (HTTP 403 or connection refused).
3. Review logs for confirmation:
    - **AWS WAF / perimeter** → blocked or allowed request entries
    - **SAP Gateway / API Management** → proxy access logs
    - **Application** → unauthorized or denied attempts

Maintain

1. Revalidate allowlists during major releases or quarterly reviews.
2. Update entries when client subnets, VPN egress IPs, or cloud endpoints change.
3. Remove decommissioned IPs/domains to keep rules minimal.
4. Document each entry with owner, change date, and business justification.

Avoid Common Pitfalls

- NAT or proxy may mask the client IP → ensure headers preserve the real source IP.
- Overly broad CIDRs (e.g., `/16`) expand attack surface → prefer `/32` or the smallest viable range.

4.4. Configure the Access Control Policy

Use SAP API Management's **Access Control** policy to enforce IP allowlisting at the API-proxy layer as an additional defense-in-depth control.

When to use

- You need proxy-level control/auditing per API.
- Compliance requires layered enforcement beyond WAF/firewall.
- Temporary, API-specific allowlists are needed without changing perimeter rules.

Follow the steps below to apply the AccessControl policy in your API Proxy:

1. Select **API proxy** and click **Policies**.
2. Click **Edit**.
3. Select **ProxyEndpoint**.
4. Select **PreFlow**.
5. Under **Traffic Management Policies**, choose **Access Control**.
6. Click on the **+** button to add the policy.
7. Enter the **Policy Name**, set the Stream as Incoming Request and click **Add**.
8. Click **Save**.
9. Deploy this revision
10. Test from allowed vs. non-allowed IPs to confirm behavior.

Important Considerations

- Enforcement is **per-proxy**; perimeter/WAF rules still apply first.
- Use `/32` for exact IPs; use CIDRs (e.g., `/24`) only when justified.
- Ensure Cloud Connector/API Gateway preserves the client IP (e.g., `X-Forwarded-For`) so the policy matches the real source.
- Keep proxy allowlists in sync with perimeter rules to avoid drift.

5. **Install Supporting Systems and iCWP Integration Suite**

Enable SAP NetWeaver Gateway and configure virus scan handling so iCWP OData services can run securely.

The tasks in this section cover:

1. Activating the Gateway runtime and applying required profile parameters.
2. Configuring virus scan handling according to your organization's security policy.
3. Verifying that Gateway is operational and aligned with both functional and security requirements.

By completing these steps, the SAP system will have a fully enabled Gateway framework with approved security controls, ready to support iCWP integration.

5.1. Verify Dependent Systems Installation

Before beginning the installation, make sure the following prerequisites are met.

Component	**Details**
SAP BTP	Business Technology Platform (BTP) access
SAP Cloud Connector	Latest version installed
SAP NetWeaver Gateway	Version 7.4 or higher with SAP_GWFND 740 SP13 or above SAP
SPAM	Version 69 or above (*required only for Add-On installation*)
SAP ECC	ECC 6.0 EHP 5 with SAP_BASIS 702 and above (recommended) Lower versions may require evaluation • If the deployment option is embedded, then the NetWeaver component should be version 7.40 SAP_GWFND SP13 or above.

SAP S/4HANA	Versions 1809, 1909, 2021, 2022, 2023 Lower versions may require evaluation
Security Mechanism	Authentication compatibility requires evaluation
System Access	SAP Basis and Security access per access sheet (minimum display access)
Portal Access	Access to CWP portal with appropriate roles assigned

5.2. Install and Verify iCWP Integration Suite Add-Ons

Install the iCWP Integration Suite add-ons using SAP's standard procedure and verify deployment in ECC and Gateway and then validate them using the steps shown below.

Follow the steps below to install the iCWP Integration Suite add-ons in your SAP environment:

1. Obtain the latest add-ons and support packages from your Innovapptive representative.

   These will typically be provided in a compressed format (e.g., .zip, .rar) via email or FTP.

2. Use the SAP standard procedure to install the add-ons.

   Reference:SAP Add-On Installation Guide.

3. Depending on your system landscape, refer to the appropriate table for the required components:

Table 5-1 ECC Add-On and Support Packages (Hub System)

Type	Add-On	Description	Dependency
Add-On	EWX0021310268_-0000000.PAT	iCWP integration suite 2506 ECC Add-on	None

Table 5-2 Gateway Add-On and Support Packages (Hub System)

Type	Add-On	Description	Dependency

Add-On	NZY0090055495_-0000056.PAT	iCWP integration suite 2506 GW Addon	None

Table 5-3 ECC Embedded Add-On and Support Packages

Type	Add-On	Description	Dependency
Add-On	MWX0021310268_-0000000.PAT	iCWP integration suite 2506 EMB Addon	None

Table 5-4 S/4HANA Add-On and Support Packages

Type	Add-On	Description	Dependency
Add-On	C230021310268_-0000019.PAT	iCWP integration suite 2506 HANA Addon	None

Note:
These tables should be populated with component names and dependencies as shared by Innovapptive.

Verify Installed Add-Ons For SAP ECC Systems

1. Log in to the SAP ECC system.
2. Execute transaction code SE37.
3. In the **Function Module** field, enter /INVICL/*.
4. Press <kbd>F4</kbd> to view available function modules.
5. Capture a screenshot of the list for validation and reference.

Verify Installed Add-Ons For SAP Gateway Systems

1. Log in to the SAP Gateway system.
2. Execute transaction code SE24.
3. In the object type field, enter /INVICL/*.
4. Press <kbd>F4</kbd> to display object classes.

The iCWP Integration Suite add-ons are installed and confirmed in ECC and Gateway systems.

5.3. Activate OData Services

Enable iCWP integration by activating the OData service, configuring the system alias where required, clearing SAP Gateway caches, and validating service responses.

After installing the iCWP add-ons, complete OData enablement in four stages to ensure endpoints are active and responding correctly.

1. Log in to the SAP Gateway system.
2. Execute transaction /IWFND/MAINT_SERVICE.
3. Locate the service /INVICL/ICWP_INTEGRATION_SUITE_SRV/.
4. From the ICF Node drop-down, select **Activate**.
5. When prompted, select the appropriate package and click **Continue**.

   The package should be created with namespace Z. In some cases, a local object may be used.

Configure System Alias (If applicable)

For ECC Systems (Route-Based Implementation)

a. Click **Add System Alias**.
b. Choose **New Entries**.
c. Assign the appropriate alias (e.g., <SID>) and save.

> **Note:**
> For S/4HANA Systems (Co-deployment) system alias is not required

Clean SAP Gateway Cache

1. Execute transaction */n/iwfnd/cache_cleanup*
   a. Select **Cleanup Cache for /INVICL/* Models** and click **Execute**.
   b. .Model Identifier - **/INVICL/ICWP_INTEGRATION_SUITE_M_0001_BE**
   c.
2. Execute transaction */n/iwbep/cache_cleanup*.

   Select **Cleanup Cache for all Models** and click **Execute**.

Validate OData Services

1. Execute transaction */n/iwfnd/maint_service* in the Gateway system.
2. Filter by **Technical Service Name** using /INVICL*.
3. For each listed service:
    a. Click **SAP Gateway Client**.
    b. On the next screen, click **Execute**.
    c. Ensure the status code **200** is returned in green.

OData service is active, required aliases (if any) are configured, caches are cleared, and validation confirms HTTP 200 responses for `/$metadata` and sample entity requests.

5.4. SAP Transactions and Tables Reference

Use this reference to quickly locate the SAP transactions and database tables most relevant for iCWP and iMaintenance operations.

These help Basis and functional teams validate configurations, troubleshoot issues, and confirm data flows without navigating through the full SAP menu paths.

Module	**Tables**	**Transaction Codes**
Work Orders	AUFK, AFIH, AFKO, RESB, MARD, MARA	IW31, IW32, IW33, IW34, IW39, IW41, IW42, IW43, IW45, MIGO, MMBE
Notifications	QMEL, QMIH, QMSM, QMUR, QMFE, QPCT	IW21, IW22, IW23, IW29
Equipment	EQUI, EQUZ, EQKT	IE01, IE02, IE03, IH08
Functional Locations	IFLO, IFLOT, ILOA	IL01, IL02, IL03, IH06
Measuring Points	IMPTT, IMRG	IK01, IK02, IK03, IK11, IK13, IK34
Time Sheets	CATSDB	CAT2
Usage Decisions	QALS, QAMV	QA03
Inspection Lots	QALS, QMFEL	QA13

5.5. Activate and Configure SAP NetWeaver Gateway

Enable SAP NetWeaver Gateway to run iCWP OData services and configure virus scan handling as per your security policy.

To activate the SAP NetWeaver Gateway:

1. In transaction SPRO, open SAP Reference IMG and navigate to SAP NetWeaver, SAP Gateway > OData Channel > Configuration > Activate or Deactivate SAP NetWeaver Gateway.
2. Select **Activate** to enable the Gateway framework.
3. Set the following profile parameters as required:

Parameter	Value / Recommendation	Purpose
`login/create_sso2_ticket`	`2`	Enable SSO ticket creation
`icm/HTTPS/verify_client`	`1`	Enforce client certificate check
`icm/HTTPS/verify_client_with_issuer`	`CN=*`	Restrict trusted issuers

Configure Virus Scan Handling

1. Open transaction **/IWFND/VIRUS_SCAN**.
2. If a virus scan profile is available, assign it to enforce file scanning for uploads and attachments.
3. If no profile is available, **Select the *Virus Scan Switched Off* check box and execute.**

   Use this only as a temporary measure, with explicit approval from your IT security team.

SAP NetWeaver Gateway is activated, required profile parameters are applied, and virus scan handling is configured either with a valid scan profile (recommended) or temporarily disabled under approved exception.

6. SAP BTP and On-Premise Integration via Cloud Connector

Set up secure connectivity between SAP BTP and your on-premise SAP systems with the SAP Cloud Connector. This enables iCWP OData services to be consumed on BTP without exposing inbound firewall ports.

As part of this, do the following:

1. Install and configure the SAP Cloud Connector to establish a secure tunnel between BTP and your network.
2. Map on-premise SAP systems to BTP subaccounts so ECC and S/4 systems can be registered for controlled access.
3. Define access control rules for iCWP OData services to restrict which services BTP is permitted to reach.
4. Validate connectivity from BTP destinations to confirm that iCWP can successfully consume SAP Gateway endpoints.

By completing these steps, you create a secure, controlled channel where iCWP services hosted on BTP can consume data from your on-premise SAP systems without compromising network security.

6.1. Validate Prerequisites

Confirm that your environment meets the technical and access requirements before installing the SAP Cloud Connector.

- A valid **SAP BTP Global Account**.
- A **subaccount** created in the Cloud Foundry environment.
- **SAP Cloud Connector** is installed on the target on-premise server.
- Administrative credentials to access Cloud Connector.

6.2. Install SAP Cloud Connector

Install the SAP Cloud Connector on the designated on-premise server to enable secure communication between your SAP systems and BTP.

To install SAP cloud connector:

1. Get the latest SAP Cloud Connector version from the **SAP Software Download Center**.
2. Launch the installer and follow the on-screen instructions to complete the installation.
3. Start the Cloud Connector service.
4. Open the Cloud Connector UI in a browser by entering the host address and port number, for example: `https://<Host Address>:<Port>`
5. Log in using the default administrator credentials.

6.3. Connect Cloud Connector to BTP Sub-Account

Connect the installed Cloud Connector to your SAP BTP subaccount to establish trust between the on-premise system and BTP.

To connect cloud connector to BTP sub-account:

1. In the Cloud Connector UI, click **Add Sub-account**.
2. Enter the following details:
    - **Region**: Select the region matching your BTP subaccount (e.g., eu10, us10)
    - **Subaccount ID**: Available in the BTP cockpit under *Subaccount Overview*
    - **Subaccount Name**: (Optional, for internal reference)
    - **Subaccount User**: A user with at least **Subaccount Administrator** role
    - **Password**: Password for the subaccount user
    - **Location ID** *(optional)*: Use if multiple Cloud Connectors are configured for redundancy or load balancing.
3. Click **Save** to establish the connection.
   You will see the screen showing the **Subaccount** details and confirmation that the connection to SAP BTP has been established.

6.4. Configure On-Premise System Mapping

Map your on-premise SAP system in Cloud Connector so BTP can reach the required internal host and port.

To configure on-premise system mapping:

1. Open the Cloud Connector UI in a browser.
2. In the **Subaccount** view, select **Cloud To On-Premise**
3. Click **Add System Mapping**.
4.

    - Select **ABAP System** (for SAP Gateway/OData). Select **SAP HANA** if exposing HANA endpoints.
5. Select **HTTPS** as the protocol.
6. Enter the **internal host/IP address and port** of the target **RISE with SAP** system hosted on the hyperscaler.
7. Define the **Virtual Host** and **Virtual Port**. For example: `<Virtual Hostname> <Port>`.
8. Click **Next**.
9. Leave **Principal Propagation** and **System Certificate for Logon** disabled unless explicitly required.
10. In the **Host in Request Header** field, select **Use Virtual Host**.
11. Enter **Description** (optional).
12. Review the configuration and click **Finish**.
13. If the configuration is correct, the status shows **Reachable**.
14. Click the **search icon** under the **Actions** column to check if the system is reachable.

6.5. Add OData Service Endpoints Under Connected System

Add the required OData service endpoints under the connected SAP system in Cloud Connector so BTP can access iCWP services securely.

To add OData service endpoints under connected system:

1. Navigate to the **Resources of <External Virtual Host>** tab under your mapped system.
2. Click **+ Add Resource** .
3. Enter the required details: **URL Path**: For example, /sap/opu/odata/invicl/.
4. Select Path And All Sub-Paths.
5. Click on Save the access control settings.

Note:
This step ensures that only specific paths and services are exposed to SAP BTP.

6.6. Test the Connection from BTP

Verify that your Cloud Connector is correctly connected to the BTP sub-account.

To test the connection from BTP:

1. Log in to the SAP BTP Cockpit.
2. Navigate to your subaccount → **Connectivity** → **Cloud Connectors**.
3. Confirm that:
    - The Cloud Connector status shows **Connected**
    - The mapped system is visible and marked **Reachable**

7. Configure API Management in SAP BTP

Create an API Proxy in SAP BTP to expose iCWP OData services from your connected SAP system. The proxy uses the API Provider details from Cloud Connector and makes the service available for policy enforcement and external consumption.

To complete this setup, perform the following steps:

7.1. Verify Prerequisites

Verify that the foundational setup in SAP BTP is complete before enabling API Management. These prerequisites confirm that the right environment and services are available for iCWP integration.

- A subaccount created in **SAP BTP (Cloud Foundry environment)**
- **SAP Cloud Connector** already set up and connected to your subaccount
- The following services subscribed under your subaccount:
  - **API Management, API Portal**
  - **Cloud Connector** (if exposing on-premise APIs)

7.2. **Enable API Management in BTP**

Activate API Management in your SAP BTP subaccount and assign the required roles. These roles control who can configure API providers, create proxies, and manage policies for iCWP services.

Without these roles, key tasks such as creating API providers or deploying proxies cannot be performed.

Role Name	**Purpose**
Subaccount Viewer	Read-only access to the subaccount

Subaccount Service Administrator	Administrative access to service brokers and environments on a subaccount level
APIManagement.SelfService-.Administrator	API Portal Onboarding
APIPortal.Administrator	APIPortalAdmin
APIPortal.Developer	Manages API proxies and products with read-only access to all other entities.
APIPortal.Configurator	Manages entities such as API providers, certificates, and rate plans.
APIPortal.Service.CatalogIntegration	CatalogIntegration
AuthGroup.SelfService.Admin	Self Service Admin developer hub
AuthGroup.API.Admin	Manages Users and applications in SAP API Management developer hub
AuthGroup.ContentAutho	Manages Content
AuthGroup.API.ApplicationDeveloper	Application Developer in SAP API Management developer hub
AuthGroup.Content.Admin	Manages Content in SAP API management developer hub
AuthGroup.Site.Admin	Manages Site updates in SAP API management developer hub
AuthGroup.APIPortalRegistration	Role to create API Portal Connection Request

*Optional: CatalogIntegration is required only for Business Hub publishing.

7.3. Configure API Provider

Define an API Provider in SAP API Management to connect to your on-premise SAP system through Cloud Connector. The provider stores the virtual host, port, and connectivity settings that proxies use to access iCWP OData services.

To configure API provider:

1. Navigate to your BTP subaccount in the cockpit.
2. Navigate to **Instances and Subscriptions**.
3. Under **API Management, API Portal**, click **Go to Application**.
4. Navigate to **Configure**, **API Providers** in the API Portal dashboard.
5. Click **Create** and fill in the following details:
    - **Name**: Any identifier for your API provider.
    - **Description**: Provide description (This is optional).
6. Navigate to the **Connection** tab. and enter the following details:
    - **Type**: Select **On Premise**
    - Enter the **Virtual Host, Port, and Location ID** that were configured during the Cloud Connector setup, where the BTP subaccount and the on-premise system were added. Refer to the topic Configure On-Premise System Mapping *(on page 34)*
    - **Authentication** and **Additional Properties** should be **None**
7. Navigate to the **Catalog Service Settings** and enter the following:
    - **Path Prefix:** /sap/opu/odata
    - **Service Collection URL: For ECC -** /iwfnd/catalogservice/ServiceCollection **For S/4 HANA** - /IWFND/CATALOGSERVICE;v=2/ServiceCollection
    - **Authentication Type:** Basic
    - **Username:** System UserID
    - **Password:** System UserID Password
8. Click **Save**.
9. After saving the configuration successfully, follow these steps:
    a. Click **Test Connection**.
        - If the connection is successful, one of the following messages is displayed:
            - **Connection to the system was successful with response code: 200; Message: OK**.
            - **System is up and reachable. However, the ping check responded with code: 405; Message: Method Not Allowed**.
        - Both messages indicate a successful connection.
        - If neither of these messages appears, it means the configuration details are incorrect.

7.4. Create an API Proxy

Create an API Proxy in SAP BTP to expose iCWP OData services from your connected SAP system. The proxy uses the API Provider details from Cloud Connector and makes the service available for policy enforcement and external consumption.

Create an API Proxy to act as a secure, managed layer between external consumers and your backend service.

1. In the API Portal, navigate to **Develop** > **APIs**.
2. Click **Create**.
3. In the **Create API** dialog:
    a. Select **API Provider** as the source.
    b. Select the **API Provider** created in the previous step.
    c. Click the **Discover** button.
    d. Choose **OData** service name and click OK (OData: **ICWP_INTEGRATION_SUITE_SRV**).
4. Click **OK** and provide **API Name**, **Title** and **Version**.
5. Click **Create**.
6. Click **Save**.
7. Click **Deploy**.
8. Select the API proxy, click on the **three dots in the top-right corner**, and choose **Synchronize**.
9. Click **Yes** and then **Deploy**.

7.5. Secure API Proxies with Policy-Based Authentication

After creating an API proxy, you must apply authentication policies to control secure access. If no policies are configured, users are prompted for SAP credentials by default.

SAP BTP API Management supports two authentication methods for external consumers:

- **API Key**
- **OAuth 2.0**

These ensure secure access between Innovapptive Cloud and SAP BTP, while communication between **SAP BTP API Management** and the **SAP backend system (SAP Gateway)** continues to use Basic Authentication with a dedicated service **Service User ID** (system user).

To enable API Key-based authentication in SAP BTP API Management, follow the steps below:

1. Create a Key Value Map.
2. Define API Key validation policies.
3. Assign API proxies to a product and publish it.
4. Create an application and subscribe it to the product.
5. Share the API key with the external consumer.
6. Test the API in Postman using the API key.

7.5.1. Create a Key Value Map

Set up a Key Value Map (KVM) in API Management to securely store the SAP service user credentials. The values are encrypted and later referenced in authentication policies.

To create a key value map:

1. In the API Management console, click **Configure** from the left panel.
2. Click the **Key Value Maps** tab and click **Create**.
3. Click **Add** and enter these keys:
    - **UserID** <SAP Service User ID>
    - **Password** <SAP Service User Password>
4. Select **Encrypt Key Value** for both entries.
5. Click **Save**.

7.5.2. Define API Key Validation Policies

Validate incoming API calls using API Key policies in SAP BTP API Management. This ensures only authorized clients (with valid keys) can access your iCWP services.

Define and Verify Key Policy

Add a Verify API Key policy to your API Proxy. This policy checks the API key passed in the `x-api-key` header against the keys associated with registered applications in the Developer Portal. Only authorized consumers with valid keys can access the API. The policy is typically applied in the **PreFlow** of the **ProxyEndpoint**.

To define and verify key policy:

1. In the API Management console, click **Develop**.
2. Select the relevant **API Proxy**.
3. Open the **Policies** tab and click **Edit**.
4. In the **ProxyEndpoint**, select **PreFlow**.
5. From **Security Policies**, select **Verify API Key** and click **+** to add it.
6. Enter the **Policy Name**, set **Stream = Incoming Request**, and click **Add**.
7. In the policy editor, replace the default content with the following XML

   Policy Message:

```
<!--Specify in the APIKey element where to look for the variable containing the api key-->
<VerifyAPIKey async='true' continueOnError='false' enabled='true'
xmlns='http://www.sap.com/apimgmt'>
 <APIKey ref='request.header.x-api-key'/>
</VerifyAPIKey>
```

8. Click **Update**.
9. Click **Save**, then **Redeploy** the API Proxy so the policy takes effect.

Define the Key Value Map Operations Policy

Add a Key Value Map (KVM) Operations policy to your API Proxy. This policy retrieves stored credentials (for example, the SAP service user ID and password) from the Key Value Map and makes them available for downstream policies.

1. In the API Management console, select the relevant **API Proxy**.
2. Open the **Policies** tab and click **Edit**.
3. In the **ProxyEndpoint**, select **PreFlow**.
4. From **Mediation Policies**, select **Key Value Map Operations** and click **+**.
5. Enter the **Policy Name**, set **Stream = Incoming Request**, and click **Add**.
6. In the policy editor, replace the default content with the following XML:**Policy Message:**

```
<!-- Key/value pairs can be stored, retrieved, and deleted from named existing maps by configuring this policy
 by specifying PUT, GET, or DELETE operations -->
<!-- mapIdentifier refers to the name of the key value map -->
<KeyValueMapOperations mapIdentifier="iCWPCredentials" async="true" continueOnError="false" enabled="true"
 xmlns="http://www.sap.com/apimgmt">
 <!-- PUT stores the key value pair mentioned inside the element -->
 <Get assignTo="private.username">
    <Key>
      <Parameter>UserID</Parameter>
    </Key>
  </Get>
  <Get assignTo="private.password">
  <Key>
      <Parameter>Password</Parameter>
  </Key>
  </Get>
 <!-- the scope of the key value map. Valid values are environment, organization, apiproxy and policy -->
 <Scope>environment</Scope>
</KeyValueMapOperations>
```

7. Click **Update**.
8. Click **Save**, then **Redeploy** the API Proxy so the policy is active.

Define the Basic Authentication Policy

Add a Basic Authentication policy to your API Proxy. This policy encodes the SAP service user credentials (retrieved from the Key Value Map) into the `Authorization` header, enabling secure downstream calls from API Management to the SAP backend.

1. In the API Management console, select the relevant **API Proxy**.
2. Open the **Policies** tab and click **Edit**.
3. In the **ProxyEndpoint**, select **PreFlow**.
4. From **Security Policies**, select **Basic Authentication** and click **+**.
5. Enter the **Policy Name**, set **Stream = Incoming Request**, and click **Add**.
6. In the policy editor, replace the default content with the following XML:

Policy Message

```
<BasicAuthentication async='true' continueOnError='false' enabled='true' xmlns='http://www.sap.com/apimgmt'>
  <!-- Operation can be Encode or Decode -->
 <Operation>Encode</Operation>
 <IgnoreUnresolvedVariables>true</IgnoreUnresolvedVariables>
  <!-- for Encode, User element can be used to dynamically populate the user value -->
 <User ref='private.username' />
  <!-- for Encode, Password element can be used to dynamically populate the password value -->
 <Password ref='private.password' />
  <!-- Source is used to retrieve the encoded value of username and password. This should not be used if the
 operation is Encode-->
  <Source>request.header.Authorization</Source>
  <!-- Assign to is used to assign the encoded value of username and password to a variable. This should not be
 used if the operation is Decode -->
 <AssignTo createNew="false">request.header.Authorization</AssignTo>
</BasicAuthentication>
```

7. Click **Update**.
8. Click **Save**, then **Redeploy** the API Proxy to apply the changes.

7.5.3. Create a Product and Bundle the API

Create a product in API Portal and bundle the iCWP API so it's available in Developer Hub.

To create a product and bundle the API:

1. Open the **API Portal** and select **Develop** on the left-hand panel.
2. Click on the **Products** tab and click **Create** to start creating a new product.
3. Enter the Product Name as **<iCWPIntegrationSuite>** and the Title as **<iCWP Integration Suite>**.
4. Click the **APIs** tab and click **Add** to view the list of deployed APIs.
5. Select the API **ICWP_INTEGRATION_SUITE_SRV** and click **OK.**

6. Click **Publish** to make the product available in the **Developer Hub (API Business Hub Enterprise)**.
7. Click on More icon on top-right corner.

   You will see two options: **Client SDK** and **Developer Hub**. Click on **Developer Hub** to view the published products.

   The **Developer Hub portal** opens with the **Published Products** list.

7.5.4. Create an Application to Generate Application Key and Secret Key

Create an application in Developer Hub, subscribe it to the published product, and generate the **Application Key** (and optionally a **Secret Key**) used to authenticate API calls.

To create an application to generate application key and secret key:

1. In Developer Hub, select the **published product**.
2. Click the **Subscribe** drop-down and select **Create New Subscription for Application**.
3. Enter **Application Name** = `<iCWPIntegrationSuite>` and **Short Text** = `<iCWP Integration Suite>`

7.5.5. Test API Proxy with API Key Authentication

Call the API Proxy using the generated Application Key to validate API Key–based access..

To test API proxy with API key authentication:

1. Open **API Portal**.
2. Click **Develop** on left-side panel and copy or note the **API Proxy URL** on the right-side.
3. Click the More icon on top-right and select **Developer Hub**.
4. Copy or note the **Application Key**.
5. Open **Postman**.
6. In Postman, set **Auth Type** to **API Key**.
   - Key: **x-api-key**
   - Value: **<Application Key>** (copied from the Developer Hub)
7. Set the **Request URL** to the **API Proxy URL** (copied from the API Portal).

   Case 1: Passing correct API key to the Authentication

   Case: 2 Passing Incorrect API key to the Authentication

8. Configure OAuth 2.0 Authentication

Enable OAuth 2.0 authentication in SAP API Management so external consumers can securely obtain and use access tokens when calling iCWP APIs.

To enable OAuth authentication for an API Proxy, configure the following activities:

- **Create an OAuth Token API Proxy** – define the token endpoint using an OAuth v2.0 policy in *GenerateAccessToken* mode.
- **Create the actual API Proxy** – secure it with a *VerifyOAuthToken* policy and configure backend authentication.
- **Publish the API as a Product and create an Application** – generate the Client ID and Client Secret needed for token requests.
- **Test the OAuth flow** – generate a token and invoke the iCWP API with it.

8.1. Create OAuth Token API Proxy

Set up an OAuth Token API Proxy in SAP API Management. This proxy acts as the token endpoint, issuing access tokens that external consumers use to call protected iCWP APIs.

- Create an API Proxy in API Management (use a dummy URL as backend).
- Add OAuth v2.0 Policy in GenerateAccessToken mode.
- Deploy the proxy – this acts as the token endpoint.
- Add OAuth Token Generation API Proxy to the Product and Republishing
- Test OAuth Token Generation using the API proxy.

8.1.1. Create API proxy in API management

Create a new API Proxy in API Management with a dummy backend. This proxy serves as the OAuth token endpoint used to generate access tokens for iCWP APIs.

To create an API proxy in API management:

1. In the **API Portal**, navigate to **Develop** > **APIs**.
2. Click **Create**.
3. In the **Create API** dialog:
    a. Select **URL** as the source.
    b. Enter Working Dummy URL: **<** https://www.innovapptive.com/ **>** (it can be any working URL).
    c. Enter **Name : ICWPGenerateoAuthToken** and **Title: ICWP Integration Suite Generate oAuth Token** for the **API Proxy**.
    d. Provide **API Base Path**:**/token**.
    e. Click **Create** to generate the proxy.
4. Go to **Proxy Endpoint** → **Route Rules:**
    a. Route Rule Name: **Default**.
    b. Target Endpoint: **None**.
    c. Click **Save**.

8.1.2. Add OAuth v2.0 Policy to Generate Access Tokens

Attach an OAuth v2.0 policy to the API Proxy in *GenerateAccessToken* mode. This policy defines the token endpoint, issues OAuth access tokens using the `client_credentials` grant type, and returns them in the response for use by external consumers.

1. Select **API proxy** and click **Policies**.
2. Click **Edit**.
3. Select **ProxyEndpoint**.
4. Select **PreFlow**.
5. Under **Security Policies**, choose **OAuth v2.0**.
6. Click on the **+** button to add the policy.
7. Enter the **Policy Name**, set the **Stream** as **Incoming Request** and click **Add**.
8. Replace the existing default content in the Body section with the following policy XML and click **Update**.

    Policy Message:

```
<OAuthV2 async="false" continueOnError="false" enabled="true" xmlns="http://www.sap.com/apimgmt">
   <!-- By default, VerifyAccessToken expects the access token to be sent in an Authorization header. You can
 change that default using this element<AccessToken> -->
   <!-- If you want to pass access token in an customer header "access_token": -->
```

```
   <!-- <AccessToken>request.header.access_token</AccessToken> -->
   <!-- If you want to pass access token in query param "access_token": -->
   <!-- <AccessToken>request.queryparam.access_token</AccessToken> -->
   <!-- this flag has to be set when you want to work with third-party access tokens -->
   <ExternalAuthorization>false</ExternalAuthorization>
   <!-- valid values are GenerateAccessToken, GenerateAccessTokenImplicitGrant, GenerateAuthorizationCode ,
    RefreshAccessToken , VerifyAccessToken , InvalidateToken , ValidateToken  -->
   <Operation>GenerateAccessToken</Operation>
   <GenerateResponse enabled="true"/><SupportedGrantTypes>
       <GrantType>client_credentials</GrantType>
       </SupportedGrantTypes>
   <Tokens/>
</OAuthV2>
```

9. Click **Save**.
10. In the **API Portal**, navigate to **Develop**.
11. Click **Click to Deploy** to apply the changes.

8.1.3. Add OAuth Token Proxy to Product and Republish

Update the existing iCWP product to include the OAuth token generation proxy. Once added, republish the product so both the actual API proxy and the token proxy are available in Developer Hub.

To add OAuth token generation API proxy to product:

1. In the **API Portal**, navigate to **Develop** → **Products** tab,
2. Click on the already created product **iCWPIntegrationSuite**.
3. Go to the **APIs** tab and click **Edit**.
4. Click **Add**.
5. In the API list, select the OAuth token generation API proxy: **ICWPGenerateoAuthToken** and click **OK**.
6. Click **Publish**.

8.1.4. Test OAuth Token Generation using API Proxy

Use Postman to request an OAuth 2.0 token from the Generate OAuth Token API Proxy.
This validates that the proxy issues access tokens using the client ID and secret from the Developer Hub.

To test OAuth token generation using an API proxy:

1. Open **Postman** and go to the **Authorization** tab.
2. Set **Auth Type** to **OAuth 2.0**.
3. Fill in the following details:
    a. **Token Name**: any descriptive name
    b. **Grant Type**: *Client Credentials*
    c. **Access Token URL**: URL of the **Generate OAuth Token API Proxy**
    d. **Client ID**: Application Key from **Developer Hub**
    e. **Client Secret**: Secret Key from **Developer Hub**
4. Scroll down and click **Get New Access Token**.

8.2. Add OAuth v2.0 Policy to Verify Access Tokens

Attach an OAuth v2.0 policy in *VerifyAccessToken* mode to the actual API Proxy. This ensures only requests with a valid token (issued by the token proxy) can access iCWP APIs.

To add OAuth v2.0 policy:

1. Select **API proxy** and click **Policies**.
2. Click **Edit**.
3. Select **ProxyEndpoint**.
4. Select **PreFlow**.
5. Under **Security Policies**, choose **OAuth v2.0**.
6. Click on the **+** button to add the policy.
7. Enter the **Policy Name**, set the **Stream** as **Incoming Request** and click **Add**.
8. Replace the existing default content in the Body section with the following policy XML and click **Update**.

    Policy Message:

```
<OAuthV2 async="false" continueOnError="false" enabled="true" xmlns="http://www.sap.com/apimgmt">
   <!-- By default, VerifyAccessToken expects the access token to be sent in an Authorization header. You can
 change that default using this element<AccessToken> -->
   <!-- If you want to pass access token in an customer header "access_token": -->
   <!-- <AccessToken>request.header.access_token</AccessToken> -->
```

```
    <!-- If you want to pass access token in query param "access_token": -->
    <!-- <AccessToken>request.queryparam.access_token</AccessToken> -->
    <!-- this flag has to be set when you want to work with third-party access tokens -->
    <ExternalAuthorization>false</ExternalAuthorization>
    <!-- valid values are GenerateAccessToken, GenerateAccessTokenImplicitGrant, GenerateAuthorizationCode ,
     RefreshAccessToken , VerifyAccessToken , InvalidateToken , ValidateToken  -->
    <Operation>VerifyAccessToken</Operation>
    <GenerateResponse enabled="true"/><SupportedGrantTypes/>
    <Tokens/>
</OAuthV2>
```

9. Click **Save**.

   The **Verify OAuth Access Token** policy should be placed **first in the execution flow**. You can change its position by clicking on the **navigation arrows.**

8.3. Apply OAuth 2.0 Policy Between External Consumer and BTP API Management

Enable OAuth 2.0 authentication at the SAP BTP API Management layer to secure external access. Internally, API Management continues to call the SAP backend using Basic Authentication stored in Key Value Maps. To support this hybrid model, add a policy that removes the OAuth token from the request context after verification, ensuring clean forwarding to the backend.

To add policy to remove OAuth token:

1. Select **API proxy** and click **Policies**.
2. Click **Edit**.
3. Select **ProxyEndpoint**.
4. Select **PreFlow**.
5. Under **Mediation Policies**, choose **Assign Message**.
6. Click on the **+** button to add the policy.
7. Enter the **Policy Name**, set the **Stream** as **Incoming Request** and click **Add**.
8. Replace the existing default content in the Body section with the following policy XML, and click **Update**.

   Policy Message:

```
<!-- This policy can be used to create or modify the standard HTTP request and response messages -->
<AssignMessage async="false" continueOnError="true" enabled="true" xmlns='http://www.sap.com/apimgmt'>
    <Remove>
      <Headers>
<Header name="Authorization"></Header>
          </Headers>
     </Remove>
<IgnoreUnresolvedVariables>true</IgnoreUnresolvedVariables>
<AssignTo createNew="false" type="request"></AssignTo>
</AssignMessage>
```

9. Click **Save**.

8.4. Test OAuth using Postman

To test OAuth authentication using Postman:

1. Open Postman and set the Authorization Type to **Bearer Token**.
2. Provide the OAuth token (you can obtain this token from the previous steps explained in the document).
3. If the token is valid, you will receive a successful response with status code **200 OK**.
4. If the token is invalid, the response code will be **401 Unauthorized**.
5. If the token is expired, the response code will be **401 Unauthorized**.

9. Handle CSRF Tokens in API Management

Enable server-to-server authentication between SAP API Management and the backend system, so API consumers don't need direct backend credentials.

For `POST`, `PUT`, and `DELETE` requests, configure a ServiceCallout `scCSRF` policy to automatically fetch and pass the required X-CSRF token from the backend.

To handle CSRF token in API management:

1. Select **API proxy** and click **Policies**.
2. Click **Edit**.
3. Select **ProxyEndpoint**.
4. Select **PreFlow**.
5. Under **Extension Policies**, choose **Service Callout**.
6. Click on the **+** button to add the policy.
7. Enter the **Policy Name**, set the Stream as Incoming Request and click **Add**.
8. Replace the existing default content in the Body section with the following policy XML and click **Update**.

   Condition String:

   ```
   (request.verb = "POST" OR request.verb = "PUT" OR request.verb = "DELETE")
   ```

   Policy Message:

   ```
   <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
   <ServiceCallout async="true" continueOnError="false" enabled="true" xmlns="http://www.sap.com/apimgmt">
       <Request>
     <Set>
       <Headers>
         <Header name="x-csrf-token">fetch</Header>
         <Header name="Authorization">{request.header.Authorization}</Header>
       </Headers>
       <Verb>GET</Verb>
     </Set>
     <IgnoreUnresolvedVariables>false</IgnoreUnresolvedVariables>
   </Request>
       <Response>calloutResponse</Response>
       <Timeout>30000</Timeout>
   ```

```
<HTTPTargetConnection>
     <APIProvider>EW8API</APIProvider>
       <Path>/sap/opu/odata/invicl/ICWP_INTEGRATION_SUITE_SRV</Path>
</HTTPTargetConnection>
</ServiceCallout>
```

Note:

Make sure to change the API Provider name in the policy, for example:

```
<HTTPTargetConnection>
   <APIProvider>[Provide API Provider Name here ]</APIProvider>
   <Path>/sap/opu/odata/invicl/ICWP_INTEGRATION_SUITE_SRV</Path>
</HTTPTargetConnection>
```

9. Click **Save**.
10. Enter the **Policy Name**, set the Stream as Incoming Request, and click **Add**.
11. Replace the existing default content in the Body section with the following policy XML, and click **Update**.

Condition String:

```
(request.verb = "POST" OR request.verb = "PUT" OR request.verb = "DELETE")
```

Policy Message:

```
<!-- This policy can be used to create or modify the standard HTTP request and response messages -->
<AssignMessage async="false" continueOnError="false" enabled="true" xmlns='http://www.sap.com/apimgmt'>
<!-- Sets a new value to the existing parameter -->
<Set>
   <Headers>
      <Header name="x-csrf-token">{callOutResponse.header.x-csrf-token}</Header>
      <Header
 name="Cookie">{callOutResponse.header.Set-Cookie.1};{callOutResponse.header.Set-Cookie.2};{callOutResponse.hea
der.Set-Cookie.3}</Header>
    </Headers>
</Set>
```

```
<IgnoreUnresolvedVariables>false</IgnoreUnresolvedVariables>

<AssignTo createNew="false" type="request">request</AssignTo>

</AssignMessage>
```

Note:
Some customers may receive only two cookies instead of three. In such cases, update the policy accordingly.

12. Click **Save**.

10. Troubleshoot API Errors and Metadata Issues

This chapter outlines common issues encountered during API consumption via SAP API Management, especially when integrating with SAP backend systems. It also explains how to apply policy-based workarounds and handle metadata changes in OData services.

10.1. Metadata Does Not Populate in S/4 HANA Systems

In SAP HANA-based systems, you may encounter issues where OData metadata does not populate when the API is invoked.

Cause

This error occurs when the backend system compresses metadata responses using the Accept-Encoding header. Some versions of HANA do not handle decompression properly, causing the metadata fetch to fail.

Resolution

Use the **Assign Message** policy in the API Proxy to override the Accept-Encoding header.

1. Select **API proxy** and click **Policies**.
2. Click **Edit**.
3. Select **ProxyEndpoint**.
4. Select **PreFlow**.
5. Under **Mediation Policies**, choose **Assign Message**.
6. Click on the **+** button to add the policy.
7. Enter the **Policy Name**, set the Stream as Incoming Request and click **Add**.
8. Replace the existing default content in the Body section with the following policy XML, and click **Update**.
9. **Policy Message:**

```
<!-- This policy can be used to create or modify the standard HTTP request and response messages -->
<AssignMessage async="false" continueOnError="false" enabled="true" xmlns='http://www.sap.com/apimgmt'>


 <!-- Sets a new value to the existing parameter -->
 <Set>
  <Headers>
       <Header name="Accept-Encoding">gzip,deflate</Header>
```

```
      </Headers>
 </Set>
 <IgnoreUnresolvedVariables>false</IgnoreUnresolvedVariables>
 <AssignTo createNew="false" type="request"></AssignTo>
</AssignMessage>
```

10. Click **Save**.
11. Now, the policies have been successfully defined. Proceed to deploy the API proxy to apply the changes.