# PROCESS MINING INSTALLATION COOKBOOK

ARIS Process Mining 10

October 2026

This document applies to ARIS Process Mining Version 10 and to all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © 2020-2026 SAG Aris GmbH, Saarbrücken, Germany and/or its subsidiaries and/or its affiliates and/or their licensors.

The name ARIS and all SAG Aris GmbH product names are either trademarks or registered trademarks of SAG Aris GmbH and/or its subsidiaries and/or its affiliates and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners.

The Software may contain or be distributed with certain commercial third party components (“Third Party Components”), which may be subject to special license terms and conditions (“Third Party Terms”) located at https://aris.com/terms.

Third Party Terms include important licensing and warranty information and disclaimers of third party licensors. For the avoidance of doubt, if Customer uses the Software in accordance with the Documentation, the Third Party Terms do not restrict the rights granted under this Agreement. In the event that a third party manufacturer no longer provides active support for any Third Party Component, Supplier will use commercially reasonable efforts to respond to any issues with the Software related to the Third Party Components.

Use, reproduction, transfer, publication or disclosure is prohibited except as specifically provided for in your License Agreement with SAG Aris GmbH and/or its subsidiaries and/or its affiliates and/or their licensors.

# Table of Contents

1. Overview ..... 4
2. Additional documentation referenced in this document ..... 5
3. Requirements ..... 6
3.1. Installation packages ..... 6
3.2. Hardware and infrastructure ..... 6
3.2.1. Server machines ..... 6
3.2.2. Relational database server ..... 7
3.2.3. External load balancer ..... 7
3.2.4. Network shares ..... 8
4. Sizing a combined ARIS installation ..... 9
5. Preparing one or more servers for use as worker nodes ..... 10
5.1. Additional steps after installing the ARIS Cloud Agent on Windows servers ..... 10
5.2. Additional steps after installing the ARIS Cloud Agent on Linux servers ..... 12
5.2.1. Mounting the network share on Linux environments ..... 12
6. Providing and filling a runnable repository server ..... 13
7. Preparing ARIS Cloud Controller ..... 14
8. Overview of the additional runnables ..... 15
9. Example scenario ..... 17
9.1. Prerequisites ..... 17
9.2. Overview of the example scenario (case-centric) ..... 17
9.3. Overview of the example scenario (object-centric) ..... 18
9.4. Deploying the ARIS and Process Mining runnables ..... 19
9.4.1. Configuring the Zookeeper cluster ..... 19
9.4.2. Configuring the Elasticsearch cluster ..... 20
9.4.3. Configuring the Cloudsearch cluster ..... 20
9.4.4. Configuring the ARIS microservices ..... 21
9.4.5. Configuring the mining-specific microservices (case-centric) ..... 22
9.4.6. Configuring the mining-specific microservices (object-centric) ..... 23
9.4.7. Adding additional services to enable the object-centric process mining feature ..... 23
9.5. Additional configuration steps ..... 25
9.5.1. Configuring the external DBMS and adding the JDBC drivers ..... 25
9.5.2. Registering an external mail server ..... 26
10. Patching or updating a combined ARIS installation ..... 27
11. Tenant management ..... 30
11.1. Backup modes ..... 30
11.2. Classical tenant backup and restore with the TM CLI ..... 31
11.3. Direct-2-network share backup and restore with the TM CLI ..... 31
11.3.1. D2S backup and restore via the UI ..... 32
11.3.2. D2S backup and restore via the TM CLI ..... 33
11.3.3. Incremental D2S backups ..... 34
12. Support and legal information ..... 38
12.1. Supported Web browsers ..... 38
12.2. Documentation scope ..... 38
12.3. Restrictions ..... 38
12.4. Support ..... 39
12.5. Legal notices ..... 39

# 1. Overview

This document outlines the requirements and procedure to install ARIS with ARIS Process Mining on-premise (short: Process Mining). The document assumes general familiarity with ARIS installation terminology and technology (in particular ARIS Cloud Controller CLI). (Chapters **ARIS Architecture** and **ARIS provisioning & setup** of [1] [5] give you a quick overview if needed.)

For the sake of simplicity, the installation of ARIS with ARIS Process Mining on-premise is referred to as a **combined ARIS installation** in the following.

# 2. Additional documentation referenced in this document

In addition to the steps outlined in this document, we refer to the following documents, indicating the list numbers:

- [1] ARIS Distributed Installations Cookbook
- [2] ARIS System Requirements
- [3] ARIS Server Installation - Windows
- [4] ARIS Server Installation - Linux
- [5] ARIS Update Cookbook
- [6] Tenant Management Online Help
- [7] ARIS Server Update Installation Guide

You find the documents on the ARIS Documentation website.

# 3. Requirements

## 3.1. Installation packages

To install ARIS with ARIS Process Mining on-premise, two installation packages are required:

- the regular ARIS DVD (containing the agent setups, the regular ARIS server and ARIS client setups, documentation, and the ARISrunnables)

  **Note**

  Do not use the initial release version of ARIS Process Mining on-premise to install ARIS or to update an existing ARIS installation. Such installations do not receive any kind of support and might not be updateable to newer versions. This version can only be used for fresh installations of ARIS and ARIS Process Mining on-promise.
- a new ARIS Process Mining image, containing the additional ARIS Process Mining runnables and the ARIS Process Mining-specific **generated.apptypes.mining.cfg** file.

## 3.2. Hardware and infrastructure

This section gives an overview of the required hardware and IT infrastructure. This includes the following:

- multiple server machines for hosting the ARIS server components.
- a relational database system, which should offer high-availability capabilities, if high-availability of the ARIS installation is required.

  While ARIS ships with a relational database system runnable (based on PostgreSQL), the expected scope of a combined ARIS installation the shipped database is beyond what this can handle. Therefore, using an external DBMS is mandatory.
- an external, HTTP(S) aware loadbalancer, which should offer high-availability capabilities, if high-availability of the ARIS installation is required.
- one or more network shares, which should offer high-availability and/or have a production-ready backup solution, as at least some of these shares will hold actual application data.

### 3.2.1. Server machines

To install ARIS Process Mining, you need multiple server machines. The exact number of servers and their system requirements (in particular number of CPU cores, available RAM, free disk space) depends on various factors, discussed in the “Sizing” section below.

The server machines will host the ARIS and Process Mining “runnables” (i.e., the different independent server-side components that all together form an ARIS installation).

**Warning**
Note that for purely functional tests with small number of users and limited amounts of mining source data, it is possible to work with a single worker node holding one instance of each type of runnable.

Under the same considerations, it is also possible to do without an external DBMS. In that case, the step of registering an external DBMS discussed later can be skipped, replacing it with configuring the PostgreSQL runnable shipped with ARIS.

This system must not be used in any production environment under any circumstances.

The worker nodes can be either Windows- or Linux-based (but no mix of Linux and Windows machines).

For details on the currently supported Windows versions resp. Linux distributions, refer to the “ARIS System requirements” document [2]).

### 3.2.2. Relational database server

In addition to the server hardware hosting the ARIS runnables, an relational database server is required.

**Note**

Note that the DBMS is used as a backend only for some types of data held by the ARIS installation (for example, ARIS models, document storage metadata, comments and feeds, state of governance processes). Other types of data managed by ARIS (users, groups, document indexes, ARIS Process Mining configuration and dashboards, mined process data, etc.) is held in other backends (Elasticsearch or network shares, see below).

In particular, this means that in order to backup all your ARIS installation’s data, it is not sufficient to do a backup of the DBMS. ARIS offers a backup and restore mechanism that extracts all data of an ARIS tenant, regardless of the backend, which holds the data at runtime. See the section Tenant management in ARIS with Process Mining [30] below for an overview about ARIS multi-tenancy and the backup and restore functionality.

For supported DBMS types, see the **ARIS System Requirements** document. If the combined ARIS installation should offer high-availability, the DBMS needs to be highly available.

### 3.2.3. External load balancer

If the combined ARIS installation should offer high-availability, an external highly available layer-7 loadbalancer is required. (Refer to sections **Load balancing and high availability** to **The X-Forwarded-* headers** of [1] [5] for a motivation and details.) In the examples below, we assume that the external loadbalancer is reachable via a DNS entry (for example, “myaris.mycompany.com”) and that the loadbalancer is configured to forward the traffic to the ARIS loadbalancer runnables.

Note that the external loadbalancer needs to pass the original client IP address with the X-Forwarded-For header to the ARIS loadbalancer. Further, the protocol (http or https), hostname, and the port through which users access the external loadbalancer needs to be passed to the ARIS loadbalancers with the X-Forwarded-Proto, X-Forwarded-Host and X-Forwarded-Port headers, respectively.

### 3.2.4. Network shares

Finally, for data storage and for data exchange between the ARIS runnables, network file shares are used. The following different kinds of data that reside on (or are exchanged via) file shares exist. We refer to them as logical shares:

- document storage binary data (referred to as the **adsdata** (logical) share):
  This is the same as in regular ARIS installations without Process Mining. (Details on how to use this share can be found in the chapter **Configuring document storage for high availability** of [1] [5].)
- tenant backups (referred to as the **tmdata** (logical) share). This share holds tenant backups that are done through the tenant management application.
- persistent Process Mining source data (referred to as the **persistence** (logical) share) - as the naming suggests, this share holds the source data on which Process Mining is performed. This data is kept in the system to allow recalculation of the mined data.
- Process Mining data exchange share (referred to as the **filestorage** (logical) share) - used for data exchange between the mining components.
- Process Mining OCPM process data share (referred to as the **dbfiles** (logical) share) - used to store the OCPM process data.

We refer to these five types of “data on a share” as logical network shares.

**Note**

Note that all these five logical shares can be arbitrarily put on one or multiple “physical” network shares (assuming it meets available disk space and I/O performance requirements). However, when you use the same “physical” network share for more than one “logical” share, it is mandatory to have dedicated subdirectories for each logical share.

As a running example, we assume that we use two physical network shares: The first should contain the **tmdata** and **adsdata** logical shares. Assume we mounted this on drive letter **S**. A second physical network share mounted to drive letter **T** holds the three remaining logical shares **persistence**, **file-storage**, and dbfiles. In this case, we create two/three subdirectories in each physical share, for example, S:\adsdata and S:\tmdata and T:\persistence, T:\file-storage, and T:\dbfiles. Whenever we configure network share paths, we need to take care to use the full path including these subdirectories.

# 4. Sizing a combined ARIS installation

The number of servers or “worker nodes” needed for the combined ARIS installation and the performance and capacity of the external services (in particular the external DBMS and the network shares) depend on various factors, including but not limited to:

- the overall requirements for high-availability of the ARIS installation (that is, tolerance to temporary outages of ARIS components or entire server machines)
- for the ARIS components, the mount of various types of ARIS content, for example,
  - number of users and user groups.
  - number of ARIS modeling databases, the number of models, objects, and versions in each of these.
  - the number and total size of documents stored in ARIS document storage.
  - the number of ARIS Process Governance workflows active at any given time.
  - the number and kind of users active in parallel and the type of work they perform on the system.
  - the amount and type of ARIS customizing.
  - the number and complexity of ARIS reports and how many of them are running in a given timeframe.
- for the ARIS Process Mining components, the main factors influencing their number and vertical sizing are:
  - the number and size of ARIS Process Mining data sets (that is, the number of distinct sets of source data on which Process Mining is to be performed) and the number of activities and cases (case-centric) or respectively the number of objects and events (object-centric), and the number of attributes.
  - the number of parallel **Viewer** users (= number of people viewing Process Mining analyses and mining results).
  - the number of parallel **Analyst** users (= number of people preparing Process Mining tasks, building analyses etc.).

All these factors influence the number (= horizontal scaling), available memory, and CPU cores (= vertical scaling) of the ARIS components (the “runnables”), which in turn influence the number and “sizing” (= available CPU cores, main memory, available disk space, I/O performance) of the server machines, performance requirements, available space on the external DBMS, and the file servers providing the network shares.

Due to the hugely different usage scenarios of ARIS and ARIS Process Mining, there can obviously be no simple guideline on how to size such a deployment. Thus, this document cannot reasonably provide “ready-to-use” sizing recommendations. What the document provides, is an example scenario that is only meant to explain the basic steps needed to perform a deployment. It can also serve as a starting point for first functional testing. It must however not be used in any kind of production role. For this, a custom sizing of the environment fitting to your requirements in the various categories listed above must be obtained through the ARIS support. Even this sizing should however be considered only as a starting point that might need to be modified based on actual experience while working with the system. Here the benefits of the microservice architecture make it easy to only scale (vertically or horizontally or both (those components that turned out to be bottlenecks).

# 5. Preparing one or more servers for use as worker nodes

Please make sure that you did all the preparation steps described in chapter **Preparation steps** of [1] [5]. This includes

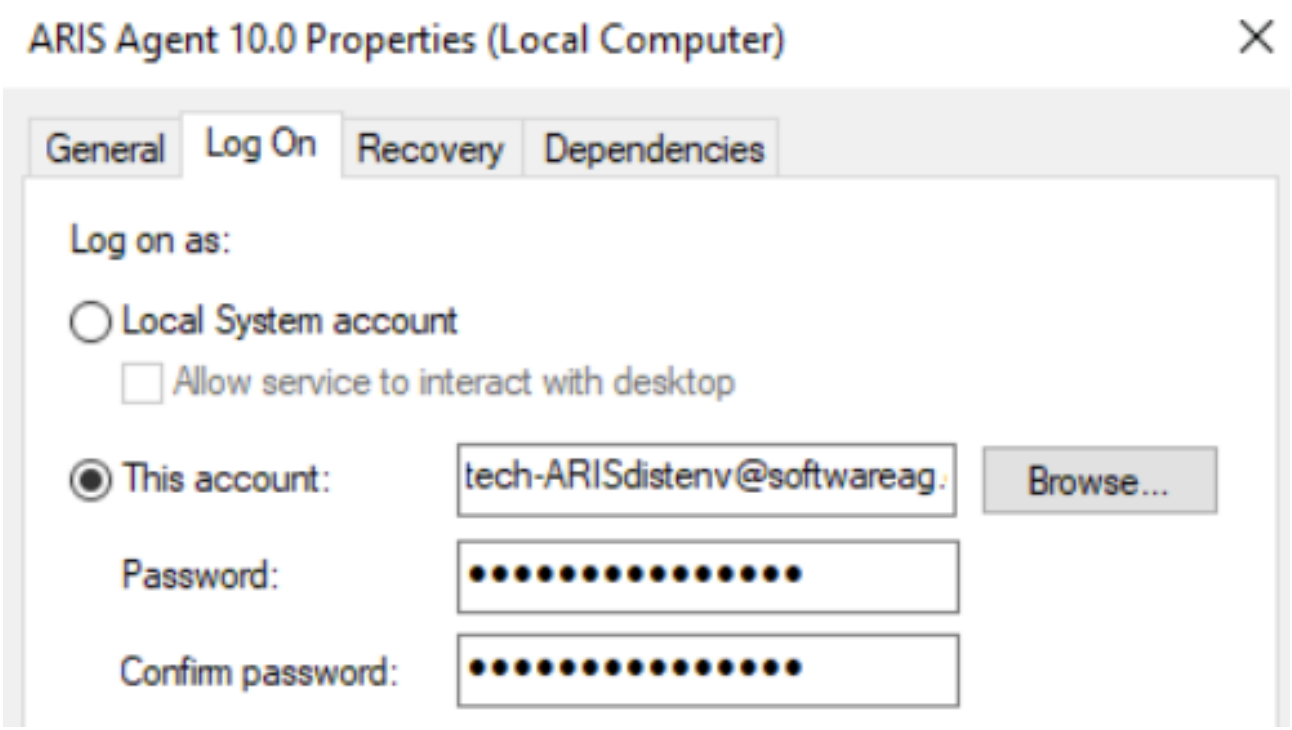
- installing the ARIS Cloud Agent on all machines that are intended to be used for ARIS (see the chapter **Install the ARIS Agent on all nods** of [1] [5] and the operating-specific installation instructions of [3] [5] and [4] [5]).

  In the remainder of this document, we refer to the machines that have the ARIS Cloud Agent installed (and are meant to become part of one single distributed ARIS installation) as “(ARIS) worker nodes”. The entirety of ARIS worker nodes are part of a single distributed ARIS installation is referred to as an “ARIS cluster”.

  In addition to the general ARIS Cloud Agent setup instructions discussed in the referenced documentation, further preparations must be done for ARIS Process Mining, which are discussed in dedicated sections for Windows and Linux servers below.
- preparing a node file and using the node file with ARIS Cloud Controller to connect to all nodes of your future ARIS cluster at once (see sections **Controlling multiple nodes with a single ACC instance** to **Using the node file** of [1] [5]).

## 5.1. Additional steps after installing the ARIS Cloud Agent on Windows servers

To mount the network shares required for ARIS Process Mining on a **Windows** system, it is necessary to run the ARIS Cloud Agent with a dedicated user that has access to the network shares. By default, the ARIS Cloud Agent and all runnables run with the **system** user, which normally cannot access remote resources. After installing the agent with the Windows installer, please locate the Windows service "ARIS Agent 10.0" (or similar) and open its properties dialog. Go to the **Log On** tab and switch from **Local System account** to **This account**. Then, enter the account name and credentials of the user who has access to the network shares, as shown in the screenshot below.




Now with the agent running under the user that has access to the shares, we need to mount them to drive letters or local paths for that user. Due to technical limitations of some of the mining components, the network shares have to be mounted to a drive letter and cannot be used directly via their UNC name. That can be done by listing the shares and directory mappings in the custom configuration file of the ARIS Window service, located at

**<installDir>\server\conf\ARISCloudAgentapp.user.conf**

For each physical network share that you plan to use with ARIS, you need to add three lines that follow this general structure

```
wrapper.share.<ShareNumber>.location=\\<SERVERNAME>\<SHARENAME>
wrapper.share.<ShareNumber>.target=<DRIVELETTER>:
wrapper.share.<ShareNumber>.type=DISK
```

where <ShareNumber> is just a running number starting at 1, <SERVERNAME> is the host name of the file share server, and <SHARENAME> is the name of the share.

For example if you have two physical network shares with UNC paths **\\serverA\shareA** and **\\serverB\shareB** that you want to map to drive letters S: and T:, you would configure them like this in the **ARISCloudAgentapp.user.conf** file:

```
wrapper.share.1.location=\\serverA\shareA
wrapper.share.1.target=S:
wrapper.share.1.type=DISK
wrapper.share.2.location=\\serverB\shareB
wrapper.share.2.target=T:
wrapper.share.2.type=DISK
```

## 5.2. Additional steps after installing the ARIS Cloud Agent on Linux servers

On a **Linux** system, the ARIS Cloud Agent and thus all runnables run by default with the **aris10** user. However, when the agents on different nodes, the **aris10** user might different user and group IDs (UID and GID, respectively) on each of them. After installing the agent on all your future ARIS Linux worker nodes, check the **aris10** user's UID and GID by running the command:

```
id aris10
```

on each of them. You will see an output like this

```
uid=1003(aris10) gid=1003(aris10) groups=1003(aris10),10(wheel)
```

Make sure that the numerical UID and GID for the **aris10** user and group are the same on all nodes. If the UID or GID are different, on some nodes, you can change them by replacing the IDs in the **/etc/passwd** and **/etc/groups** file respectively. Make sure that you do not use a UID or GID, respectively, for the **aris10** user that is already in use by a different user or group, respectively.

### 5.2.1. Mounting the network share on Linux environments

To mount the different physical network shares on each node, you need to add a line for each network share to each node's **/etc/fstab** file. The parameters vary depending on the type and technical details of the network share. The following examples are therefore only intended as a general guideline. Contact your network file server administrator to receive the parameters for your specific infrastructure and environment.

#### Example 1.

This example shows structure of a **fstab** line for mounting an **SMB** share:

```
//<SERVERNAME>/<SHARENAME> <MOUNT_POINT> cifs vers=3.0,uid=aris10,credentials=/
localfs/nfs-root/.smbcredentials,dir_mode=0777,file_mode=0777 0 0
```

#### Example 2.

//smbserver.example.com/miningSrcData /mnt/ cifs vers=3.0,uid=aris10,credentials=/localfs/nfs-root/.smbcredentials,dir_mode=0777,file_mode=0777 0 0

Structure of a **fstab** line for mounting an **NFS** share:

<SERVERNAME>:<PATH> <MOUNT_POINT> nfs defaults 0 0

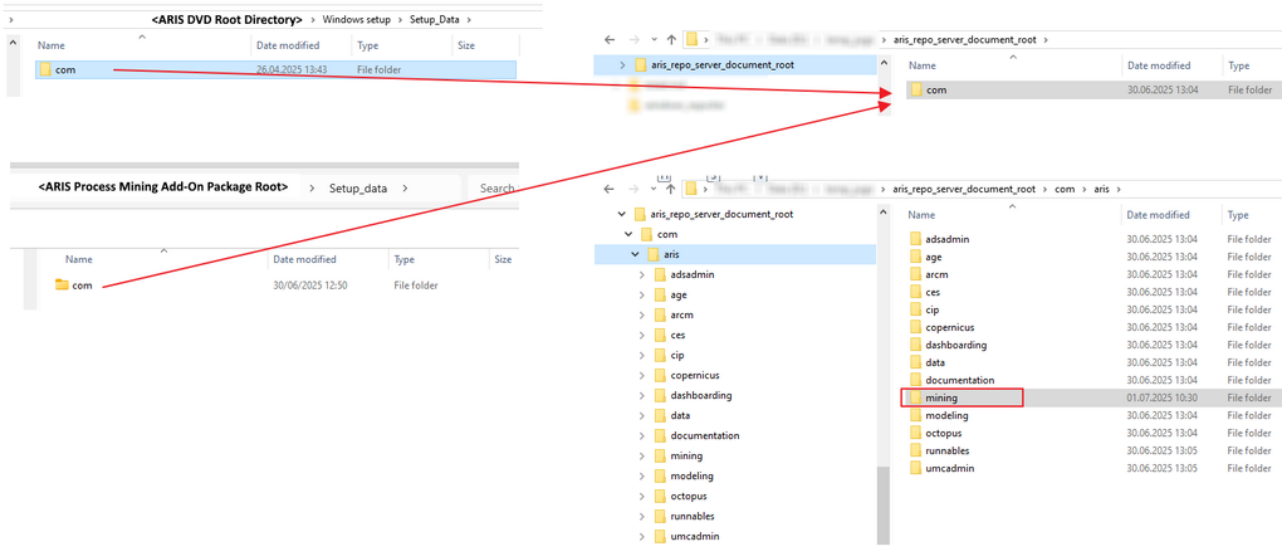
nfsserver.example.com:/aris/miningSrcData /mnt/miningSrcData nfs defaults 0 0

# 6. Providing and filling a runnable repository server

To install the ARIS runnables, the agents - just as in a regular distributed ARIS installation - need to be able to download the ARIS runnables from a repository server. (The section **Preparing and using a remote repository** of [1] [5] explains how to set up this server so that agents can use it.) The section includes a simple example configuration to use a plain Apache **HTTPD** server as repository, if no other suitable file or web servers exist in organization's intranet that can serve the ARIS runnables.

In addition to the steps already outlined - such as copying the ARIS runnables from the ARIS DVD image (located in the **Windows setup/Setup_Data** subfolder) to the directory served by the repository server - you must also copy the ARIS Process Mining runnables. These additional runnables are found on the Process Mining DVD image, specifically in its **Setup_Data** subfolder. All runnables should be placed in the same target directory, as illustrated in the figure below.

Verify that both the ARIS runnables and the Process Mining runnables are correctly placed by opening the **com/aris** folder. Ensure that the expected directories - especially the **mining** directory — are present, as shown in the lower-right corner of the screenshot.




Please note that this process must be repeated using the updated installer packages before deploying a patch.

# 7. Preparing ARIS Cloud Controller

As with a distributed ARIS only installation, the ARIS Cloud Controller tool must be provided with different configuration files:

- the **aforementioned node** file, listing all machines which are to host ARIS runnables (the “worker nodes”), (see chapters **Adding our worker nodes to ACC** to **Creating a node file** of [1] [5]).
- the ARIS-specific ARIS Cloud Controller configuration file **generated.apptypes.cfg**, found on the ARIS DVD at **Windows setup\ARIS_Server\generated.apptypes.cfg**, (see the chapter **Get and use a generated.apptypes.cfg file with your ACC** of [1] [5]).

To make ARIS Cloud Controller aware of the additional Process Mining runnables, you further need the additional Process Mining-specific ARIS Cloud Controller configuration file, located directly in the root of the Process Mining DVD, named “**generated.apptypes.mining.cfg**”. Copy this file next to the ARIS-specific **generated.apptypes.cfg** file.

To make ARIS Cloud Controller use both of these files, simply use the acc’s **-c** command line switch twice, first referencing the ARIS-specific file, second referencing the Process Mining-specific file.

**Note**

Note that the order in which the two **acc** configuration files are passed to ARIS Cloud Controller on the command line is important. Always make sure to pass the ARIS-specific file **generated.apptypes.cfg** with the first **-c** switch, the Process Mining-specific file **generated.apptypes.mining.cfg** with the second **-c** switch.

Assuming that you have a node file **nodes.nf** and assuming that it is located in the same directory as the ARIS Cloud Controller script files **acc.bat** and **acc.sh**, and that the **generated.apptypes.cfg** and **generated.apptypes.mining.cfg** files are located in the same directory, the ARIS Cloud Controller start command is

```
acc -n nodes.nf -c generated.apptypes.cfg -c generated.apptypes.mining.cfg
```

(If your configuration files are located in a different directory than the ARIS Cloud Controller script file, you can also use absolute or relative paths to reference them).

**Note**

Whenever in the following we use **Start ARIS Cloud Controller**, use only the command mentioned above.

# 8. Overview of the additional runnables

The ARIS Distributed Installations Cookbook [5] already describes the well-known ARIS runnables in more details. For ARIS Process Mining, a number of additional (or alternative) runnables are needed:

- **Mining-elasticsearch** - a mining-specific variant of the well-known Elasticsearch backend. In combined ARIS installations, the **mining-elasticsearch** runnable replaces the default Elasticsearch runnable. Since ARIS Process Mining stores the usually large amounts of mining data in Elasticsearch, an Elasticsearch cluster for ARIS with Process Mining requires a significantly larger Elasticsearch cluster.
- **CDF** - while technically not a new runnable (as it is also needed when using plain ARIS with the **hdserver** add-on runnable), CDF is also a mandatory requirement for Process Mining and thus has to be part of a combined ARIS installation (even if no **hdserver** runnables are being used).
- **Miningserver** - mandatory, contains the ARIS Process Mining user interfaces and parts of the Process Mining business logic.

- **Spark-related runnables** - Spark is the underlying technology for most of the actual Process Mining business logic for case-centric process mining. In the scope of ARIS Process Mining, we can distinguish Spark-based tasks that are comparatively short-running (from a few seconds to about 1 minute) and have limited compute resource requirements from those that are long-running (minutes to hours) and require large amounts of compute resources. The term “compute resource” usually focuses on CPU and memory, but also includes (temporary) disk space and I/O bandwidth etc.

  Examples for short-running Spark tasks (referred to as (Spark) short-runners) are the calculation of source data previews and the import of small amounts of source data. Examples for long-running Spark tasks (referred to as (Spark) long-runners) are the import of large amounts of source data, the actual “mining” of processes on the source data, or the backup and restore of mining-related data during the backup and restore tenant operations.

  Each long-runner consists of one “driver” process and one or many “executor” processes.

  Spark-based tasks are split across three different flavors of Spark runnables:

  - **Leansparkservice (LSS)** - executes short-runners, offering minimal startup overhead, but places strict limits on the available resources. At any given time, a single LSS instance can run at most one short-runner. Therefore, the number of LSS instances should be decided based on the expected number of parallel executions of short-running tasks. If more short-runners are started than there are LSS instances available, the short-runners are queued for a short time. If an initially busy LSS instance becomes available again, it picks up the queued short-runner. Otherwise, the user is informed that waiting for an available LSS instance has timed out. Not used when the object-centric-only license is active.
  - **Miningsparkworker** (MSW or just “worker”) - makes (almost) all computational resources of an entire worker node (that is, an entire ARIS server machine) available for executing long-running Spark-based tasks. It is therefore recommended to set aside dedicated nodes for Spark long-runners. On these nodes, you should only install the ARIS Cloud Agent and then deploy only the **Miningsparkworker** runnable. We refer to such nodes as “Spark worker nodes”. You can have a practically unlimited number of Spark worker nodes - the actual required number depends on the amount of concurrent long-running Spark tasks and the expected duration (which is usually related to the amount and complexity of the source data on which mining is performed). If needed, additional spark worker nodes can be added to the ARIS cluster later. If an object-centric-only license is active, the worker runs only tenant management operations. As a result, the Mining Spark worker can also run on a node with other runnables.
  - **Miningsparkmaster** (MSM or just “master”) - coordinator for long-running tasks on mining data. It decides how to schedule the driver and executor processes of Spark long-runners across the different Spark worker nodes (that is, those that have the **Miningsparkworker** runnable) deployed and running on them. At least one Miningsparkmaster runnable must be configured and started at any given time. If multiple Miningsparkmaster runnables are running, one of them handles the coordination of Spark tasks, while all others act as “hot standby” to take over the role if the current coordinator fails. Since the Spark-master restarts quickly in the event of a crash and does not have high resource requirements, it can co-exist with other ARIS and Process Mining runnables on the same ARIS worker node.
- **Miningcomputeorchestrator** (used only for object-centric process mining) - orchestrates the pool of configured compute engines. It provides a free engine for a specific request (for example, data queries on a dataset or executing a data load for a dataset). If all engines are in use, it responds with an appropriate message.
- **Leancomputationservice** (used only for object-centric process mining) is responsible for executing synchronous, short-running requests, such as data previews and file uploads.
- **Computeengines** (used only for object-centric process mining) Dedicated engines that execute the following tasks: data queries, data import, data load, data modification, and tenant management actions. Engines can be configured in three sizes (S, M, L). The system automatically selects the appropriate size for each action. If no engine of the suitable size is available, a fallback mechanism uses a larger engine instead. While an engine is in use, it is exclusively attached to its assigned action and cannot be used for other tasks. It is therefore important to configure an appropriate number and sizing of engines for the number of datasets and users.

# 9. Example scenario

In the following sections, we will describe an example deployment of a combined ARIS with Process Mining.

## 9.1. Prerequisites

The example scenario will use five ARIS worker nodes (two of these are set aside as pure Spark worker nodes). The actual hardware requirements of these machines depends on the chosen sizing (= vertical scaling) for the runnables. As a rule of thumb, the machines should have 16 CPU cores, 64GB of RAM and about 1TB of available disk space. In addition, we assume that an external relational database server is available and will be used (as described in the chapter **Relational database backends** of [1] [5]).

We assume that the ARIS Cloud Agent are successfully installed on all five machines as described above and the network shares needed by the product are mounted on the machines (each physical share being mounted to the same driver letter (Windows) respectively mount point (Linux) on each node). We further assume that you have a node file prepared as discussed above, and named the nodes n1 to n5.

**Example 3.**

add node n1 <hostnameOrIpAddressOfNode1> <agentUsername> <agentPassword>

add node n2 <hostnameOrIpAddressOfNode2> <agentUsername> <agentPassword>

add node n3 <hostnameOrIpAddressOfNode3> <agentUsername> <agentPassword>

add node n4 <hostnameOrIpAddressOfNode4> <agentUsername> <agentPassword>

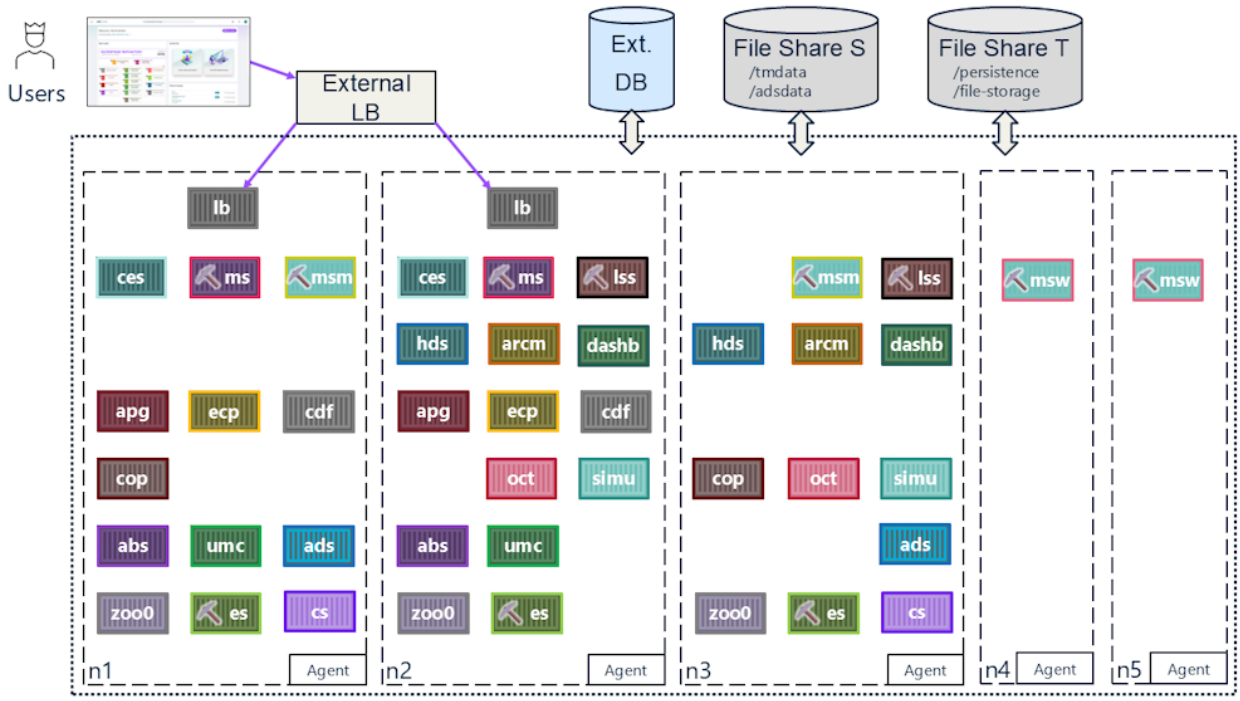
add node n5 <hostnameOrIpAddressOfNode5> <agentUsername> <agentPassword>

## 9.2. Overview of the example scenario (case-centric)

Figure 1 below shows our example scenario: nodes **n1** to **n3** hold most of the ARIS and Process Mining runnables. Assuming identically sized worker nodes, we distributed them to balance out CPU and memory usage across these three nodes. Nodes **n4** and **n5** are set aside purely for long-running Process Mining Spark tasks and therefore only hold a **Miningsparkworker** runnable each. Those runnables that are specific to mining are shown with a little pickaxe symbol.

**Example 4.**

Figure 1 - Overview of the example scenario

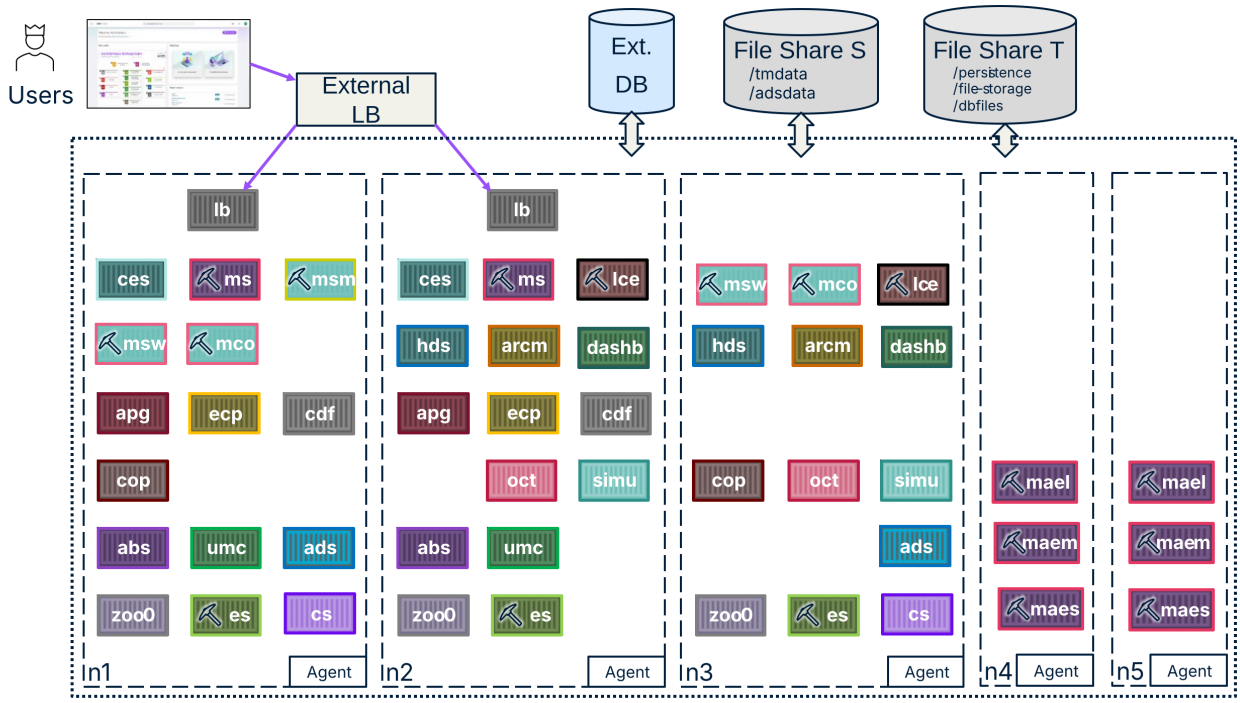



## 9.3. Overview of the example scenario (object-centric)

Nodes n1 to n3 host most of the ARIS and Process Mining runnables, distributed to balance CPU and memory usage across the three nodes. This includes the miningsparkworker (msw), which, unlike in a case-centric deployment, is co-located on one of these nodes rather than requiring a node of its own. Nodes n4 and n5 are reserved for the object-centric compute engines (sizes S, M, and L). The DuckDB temp directory used by these engines should be on local disk, and each engine is fully occupied while processing a request. The lean computation service, which handles short, synchronous requests, can also run on n4 and n5.

**Example 5.**

Figure 2- Overview of the example scenario (object-centric)




The next section describes the commands for deploying the runnables (called “configuring” in ARIS Cloud Controller terminology). A second section lists additional configuration steps, such as connecting ARIS to an external database system or a mail server. These steps are identical to those of a regular ARIS deployment set up manually with ARIS Cloud Controller, so please refer to the relevant documentation.

Continuing the example from the earlier discussion of file shares in ARIS with Process Mining, we assume two physical file shares. The first is mounted to drive letter **S** and contains the subdirectories **tmdata** and **adsdata** for the corresponding logical shares. The second is mounted to drive letter **T** and contains the subdirectories persistence and file-storage for the two remaining logical shares.

We do not go into detail on the configuration of the well-known ARIS runnables (see [1] [5] for more information). In our example commands, we generally use the "large" sizing of each runnable - depending on the sizing recommendation provided to you, you might be using the "small" or "medium" sizings instead. In an object-centric-only deployment, the exception is Elasticsearch, for which we deliberately use the "small" sizing, because in this scenario it holds only configuration data (see "Sizing" above).

Before running the configure commands, make sure all agents are set to use the runnable repository server mentioned earlier as the source for downloading runnables. For details, see the chapters **Preparing and using a remote repository** and **Making the agents on the worker nodes use the remote repository** in [1] [5]).

## 9.4. Deploying the ARIS and Process Mining runnables

### 9.4.1. Configuring the Zookeeper cluster

To deploy a Zookeeper cluster consisting of three instances for high-availability using the “large” sizing, run the commands:

```
set acc zkmgr.zookeeper.app.type=zoo_l
on n1 add zk instance using zoo
```

```
on n2 add zk instance using zoo
on n3 add zk instance using zoo
validate zk changes
commit zk changes
```

The first command makes the subsequent commands use the “large” sizing if the Zookeeper is runnable. The **add zk instance** commands “stage” the desired operation - that is, adding a Zookeeper on each node n1 to n3. The **using zoo** part of these commands overrides the default instanceId (= name of the runnable, the default naming scheme would name the first (and only) Zookeeper on each node “zoo0” instead). The **validade zk changes** command checks if the desired changes are okay, and the **commit zk changes** command applies them, that is, configure and directly start the three Zookeeper runnables.

Once the last command completes, confirm that you have a Zookeeper runnable configured (and already running on each node n1 to n3, by running:

```
on all nodes list
```

or

```
list zk instances
```

### 9.4.2. Configuring the Elasticsearch cluster

In object-centric-only mode, the Elasticsearch instances can be sized smaller. To configure a three-node Elasticsearch cluster (using the Process Mining-specific variant of the Elasticsearch runnable), run the commands

```
set acc esmgr.elasticsearch.app.type=miningelastic_l
on n1 add elasticsearch instance to cluster ariscluster using es
on n2 add elasticsearch instance to cluster ariscluster using es
on n3 add elasticsearch instance to cluster ariscluster using es
validate es changes
commit es changes
```

This configures and starts the three Elasticsearch instances. However, since Elasticsearch is involved in the tenant backup procedure, it needs to be given the path to the mount point of the **tmdata** network share that holds the tenant backups. This can be done by setting the configure parameter to the path where we mounted this share (“S:\tmdata” in our example, adjust to your mount location accordingly) with the following reconfigure commands (not that you must either use a forward slash (“/”) or an “escaped” backward slash (“\\) in the path):

```
on n1 reconfigure es ELASTICSEARCH.path.repo="s:/tmdata"
on n2 reconfigure es ELASTICSEARCH.path.repo="s:/tmdata"
on n3 reconfigure es ELASTICSEARCH.path.repo="s:/tmdata"
```

### 9.4.3. Configuring the Cloudsearch cluster

We deploy two cloudsearch instances. By setting the **zookeeper.application.instance.datacenter** parameter to different values on each instance, this gives us the scenario where all data is held in each instance,

providing high availability and load distribution (as queries can be sent to either instance), (as described in chapter **Cloudsearch clustering, variant 2: high availability and query load distribution, no data load distribution** of [1] [5]).

```
on n1 configure cloudsearch_l cs zookeeper.application.instance.datacenter=A
on n3 configure cloudsearch_l cs zookeeper.application.instance.datacenter=B
```

### 9.4.4. Configuring the ARIS microservices

The following commands configure all of our well-known ARIS microservices, two instances each for high-availability and load distribution. We also set a number of additional parameters required for the combined ARIS installation or for a distributed installation in general:

- Since we put an external loadbalancer in front of our ARIS deployment through which all ARIS users will access the system, the two loadbalancers runnables need to be configured to know the hostname of the external loadbalancer. This is set with the **HTTPD.servername** parameter (here we assume again that a DNS entry **myaris.mycompany.com** points to the external LB.), see lines 2 and 5, respectively. Further, as discussed before, the external loadbalancer must pass information about the original client's request to the ARIS loadbalancer with the various **X-Forwarded-*** headers. For security reasons, the ARIS loadbalancer however accepts these headers only from "trusted" source IP addresses (that is, the IP address(es) of the external loadbalancer). These IP addresses need to be set with a matching regex with the **HTTPD.X-Forwarded-For.trusted.proxy.regex** parameter, (as discussed in detail in the chapter **The X-Forwarded-* headers** of [1] [5]). In our example, we assume that our external loadbalancer has the IP address 10.1.2.3, resulting in a regular expression "10\\\\.1\\\\.2\\\\.3" (Note the use of four backslashes to escape the special meaning of the "dot" character in regular expressions".), see lines 3 and 6.

  Adjust the hostname and IP address to your specific environment.
- Since the **UMCAdmin** runnable also contains the tenant management application, which handles tenant backups, this is the second place (after Elasticsearch) where we need to specify the path to the **tmdata** share with the **JAVA-Dtm.backup.folder** parameter (see lines 9 and 13, adjust the folder to where you mounted the **tmdata** share in your environment).

  Further, an optimized backup mode that improves handling the large amounts of data expected in the scope of Process Mining is enabled with two more parameters, **JAVA-Dcom.aris.tm.network.share** and **JAVA-Dcom.aris.tm.partial.backup.restore.app.types**, respectively (lines 10 and 14 and lines 11 and 15, respectively).
- The setting **JAVA-Dabs.ces_active=true** on the **ABS** runnables (lines 17 and 18) enables the new improved search functionality) (provided with the help of the **CES** runnable)
- For ADS, we switch it from the default configuration, where the documents are stored as "BLOBs" in the relational DBMS to storing them on the file system, and point it to where we mounted the **adsdata** share (lines 21 and 22 resp. 24 and 25, adjust the folder to where you mounted the **adsdata** share in your environment).

Also note the commented-out commands for configuring the optional **HDServer** and **ARCM** runnables, which you can add if you want to use the ARIS Risk and Compliance functionality or expect a lot of ARIS reports, which you want to offload from the **ABS** runnables, respectively.

```
on n1 configure loadbalancer_l loadbalancer_l \
HTTPD.servername="myaris.mycompany.com" \
HTTPD.X-Forwarded-For.trusted.proxy.regex="10\\\\.1\\\\.2\\\\.3"
on n2 configure loadbalancer_l loadbalancer_l \
HTTPD.servername="myaris.mycompany.com" \
```

```
HTTPD.X-Forwarded-For.trusted.proxy.regex="10\\\\.1\\\\.2\\\\.3"
on n1 configure umcadmin_l umcadmin_l \
JAVA-Dtm.backup.folder = "s:/tmdata" \
JAVA-Dcom.aris.tm.network.share="true" \
JAVA-Dcom.aris.tm.partial.backup.restore.app.types="MININGSERVER"
on n2 configure umcadmin_l umcadmin_l \
JAVA-Dtm.backup.folder = "s:/tmdata" \
JAVA-Dcom.aris.tm.network.share="true" \
JAVA-Dcom.aris.tm.partial.backup.restore.app.types="MININGSERVER"
on n1 configure abs_l abs_l JAVA-Dabs.ces_active="true"
on n2 configure abs_l abs_l JAVA-Dabs.ces_active="true"
on n1 configure adsadmin_l adsadmin_l \
JAVA-Dcom.aris.ads.filesystem.active="true" \
JAVA-Dcom.aris.ads.filesystem.path="s:/adsadata"
on n3 configure adsadmin_l adsadmin_l \
JAVA-Dcom.aris.ads.filesystem.active="true" \
JAVA-Dcom.aris.ads.filesystem.path="s:/adsadata"
on n1 configure copernicus_l copernicus_l
on n3 configure copernicus_l copernicus_l
on n2 configure octopus_l octopus_l
on n3 configure octopus_l octopus_l
on n2 configure simulation_l simulation_l
on n3 configure simulation_l simulation_l
on n1 configure apg_l apg_l
on n2 configure apg_l apg_l
on n1 configure ecp_l ecp_l
on n2 configure ecp_l ecp_l
on n2 configure dashboarding_l dashboarding_l
on n3 configure dashboarding_l dashboarding_l
on n1 configure cdf_l cdf_l
on n2 configure cdf_l cdf_l
on n1 configure ces_l ces_l
on n2 configure ces_l ces_l
#on n2 configure hds_l hds_l
#on n3 configure hds_l hds_l
#on n2 configure arcm_l arcm_l
#on n3 configure arcm_l arcm_l
```

### 9.4.5. Configuring the mining-specific microservices (case-centric)

The following section of configure commands now adds the Process Mining-specific runnables to our example scenarios:

For high-availability (and load distribution), we have two instances each of the Process Mining Server, the Process Mining Spark Master, and the Lean Spark Service (used for Spark-based short-runners). Further, we have one **Spark worker** runnable on each of our dedicated **Spark worker** nodes n4 and n5.

Note that we need to pass the path to the **persistence** and **file-storage** shares to both all the **miningserver** and **leansparkservice** runnables with the **persistence.base-path** and **file-storage.base-path** parameters – as before, replace the paths used in our example with those of your environment.

```
on n1 configure miningserver_l ms \
   file-storage.base-path="T:/file-storage" \
```

```
      persistence.base-path="T:/persistence"
on n2 configure miningserver_l ms \
   file-storage.base-path="T:/file-storage" \
      persistence.base-path="T:/persistence"

on n1 configure miningsparkmaster_l msm
on n3 configure miningsparkmaster_l msm

on n2 configure leansparkservice_l lss \
   file-storage.base-path="T:/file-storage" \
      persistence.base-path="T:/persistence"
on n3 configure leansparkservice_l lss \
   file-storage.base-path="T:/file-storage" \
      persistence.base-path="T:/persistence"

on n4 configure miningsparkworker_l msw
on n5 configure miningsparkworker_l msw
```

### 9.4.6. Configuring the mining-specific microservices (object-centric)

Unlike in the general document, this section does not configure a Leansparkservice (LSS), because it is not needed for an object-centric-only deployment. We configure two Mining Server instances (for high availability) and the Spark coordination runnables. Note that the Miningsparkworker is placed on a regular node (n3) rather than on a dedicated Spark worker node:

```
on n1 configure miningserver_l ms \
    file-storage.base-path="T:/file-storage" \
    persistence.base-path="T:/persistence"
on n2 configure miningserver_l ms \
    file-storage.base-path="T:/file-storage" \
    persistence.base-path="T:/persistence"

on n1 configure miningsparkmaster_l msm
on n3 configure miningsparkmaster_l msm

on n3 configure miningsparkworker_l msw
```

Note the absence of any "add Spark worker node" step and of any LSS configuration here, compared to the general document. Note that, because the Miningsparkworker only coordinates the lightweight backup/restore trigger in this scenario, a single instance is generally sufficient. If you want redundancy, add a second instance on another node (for example, on n1: configure miningsparkworker_l msw).

### 9.4.7. Adding additional services to enable the object-centric process mining feature

This example shows two instances of each engine size (S, M, and L).

```
on n1 configure miningcomputeorchestrator_l mco \
    persistence.base-path="T:/persistence"
on n3 configure miningcomputeorchestrator_l mco \
    persistence.base-path="T:/persistence"
```

```
on n2 configure leancomputeengine_l lce \
    duckdb.root_folder="T:/dbfiles" \
    file-storage.base-path="T:/file-storage" \
    persistence.base-path="T:/persistence"
on n3 configure leancomputeengine_l lce \
    duckdb.root_folder="T:/dbfiles" \
    file-storage.base-path="T:/file-storage" \
    persistence.base-path="T:/persistence"

on n4 configure mininganalysisengine_s maes \
    file-storage.base-path="T:/file-storage" \
    persistence.base-path="T:/persistence" \
    duckdb.root_folder="T:/dbfiles" \
    duckdb.temp-directory="C:/dbtempfiles/meas"
on n5 configure mininganalysisengine_s maes \
    file-storage.base-path="T:/file-storage" \
    persistence.base-path="T:/persistence" \
    duckdb.root_folder="T:/dbfiles" \
    duckdb.temp-directory="C:/dbtempfiles/meas"

on n4 configure mininganalysisengine_m maem \
    connector.http.port=24600 management.server.port=24608 \
    JAVA-Dcom.sun.management.jmxremote.port="24606" \
    JAVA-Dcom.sun.management.jmxremote.rmi.port="24606" \
    file-storage.base-path="T:/file-storage" \
    persistence.base-path="T:/persistence" \
    duckdb.root_folder="T:/dbfiles" \
    duckdb.temp-directory="C:/dbtempfiles/meam"
on n5 configure mininganalysisengine_m maem \
    connector.http.port=24600 management.server.port=24608 \
    JAVA-Dcom.sun.management.jmxremote.port="24606" \
    JAVA-Dcom.sun.management.jmxremote.rmi.port="24606" \
    file-storage.base-path="T:/file-storage" \
    persistence.base-path="T:/persistence" \
    duckdb.root_folder="T:/dbfiles" \
    duckdb.temp-directory="C:/dbtempfiles/meam"

on n4 configure mininganalysisengine_l mael \
    connector.http.port=34600 management.server.port=34608 \
    JAVA-Dcom.sun.management.jmxremote.port="34606" \
    JAVA-Dcom.sun.management.jmxremote.rmi.port="34606" \
    file-storage.base-path="T:/file-storage" \
    persistence.base-path="T:/persistence" \
    duckdb.root_folder="T:/dbfiles" /
    duckdb.temp-directory="C:/dbtempfiles"
on n5 configure mininganalysisengine_l mael \
    connector.http.port=34600 management.server.port=34608 \
    JAVA-Dcom.sun.management.jmxremote.port="34606" \
    JAVA-Dcom.sun.management.jmxremote.rmi.port="34606" \
    file-storage.base-path="T:/file-storage" \
    persistence.base-path="T:/persistence" \
    duckdb.root_folder="T:/dbfiles" /
    duckdb.temp-directory="C:/dbtempfiles"
```

For high availability, we configure two instances of the orchestrator (MCO), which orchestrates the engines, and two leancomputeengines, which execute synchronous, short-running requests (for example, uploads and previews). When configuring the engines (MAES, MAEM, MAEL), be aware of two things:

- If multiple engines run on the same node, you must define different connection ports, as shown in the code sample above.
- The DuckDB temp directory must not be located on a network share. For performance reasons, it should be on the local disk of the node.

If you use multiple engines of the same size (S, M, or L), configure them with the same CPU and memory settings.

Larger data volumes may require increasing the CPU and memory settings of the L engines. To determine the right number and size of engines, see the default limits.

```
DataImport (csv file size)
0<5Gb -> s size
5-10Gb -> m size
>10Gb -> l size

DataModification (csv file size):
0<512Mb -> s size
512Mb-2Gb -> m size
>2Gb -> l size

DataLoad (csv file size):
0<10Mb -> s size
10Mb-100Mb -> m size
>100Mb -> l size

Query engine (storage size):
0<100Mb -> s size
100Mb-1000Mb -> m size
>1000Mb -> l size
```

## 9.5. Additional configuration steps

### 9.5.1. Configuring the external DBMS and adding the JDBC drivers

To register the external database system with ARIS, the ARIS Cloud Controller command **register external service** is used. The details depend on the operating system of your worker nodes and the details of the DBMS. Please refer to the existing documentation:

If you are using Windows worker nodes, refer to the following chapters of [3] [5]:

- **Installing ARIS server using a Microsoft SQL Server (mixed mode)** if you are using Microsoft SQLServer with Windows authentication in “mixed mode”,
- **Installing ARIS server using a Microsoft SQL Server (Windows authentication/SQL Server authentication)** if you are using Microsoft SQLServer with Windows and SQL Server authentication,
- **Installing ARIS server using an Oracle database** if you are using an Oracle database.

If you are using Linux worker nodes, refer to the following chapters of [4] [5]:

- **Installing ARIS server using a Microsoft SQL Server** if you are using Microsoft SQL Server
- **Installing ARIS server using an Oracle database** if you are using an Oracle database.

Since the product does not ship with the proprietary JDBC drivers needed to access the DBMSs, you further need to add them to all runnables that make use of the RDBMS, (as explained in the chapter **Completing the Installation** of [1] [5]). Only some of the ARIS (but none of the Process-Mining-specific) runnables use the RDBMS, in particular the **ABS**, **Copernicus**, **ECP**, **Octopus**, **ADSAdmin**, **Simulation**, **Dashboarding**, and **APG** runnables. If you configured the optional **HDServer** or **ARCM** runnables, they also need to be supplied with the **JDBC** driver. In our example, the corresponding **enhanceall** commands would be the following (replace the path to the driver JAR file to point to where the JDBC driver is located on the machine from which you run the ARIS Cloud Controller), with the ones for the optional runnables commented out again:

```
on all nodes enhanceall abs_l with commonsClasspath path "jdbc/driver.jar"
on all nodes enhanceall copernicus_l with commonsClasspath path "jdbc/
driver.jar"
on all nodes enhanceall ecp_l with commonsClasspath path "jdbc/driver.jar"
on all nodes enhanceall octopus_l with commonsClasspath path "jdbc/driver.jar"
on all nodes enhanceall adsadmin_l with commonsClasspath path "jdbc/driver.jar"
on all nodes enhanceall simulation_l with commonsClasspath path "jdbc/
driver.jar"
on all nodes enhanceall apg_l with commonsClasspath path "jdbc/driver.jar"
on all nodes enhanceall dashboarding_l with commonsClasspath path "jdbc/
driver.jar"
#on all nodes enhanceall hds_l with commonsClasspath path "jdbc/driver.jar"
#on all nodes enhanceall arcm_l with commonsClasspath path "jdbc/driver.jar"
```

### 9.5.2. Registering an external mail server

To register your company mail server to enable the product to send notification mails etc., you also need to register your SMTP server as an “external service”. The procedure is described in the chapter **Mailing** of [3] [5] or [4] [5]. The procedure is identical regardless of the operating system of your worker nodes.

# 10. Patching or updating a combined ARIS installation

The ARIS Update Cookbook [5] and the ARIS Server Update Installation Guide [5] give general guidance on how to update an ARIS installation, which also apply to a combined ARIS installation. This section therefore only gives a quick rundown of the common steps and focuses on specific topics to consider when updating a combined ARIS installation.

**Note**

Note that while patching or updating an ARIS installation is safe procedure, it is mandatory to perform a backup of your ARIS data before doing the update. This can be done by doing a backup of all the tenants in your ARIS installation with the tenant management application. The backup and restore procedures are described in the next chapter.

To apply a patch to your combined ARIS installation, you receive the new version of the two installation packages. The name of these packages contain a version number, following the scheme:

```
10.<YEAR>.<MONTH>.0.<BUILD_NUMBER>
10.2025.7.0.310006811
```

Year and month are probably self-explanatory. The build number is a 9- (or more) digits number, higher numbers indicating newer versions. A “patch” is usually the term used when switching to a version that has the same year and month as the currently installed version, with only a higher build number. If month and/or year change, this is considered an “update”. From a purely technical perspective, the basic steps for a patch or update are the same, however, an update might require additional steps.

**Note**

There are restrictions on which “source” version can be patched or updated to which “target” version. When a patch is supplied to you, make sure that it is only applied on the intended source versions.

The first step after receiving the new installation packages is to copy them to your **repo** server, just like when preparing the repo for the initial installation. You can copy the files into the same directory as the ones of previous versions, as the paths and file names are non-conflicting. So a single **repo** server can serve arbitrarily many ARIS versions (given there is enough free disk space).

Assuming you use the same **repo** server, it is not needed to change the **remote.repository.url** setting of all ARIS agents. If you set up a new **repo** server (or use a different subdirectory on the same server), you need to update this setting with

```
on all nodes set remote.repository.url="<NEW_REPO_URL>"
```

Contained in the installation packages are new versions of the ARIS Cloud Controller configuration files for the ARIS and the Process Mining runnables, the **generated.apptypes.cfg** and **generated.apptypes.mining.cfg** files, respectively. Use these instead of the old ones when starting ARIS Cloud Controller with the **-c** switches.

Installing a patch does not always require updating the ARIS Cloud Controller and the cloud agents - the instructions coming with the specific patch mention this. Installing an update always requires an update of agents and ARIS Cloud Controller. (Instructions on how to update agents and ARIS Cloud Controller can be found in [7] [5].)

In rare cases, when updating from a very old version, it might be necessary to first update to an intermediate version, start the system, perform some migration steps, and only then upgrade to the desired target version.

If updating the ARIS Cloud Agent and ARIS Cloud Controller was necessary, you have already been instructed to stop all runnables. If no update of the agents was needed, make sure to stop the runnables by running the ARIS Cloud Controller command:

```
on all nodes stopall
```

To perform a pre-check and “dry run” of the update, you can run the command:

```
on all nodes check updateall
```

This performs various pre-checks (for example, if there are any still running runnables, if the new version of the runnables can actually be found on the repo server etc.).

If this shows any problems, resolve them first.

To perform the actual update of all runnables, run the command:

```
on all nodes updateall
```

To avoid being asked for confirmation for certain steps, you can

also run:

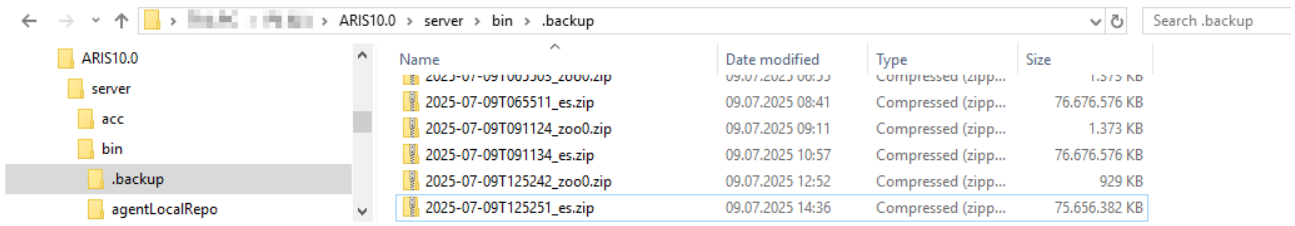
```
on all nodes force updateall
```

**Note**

The **updateall** command logs the progress on the ARIS Cloud Controller console. Note that the update steps for the individual runnables can take considerable amount of time. This is not so much because of having to download and prepare the new runnable version, but because before an update of a runnable, the agent performs a backup of the runnable’s working directory (that is., the directories located at **<installDir>/server/bin/work/wokr_<instanceId>**) to allow to easily return to the previous version in case of problems. The duration of this backup procedure depends on the number and size of files located in the runnable’s working directory. In the context of a combined ARIS installation, this is a particular issue for the **Elasticsearch** runnables, as this is where the mined process data is being stored. Thus, large amounts of process data leads to large runnable backups for Elasticsearch, and thus correspondingly large disk space requirements and duration of the backup.

Thus, if a runnable’s update seems stuck, this is most likely the backup happening under the hood. Unfortunately, there is currently no indication of the progress of this procedure.

If you are really curious if it is really the runnable backup that is taking so long, you can navigate to the directory **<installDir>/server/bin/.backup** on the worker node where the runnable that is currently updating is located. There you find a ZIP archive for each currently held runnable backup, as shown in the screenshot below:




In this example, you can see multiple rather small backups for a **Zookeeper** runnable (“zoo0”) and multiple large (~75GB) backups for an **Elasticsearch** runnable (“es”). If the backup is still being written, you see the corresponding file slowly growing.

Once the command completes, you can confirm with:

```
on all nodes list
```

that all runnables show the new version.

You can now start all your runnables again with:

```
on all nodes startall
```

**Note**

After a patch or update, some data migration steps might be necessary on the first start of the system. This can increase the startup times considerably. Please be patient and do not interrupt the startup.

# 11. Tenant management

ARIS is a multi-tenant capable application, that is, a single physical ARIS installation like the one we created in the previous sections can hold data of different organizations (this could be different companies, or just different departments of the same company) in different “tenants” (which can be seen as a “logical ARIS installation”). Even though handled by the same ARIS installation, the data (users, models, documents, dashboards, ….) in different tenants is completely separate from that of the other tenants. Any freshly set up ARIS installation comes with two tenants: the **default** tenant is meant for regular use of the product. The **master** tenant is for administering (creating, deleting, backing up and restoring) all other tenants.

In ARIS, tenant management (that is, the creation and deletion of tenants, and the backing up or restoring all data of a tenant) was originally performed with special ARIS Cloud Controller commands only.

With newer ARIS versions, a new dedicated **Tenant Management (TM)** application (included in the **UMCAdmin** runnable) was introduced. That offers both a browser-based user interface and a command line interface for all tenant management functionality. The **Tenant Management (TM)** application provides a dedicated Tenant Management online help [5].

Support for the tenant management commands in ARIS Cloud Controller was kept, but is now deprecated. Now, with the introduction of ARIS with ARIS Process Mining, the ARIS Cloud Controller tenant management functionality is no longer supported. All tenant management operations must be performed via the new tenant management application. You receive an error message if you try to use any of the ARIS Cloud Controller commands related to tenant management when Process Mining is part of an ARIS installation.

Tenant management can be used via its web-based user interface, that can be accessed at **<ARIS_Server_URL>/tm**. Documentation for the UI can be accessed via the online help in the UI. Log in with the user name and password of a user in the **master** tenant (the default users are the same as the well known ones of the **default** tenant, but of course must have their passwords changed).

As mentioned, as an alternative to the UI TM also offers a command line interface (CLI) for all relevant functionality. This is particularly useful to automate any tenant management operations (for example, performing regular tenant backups by running the TM CLI from a scheduling tool like the Windows Task Scheduler, or as a cron job on Linux). The TM CLI can be found in the **work** directory of any **UMCAdmin** runnable at **<UMCAdminWorkingDir>\tools\tm\bin**. There you find scripts files to start the CLI for both Windows (**tm.bat**) and Linux (**tm.sh**). In all the examples below, we omit the file extension - simply use the script for the OS from which you run the CLI.

Here we focus on the functions for backing up and restoring tenants.

## 11.1. Backup modes

When doing a tenant backup, TM produces an archive file that contains all data of that tenant held in all microservices. When backups are done via the UI, the files are stored in the **tmdata** share (and can be downloaded to an admin user’s machine via the UI if desired). When doing a backup with the TM CLI, the backup file is directly downloaded onto the machine from which TM is being run.

While having a single file that contains everything is convenient and the file is portable, this approach has disadvantages when the amounts of data grow, as is to be expected with ARIS Process Mining: backup times increase and can become prohibitive, the size of the backup file grow and make it unwieldy to move and handle. Therefore, with the combined ARIS installation, a new mode backup is introduced, that makes more direct use of the network share and also offers optional incremental backups (that is, a backup run contains only data that was added or changed since the previous run). We refer to this new mode as **direct-2-networkshare** (**D2S**) mode, while the existing mode is referred to as **classical backup** mode.

We first give a quick introduction on how to use the **classical backup** mode with the TM CLI. This mode is simpler to use and perfectly acceptable while the amount of data is moderate (= a few tens of GB). Then we introduce the new **D2S** mode both using it via the TM UI and the CLI.

## 11.2. Classical tenant backup and restore with the TM CLI

The basic structure of the command to run the classical backup is:

```
tm -a backup -s <ARIS_Server_URL> -u <username> -p <password> -t <tenant> -f
<backupFile>
```

The **<ARIS_Server_URL>** parameter is the same used as by the end users (the URL pointing to the external loadbalancer in our distributed environment above). **<username>** and **<password>** are the credentials of an **admin** user in the **master** tenant. **<tenant>** is the name of the tenant to be backed up. **<backupFile>** is the (relative or absolute) path to the resulting backup file. Select the appropriate type of script file for your operation system.

For example, to backup the **default** tenant on our distributed example system from above, assuming it's external LB is reachable via https://myaris.mycompany.com/, and assuming that (against recommendation) the password of the master tenant's system user was not changed from its default value **manager**, could be backed up to a file with the command:

```
tm -a backup -s https://myaris.mycompany.com/ -u system -p manager -t default
-f backup_default.zip
```

To restore a classical tenant backup file, the command structure is:

```
tm -a restore -s <ARIS_Server_URL> -u <username> -p <password> -t <tenant> -f
<backupFile>
```

for example, to restore the backup file created with the previous command, we run:

```
tm.bat -a restore -s https://myaris.mycompany.com/ -u system -p manager -t
default -f back
```

## 11.3. Direct-2-network share backup and restore with the TM CLI

When handling large volumes of data, the usual backup method via TM presents several challenges. Under the hood, the TM application interacts with nearly all relevant ARIS microservices to retrieve the data each one manages for the tenant being backed up.

The application gathers all the data from the backends, collects them into a single file, and sends it back to TM. TM collects all the backup parts provided by the microservices and then puts it into the single backup file that contains all data. Consequently, data is copied around multiple times, which makes this slow when the amount of data increases.

The **direct-2-network-share (D2S)** mode is based on the idea that applications can (if they support D2S) put their part of the tenant's data directly onto the **tmdata** network share, right next to the main backup file that is still created by TM (and which contains the tenant's data for all those applications that do not (yet) support D2S). That way, the app's data does not need to flow back to TM, need not be copied into the main backup file, and only then be put onto the **tmdata** share, but takes a more direct route.

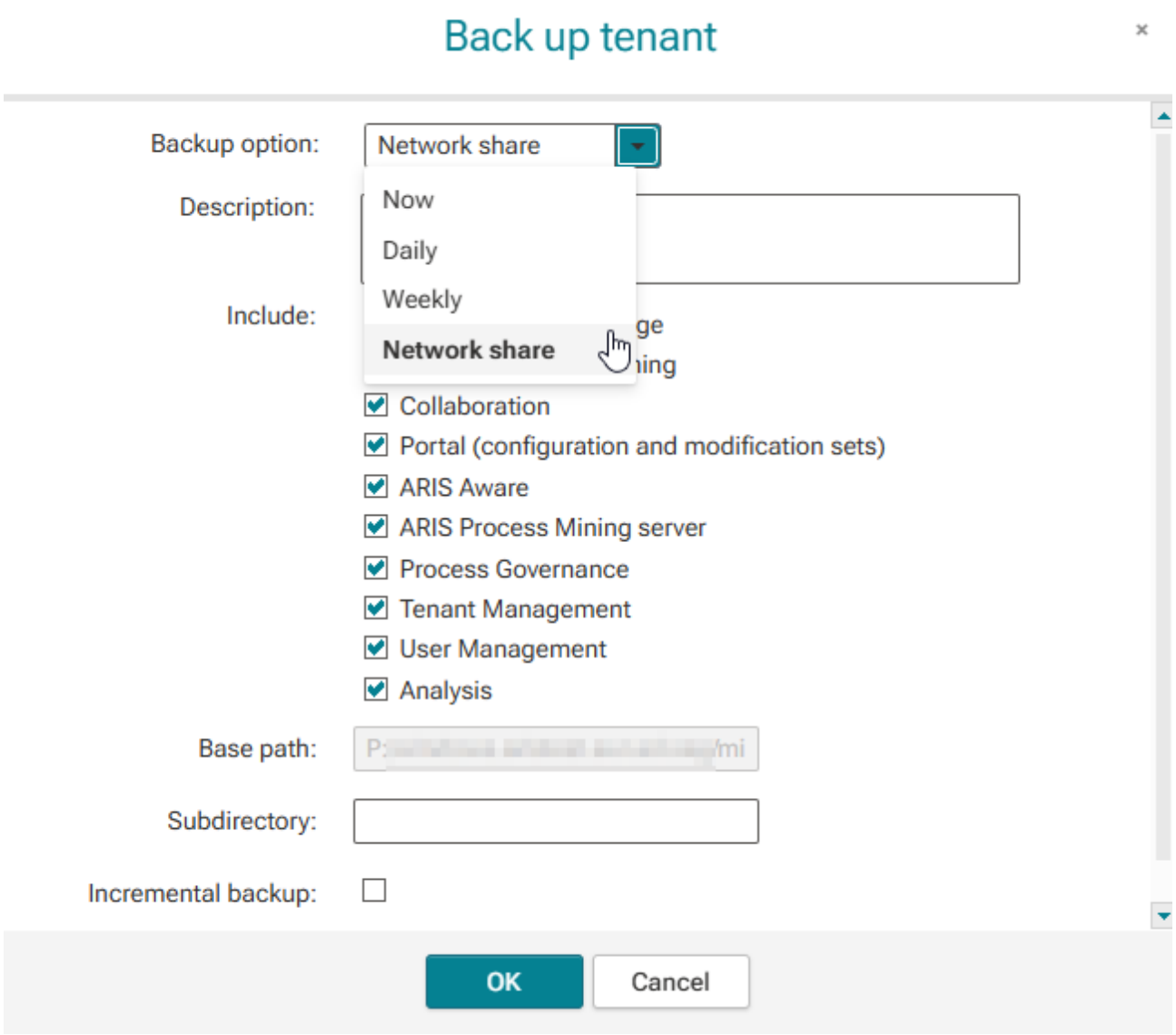
At the time of this writing, D2S mode is only supported for the ARIS Process Mining application, in future versions, more applications might support this approach.

A downside of D2S is that you do not get a single, easily portable backup file - so moving a backup created with D2S from the **tmdata** share (or downloading it via the TM UI) is not supported. This means that the safety of your backups hinges on the reliability and high-availability of the file server serving the **tmdata** share.

If you want to keep “offline” copies of your D2S backups (that is., backups that are not created on the **tmdata** share), you have the option to either set up a backup solution of the **tmdata** share itself, or doing a usual backup from time to time, which gives you a self-contained and thus fully portable backup file.

### 11.3.1. D2S backup and restore via the UI

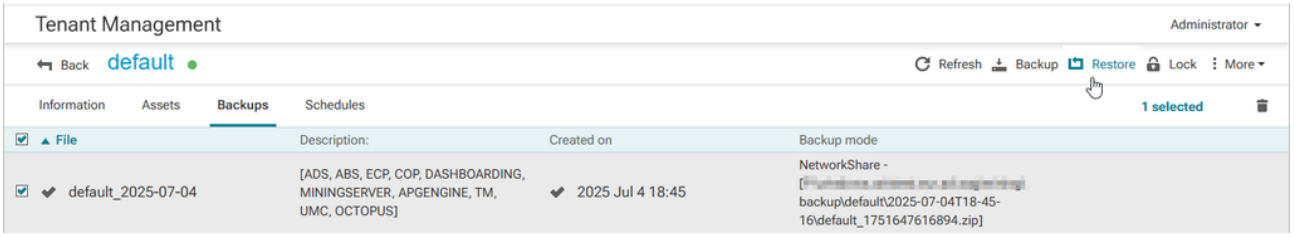
To perform a direct-2-network share backup with the UI, select **Network share** from the **Backup options** drop-down menu in the **Back up tenant** dialog, as shown in the screenshot below.



In the non-editable **Base path** field, you see the directory that you configured with the **JAVA-Dtm.backup.folder** parameter on the **UMCAdmin** runnable above. You can optionally select to put you backup into a subdirectory.

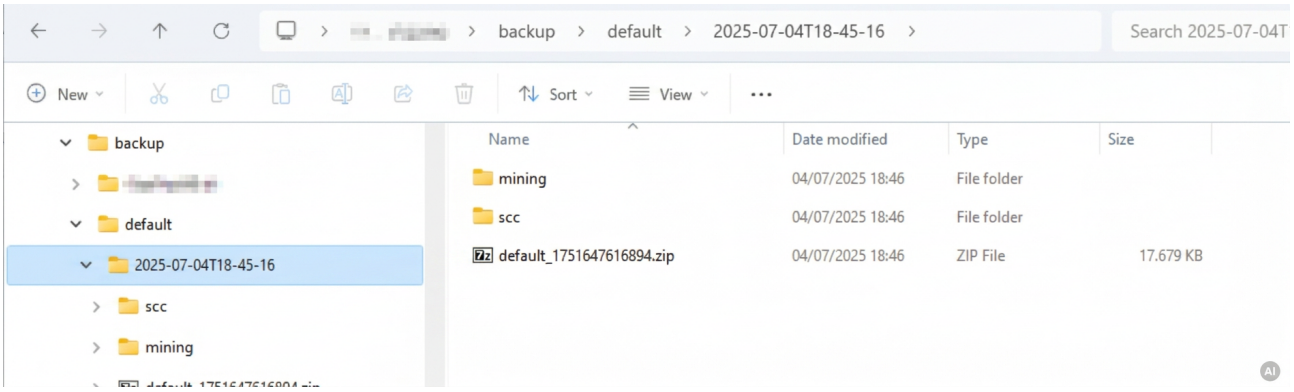
Click **OK** to start the backup. TM displays a list of backups available for that tenant, as displayed in the screenshot below. Ensure that the backup is of the type **NetworkShare** that is displayed in the **Backup mode** column.

**Example**




The **Backup mode** column also displays the path to the main backup file - for purely informational purposes. You can navigate to this folder on your share. Next to the main backup file you find a subdirectory containing the associated backup files with the Process Mining data, as shown in the screenshot below:

**Example 6.**




Ensure that the backup file is located in a subdirectory of the tenant, containing another subdirectory whose name is a timestamp. This keeps individual backups clearly separated from each other.

Restoring a backup via the UI also happens through the list of backups - just select it and click the **Restore** button.

If object-centric datasets exist, a subfolder named “scc” is created. It contains the source tables and the process data.

## 11.3.2. D2S backup and restore via the TM CLI

To perform a backup in the **direct-2-network share (D2S)** mode with the CLI, the basic structure of the command line is the following:

```
./tm.sh -a networkShareBackup -s <ARIS_Server_URL> -u <username> -p <password>
-t <tenant> -basePath <pathToTmDataShare> -path <subDirectory>
```

The **<ARIS_Server_URL>**, **<username>**, **<password>**, and **<tenant>** placeholders are the same as before. The optional **-path** parameter allows to put the backup into a subdirectory relatively to the root of the **tmdata** share.

An example command that performs a backup of the **default** tenant in **D2S** mode into a user-selected subdirectory **mySubdirectory** relative to the root of the **tmdata** share (which we assume is mounted to S:\tmdata as in the example above) would look like this:

```
tm -a networkShareBackup -s https://myaris.mycompany.com/ -u system -p manager
-t default -path mySubdirectory -basePath S:/tmdata
```

To restore a **D2S** backup via the CLI, the basic structure of the command is the following:

```
tm -a networkShareRestore -s <ARIS_Server_URL> -u <username> -p <password> -t
<tenant> -basePath <pathToTmDataShare> -f <fullPathToMainBackupFile>
```

### Example 7.

The ZIP file of a backup is located on the **tmdata** share at

s:\tmdata\mySubdirectory\default\2025-07-08T09-53-23\default_2025-07-08T09-53-23.zip

The command to restore the backup is

```
tm.bat -a networkShareRestore -s https://myaris.mycompany.com/ -u system -p
manager -t de
```

**Note**

In addition to the **networkShareBackup** and **networkShareRestore** actions, the TM CLI also offers the **partialNetworkShareBackup** and **partialNetworkShareRestore** actions. They allow to explicitly select the applications that should participate in the backup or restore operation by listing the applications as CSV with the **-app** switch. This makes them the “CLI equivalent” of de-selecting some of the applications via check boxes in the UI. Unless you have very specific reasons to only backup data for one or few specific applications, use the **networkShareBackup** and **networkShareRestore** commands to do full backups or restores.

## 11.3.3. Incremental D2S backups

Aside from improved efficiency, an additional benefit of the **D2S backup** mode is that it allows applications to support incremental backups. If a destination directory is used that already contains a previous backup (or multiple previous backups), an application with support for D2S and support for incremental backups finds the files of the previous backup(s) and only add the data that has been added or changed (and add deletion information for data that has been deleted) since the last backup was done.

**Note**

While incremental backups help to further speed up the backup procedure, it must be noted that in order to restore an incremental backup, the files of all previous backup down to the first (non-incremental) backup must be available and intact. While file corruption is rare, it can still happen. It is therefore important to combine the efficiency gains by using regular (that is, daily) incremental backups with regular full backups (that is, once a week).

The destination directory (which you can control with the **subdirectory** field on the UI or the **-path** parameter on the CLI) used for an incremental backup is an important aspect to control the incremental semantics. The

first time a new folder is used for a backup, this backup is implicitly a full backup, as no data is there that can be “added to”. Only if the same folder is used again for the next backup, an actual incremental backup is performed and only the changes are added there. If you later change the folder again, you perform again a backup. An example makes this more clear:

### Example 8.

Assume you want to have incremental backups during the week, but a full backup at the beginning of every calendar week. Just determine the calendar week of the current day and use a subdirectory following this pattern for your backups:

<YEAR>-CW<CALENDARWEEK>

**Example**

### Example 9.

2025-CW30

2025-CW31

2025-CW32

...

Assuming daily backups, with a new calendar week starting each Monday (in most locales), this will give you a full backup on each Monday, and an incremental backup on all other days of the week. If you prefer full backups on weekends, just offset your current date accordingly (that is, add one day to the actual date and the calendar-week-part of the directory name changes already on Sundays).

To perform an incremental backup on the UI, select the **Incremental backup** option in the **Back up tenant** dialog, respectively add the **-incrementalBackup true** parameter to the TM CLI’s command line.

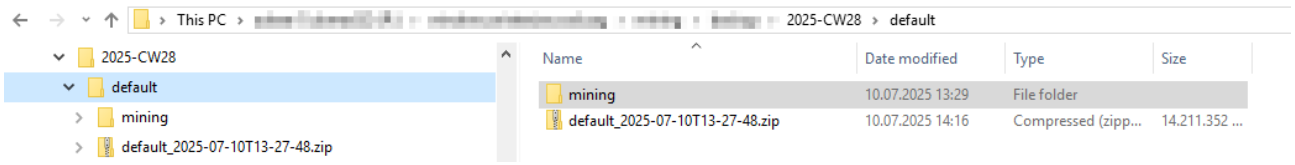
To make the (non-incremental) backup in our example above incremental, modify the command by adding this new switch and use the **-path** parameter to select a subdirectory following the **<YEAR>-CW<CALENDARWEEK>** pattern we just discussed (here assuming we run the backup in CW 28), the resulting command look like this:

```
tm -a networkShareBackup -s https://myaris.mycompany.com/ -u system -p manager
-t default -path 2025-CW28 -basePath S:/tmdata -incrementalBackup true
```

### Example 10.

After running this command, open the **tmdata** network share to view the resulting directory structure and content:

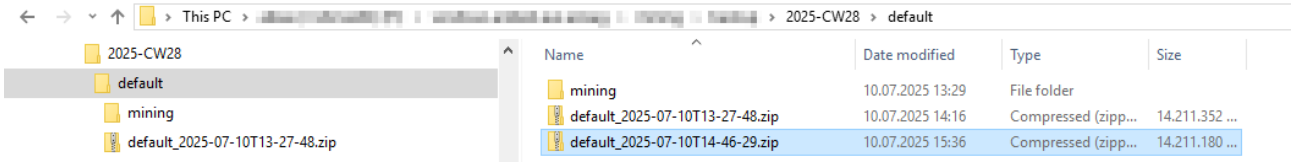
The screenshot shows the folder structure after running the first backup in incremental mode for a given subdirectory.




The selected **default** subfolder (as before in non-incremental mode) contains an automatically generated folder with the tenant name. However, different from non-incremental backups, there is no subdirectory whose name is a timestamp. The tenant-named subfolder contains the familiar “main” backup file and the **mining** directory with backup data from the D2S-enabled Process Mining application.

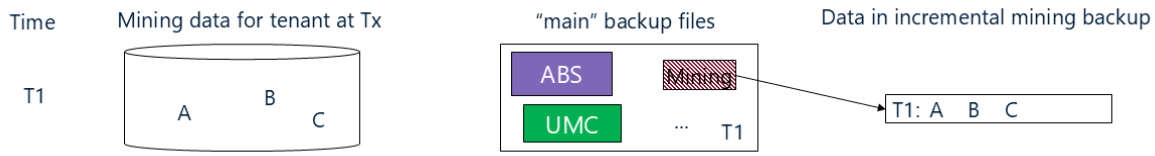
If you run the same backup command again, and assuming your tenant already contains some Process Mining data and no significant changes have been made since the last backup, you notice that the process completes much faster. This is because the Process Mining application now only stores the added or modified data, along with deletion information for any removed data (collectively referred to as the “delta”), in the **mining** subdirectory.

Further, a new “main” ZIP file is created in the **default** folder.



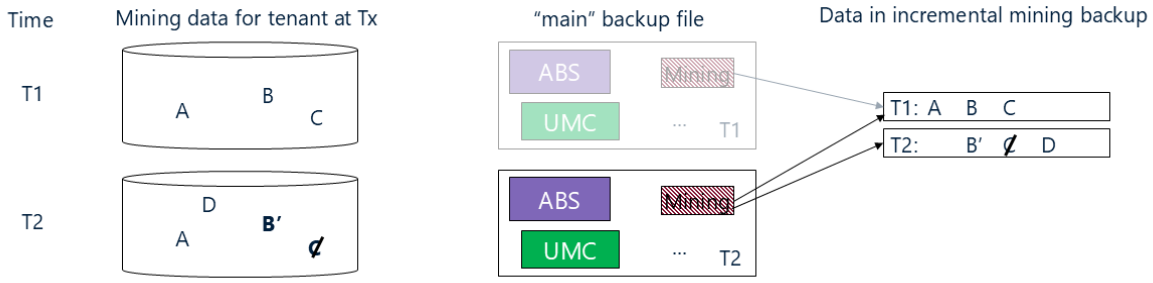

Since this ZIP file contains all the data of applications that do not support D2S (and are therefore also not capable of doing incremental backups), this is effectively a full backup of all NON-D2S applications. The “main” ZIP file is a representative of that particular backup. When restoring this backup, this is the file you must reference with the -f command line switch. Note that the main ZIP file also contains a small piece of metadata for the Process Mining part of the backup. This allows Process Mining to separate the data of possibly multiple incremental backups (which all end up in the **mining** subdirectory). And when restoring an incremental backup for a given point in time T only restores the state of Process Mining data to the point in time that particular backup was done, that is, any changes contained in subsequent incremental backups done into the same destination folder will not be restored.

The following figures illustrate the backup and restore semantics for incremental backups with an abstracted example. We use letters to represent some “data items” in the Process Mining application (that is, Process Mining data sets).

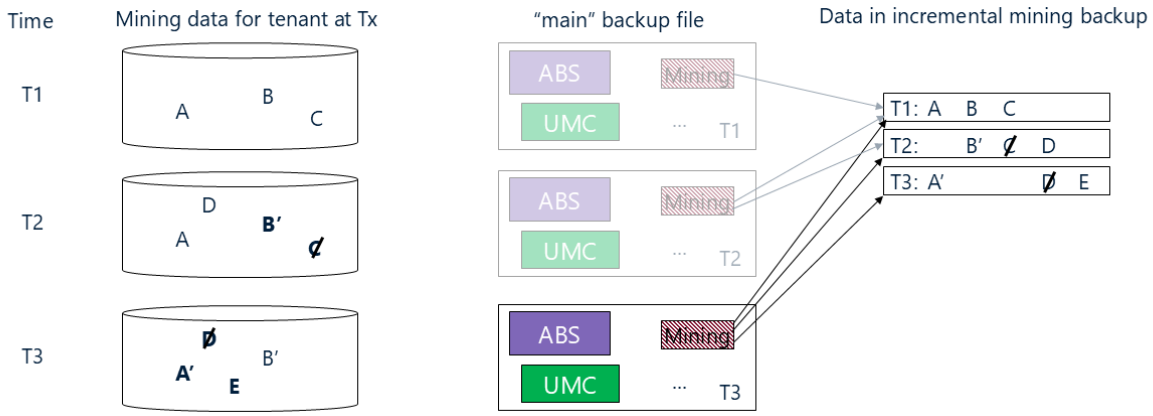



Initially, the Process Mining data set contains data items A, B, and C. The first incremental backup at T1 is effectively a full backup and thus contains A, B , and C. The “main” backup file - being not incremental - contains a full backup of the data of all non-D2S applications, plus a metadata file for any application, which supports D2S (that is, currently only Process Mining), which references the parts of the incremental Process Mining backup belong to this backup.

Afterwards, the users work with the system, delete the data item C, change B to B', and add a new data item D. The second incremental backup at T2 contains the changed B', the new D, and deletion info for C. Again we have a “main” backup file with a full backup for all other applications, and a file referencing the (now two) parts of the incremental backup of the Process Mining data, the one created at T1 and the one created with the new backup.




Further changes are made by the users, modifying A to A', deleting D, and adding E. The third incremental backup again only contains the changed parts of the Process Mining data, the "main" backup file is again a full backup of all data and a metadata file referencing the new three parts of the Process Mining incremental backups:




Restoring an incremental D2S backup works in the same way as a non-incremental D2S backup - just reference the main backup file. For example, if you restore the main backup file created at T2, the Process Mining application contains the original data items A and D and the changed data item B'.

# 12. Support and legal information

This section provides you with some general information regarding product support and legal aspects.

## 12.1. Supported Web browsers

The following Web browsers are currently supported by ARIS Process Mining.

| Browser | Version |
|---|---|
| Microsoft Edge | recommended Chromium-based version 128 or higher |
| Google Chrome | recommended Google Chrome version 129 or higher |
| Safari | recommended Safari version 15 or higher |

## 12.2. Documentation scope

The information provided describes the settings and features as they were at the time of publishing. Since documentation and software are subject to different production cycles, the description of settings and features may differ from actual settings and features. Information about discrepancies is provided in the Release Notes that accompany the product. Please read the Release Notes and take the information into account when installing, setting up, and using the product.

If you want to install technical and/or business system functions without using the consulting services provided by SAG ARIS GmbH, you require extensive knowledge of the system to be installed, its intended purpose, the target systems, and their various dependencies. Due to the number of platforms and interdependent hardware and software configurations, we can describe only specific installations. It is not possible to document all settings and dependencies.

When you combine various technologies, please observe the manufacturers' instructions, particularly announcements concerning releases on their Internet pages. We cannot guarantee proper functioning and installation of approved third-party systems and do not support them. Always follow the instructions provided in the installation manuals of the relevant manufacturers. If you experience difficulties, please contact the relevant manufacturer.

If you need help installing third-party systems, contact your local ARIS sales organization. Please note that this type of manufacturer-specific or customer-specific customization is not covered by the standard ARIS software maintenance agreement and can be performed only on special request and agreement.

## 12.3. Restrictions

ARIS products are intended and developed for use by persons. Automated processes, such as the generation of content and the import of objects/artifacts via interfaces, can lead to an outsized amount of data, and their execution may exceed processing capacities and physical limits. For example, processing capacities are exceeded if an extremely high number of processing operations is started simultaneously. Physical limits may be exceeded if the memory available is not sufficient for the execution of operations or the storage of data.

Proper operation of ARIS products requires the availability of a reliable and fast network connection. Networks with insufficient response time will reduce system performance and may cause timeouts.

## 12.4. Support

### Product support

We provide support for ARIS products to all customers with a valid support contract.

Information on product version availability and support, as well as sustained support dates, is available at ARIS Community.

Contact our Global Support services at ARIS Community to raise and update support incidents, or post ideas and feature requests.

### Community

Register with ARIS Community to download products, updates and fixes, find expert information, and interact with other users.

### Documentation

For information on how to use and activate products and for other technical support questions, visit the ARIS Documentation website.

If you want to give feedback on any documentation topic, write to documentation@aris.com.

## 12.5. Legal notices

This document applies to ARIS Process Mining Version 10 and to all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © 2020-2026 SAG Aris GmbH, Saarbrücken, Germany and/or its subsidiaries and/or its affiliates and/or their licensors.

The name ARIS and all SAG Aris GmbH product names are either trademarks or registered trademarks of SAG Aris GmbH and/or its subsidiaries and/or its affiliates and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners.

The Software may contain or be distributed with certain commercial third party components (“Third Party Components”), which may be subject to special license terms and conditions (“Third Party Terms”) located at https://aris.com/terms.

Third Party Terms include important licensing and warranty information and disclaimers of third party licensors. For the avoidance of doubt, if Customer uses the Software in accordance with the Documentation,

the Third Party Terms do not restrict the rights granted under this Agreement. In the event that a third party manufacturer no longer provides active support for any Third Party Component, Supplier will use commercially reasonable efforts to respond to any issues with the Software related to the Third Party Components.

Use, reproduction, transfer, publication or disclosure is prohibited except as specifically provided for in your License Agreement with SAG Aris GmbH and/or its subsidiaries and/or its affiliates and/or their licensors.