Arcsys Installation

Arcsys Installation Guide

25.3.2.STS
March 23, 2026

Version: 25.3.2.STS

Publication date of the document: 2026-03-23

This document is intended for anyone who requires more information on the Arcsys product.

This document is written for all owners of a valid license.

The reader agrees to respect the confidential nature of this document.

Contact details:

France	***Germany***	***USA***
INFOTEL SA *Le Valmy, 6/8/18 Avenue Léon Gaumont* *F-75020 Paris* *France*	*Insoft Infotel Software GmbH* *Sternstr. 9-11* *D-40479 Düsseldorf* *Deutschland*	*INFOTEL Corporation* *PO Box 47517 Florida* *33743 St Petersburg* *United States*
+33 (0)1 48 97 38 38	*+49 (0) 211 44 03 16-6*	*800 543 1982 – Toll-free telephone (US only)* *+1 727 343 5958*
https://techsupport.infotel.com `<software@infotel.com>`	https://techsupport.insoft-software.com `<software@insoft-infotel.com>`	https://techsupport.infotel-consulting.co.uk `<software@infotel.com>`

Table of Contents

List of Tables

Preface

1. Introduction

This document describes how to install and update the Arcsys product.

2. Reference Documents

2.1. Concepts

Arcsys Presentation Manual: **Arcsys-presentation-25.3.2.STS-en.pdf**

Arcsys Functional Description Manual: **Arcsys-functional-description-25.3.2.STS-en.pdf**

2.2. Installing and Updating

Arcsys Prerequisites Manual: **Arcsys-requirements-25.3.2.STS-en.pdf**

Arcsys Installation Manual: **Arcsys-installation-25.3.2.STS-en.pdf**

2.3. Operations

Arcsys Administration Manual: **Arcsys-administration-25.3.2.STS-en.pdf**

Arcsys Errors Manual: **Arcsys-error-25.3.2.STS-en.pdf**

2.4. GUI

Arcsys Web Interface User Manual: **Arcsys-web-25.3.2.STS-en.pdf**

Interface Guide: **Arcsys-web-end-user-25.3.2.STS-en.pdf**

2.5. Development

Arcsys API Manual: **Arcsys-api-25.3.2.STS-en.pdf**

2.6. Option guides

ArcHP Option Guide: **Arcsys-option-archp-25.3.2.STS-en.pdf**

ArcREF Option Guide: **Arcsys-option-arcref-25.3.2.STS-en.pdf**

2.7. Optional modules

BatchReporting: **BatchReporting-UserGuide-25.3.2.STS-en.pdf**

ClassAssigner: **ClassAssigner-UserGuide-25.3.2.STS-en.pdf**

MetadataReplacement: **MetadataReplacement-UserGuide-25.3.2.STS-en.pdf**

StartRetentionDateAssigner: **StartRetentionDateAssigner-UserGuide-25.3.2.STS-en.pdf**

3. Symbols and Meanings

Note

Identifies information of particular interest

Important

Identifies important information

4. Definitions and Abbreviations

See the Glossary

Part 1. Introduction

1. Prior Knowledge

1.1. Arcsys Installation and Update Package

1.1.1. Description

As outlined in Arcsys Prerequisites Manual, Arcsys is made up of the **Arcsys core**, **optional Arcsys modules associated with the core,**and **connectors and other optional modules**.

The **Arcsys Installation and Update Package** is a compressed file called `Arcsys-<Version>.tar.gz` which is retrieved with each minor or major version of Arcsys. It includes in a folder called `ArcsysPackage`:

- for major versions:
  - all the Arcsys core modules and optional Arcsys modules in their latest version at the time of retrieval;
  - an `ArcsysInstaller` script is used to start the installation of each of these modules;
  - an `ArcsysUpdater` script is used to start the update of each module.
- for minor versions:
  - the Arcsys core modules and the optional Arcsys modules are updated in the minor version;
  - an `ArcsysUpdater` script is used to start the update of each module retrieved.

1.2. Installing and Updating A Given Version

Installing or updating Arcsys to a given major version (e.g. 5.0.0.0) requires you have the **Arcsys Installation and Update Package for this major version** and apply the installation or update procedure to this major version.

Installing or updating Arcsys to a given minor version (e.g. 5.0.1) requires you have the **Arcsys Installation and Update Package for the preceding major version** (in this case, 5.0.0.0), install or update to this version, then update all the components affected by the minor version using the **Arcsys Installation and Update Package of the minor version**.

2. Requirements Check

Before installing or updating Arcsys, you must first:

- Check that the hardware and software requirements specified in Arcsys Prerequisites Manual are met. An Arcsys update can result in changes in the requirements. For example, the JVM version or the JEE application server version required may increase.
- For installation, proceed with the prior configuration of the following external dependencies, also described in Arcsys Prerequisites Manual:
  - LDAP directory (mandatory);
  - Database (mandatory).

Note

If these conditions are not met, the operation of Arcsys software cannot be guaranteed and/or the installation or update process cannot be completed.

3. Deprecated features status

As any product, a feature in Arcsys may become "deprecated" in a major version, generally because it is replaced by another feature. A deprecated feature stays first in a "Deprecated" status, then changes to a "Removed" status.

- During its "Deprecated" status, the feature is still maintained and bug fixes are performed. However, the support is limited. In the paragraph describing the feature, the following note is visible: "This feature is deprecated from version *x*."
- When the feature is in "Removed" status, the feature is not maintained anymore and the feature becomes undocumented.

3.1. Deprecated features list

The following table lists for each deprecated feature the version from which it is deprecated, the feature to use instead (if available) and the version in which the feature has been or will be completely removed from the product.

Note

The feature may finally be removed *after* this target version, but it will never be removed *before* the announced version.

Feature name	Deprecated from version	Reason for deprecation	Removed from version	Feature to use instead
RMI and SOAP API	5.1.2.0	Technological deprecation	Not planned	Arcsys REST API
Active and inactive status of repositories, collections, archives	5.2.0.0	Functional deprecation	2024.1.LTS (already removed in STS branch)	Not replaced
ArcDATA Option to archive and retrieve databases in Arcsys	5.2.0.0	Rarely or never used feature	2024.1.LTS	Client development using APIs.
End user screens in Arcsys Web Agent	5.2.0.0	Technological deprecation	Not planned	Progressively replaced by ArcWeb
"TAR equivalent" flag in collection set to false	5.2.0.0	Rarely or never used feature	2024.1.LTS (already removed in STS branch)	By default, all collections will be "Tar equivalent".
Number of copies field on a storage policy	5.2.0.0	Rarely or never used feature	2024.1.LTS	Not replaced
EAD for metadata	5.2.0.0	Rarely or never used feature	2024.1.LTS	Not replaced

Feature name	Deprecated from version	Reason for deprecation	Removed from version	Feature to use instead
Workflow	This feature was directly removed without having the "deprecated" status because it is not used in production.	Rarely or never used feature	2024.1.LTS (already removed in STS branch)	Handling of pre-disposal workflow, available from 2024.1.LTS
v1 and v2 of the REST API	2024.1.LTS	Technological deprecation	2027.1.LTS. In STS branch: 25.2.STS	Further versions of the REST API must be used instead
ACSLS	This feature was directly removed without having the "deprecated" status because it is not used in production.	Rarely or never used feature	2024.1.LTS (already removed in STS branch)	SCSI
Volumetrics graphs in Arcsys Web Agent	This feature was directly removed without having the "deprecated" status because it is not used in production and would cause compatibility issues.	Rarely or never used feature + technical incompatibility	2024.1.LTS (already removed in STS branch)	Raw data of the volumetrics screens, or BatchReporting
Automatic creation of lot copy requests when adding a new pool to a storage policy	This feature was directly removed without having the "deprecated" status.	Poor performance of requests creation	2024.1.LTS (already removed in STS branch)	CopyRequestManager
ARCH_REPOSITORY_DIR and *REST_REPOSITORY_DIR* parameters of the Arcsys Application Agent	This feature was directly removed without having the "deprecated" status.	Rarely or never used feature	2024.1.LTS (already removed in STS branch)	*ARCH_REPOSITORY_PATH* and *REST_REPOSITORY_PATH* can be used for current architecture needs
Forcing migrations by changing the start retention date with	This behaviour was directly removed	The ability to set a new retention start date was abnormal if	2024.1.LTS (already	The correct approach to manage media

Feature name	Deprecated from version	Reason for deprecation	Removed from version	Feature to use instead
a retention profile with start retention trigger=archiving date or start retention trigger=event code	without having the "deprecated" status.	there was a retention profile that wouldn't take into account the retention start date. It was fixed by #72347 fix in 2024.1.LTS.	removed in STS branch)	migrations is to modify the pools delay.
Full text field	This feature was directly removed without having the "deprecated" status.	Rarely or never used feature	2024.1.LTS	Full text in web interface is only supported in ArcWeb Module
NetBackup and TSM media managers	This feature was directly removed without having the "deprecated" status.	Rarely or never used feature	2024.1.LTS	ArcMover
ADMIN role	2024.1 LTS	The ADMIN role is too generic and unclear, leading to inconsistencies in what is considered 'admin' across different clients. Instead, roles should be defined based on specific functionalities. This is why the ADMIN role is being phased out and progressively replaced by more specific and meaningful roles.	2027.1.LTS	Other existing roles
Removal of private key with INTERNAL encryption protocol / Removal of genkey binary	This feature was directly removed without having the "deprecated" status.	Rarely or never used feature	2024.1.LTS	SSL encryption protocol
CMX	This feature was directly removed without having the "deprecated" status because	Rarely or never used feature	25.1.STS	S3

Feature name	Deprecated from version	Reason for deprecation	Removed from version	Feature to use instead
	it is not used in production.			
Removal of the support of signature version 2 for Cloud S3 storage	This feature was directly removed without having the "deprecated" status.	Rarely or never used feature	25.3.STS	Signature version 4

Table 3.1. Deprecated features

4. Before Installing or Updating

4.1. Installing the Package on the Hosts

- Copy the Arcsys Installation and Update Package on each of the hosts where at least one Arcsys module is to be installed or updated. Note: to update the Arcsys Database, the updater must be installed on the machine hosting the Arcsys Engine.

Note

This package can be located in a temporary directory. It can be deleted at the end of the installation procedure. You should nonetheless save or archive the `INSTALL.properties` configuration file for any subsequent reinstallation procedure.

4.2. Decompressing the Package

The Arcsys Installation and Update Package is retrieved as a compressed file. It must be decompressed before it can be run.

- On UNIX:

```
gzip -d Arcsys-<Version>.tar.gz
tar -xvf Arcsys-<Version>.tar
```

- On Windows: Use your preferred utility (for example, 7zip) to decompress the file `Arcsys-<Version>.tar.gz`

4.3. License

4.3.1. Getting a license file

In order to use Arcsys, you must have a valid license file. Infotel can provide you with a license file through the Tech Support Portal.

The process to obtain a license file may involve running a small program to gather some technical information about the server that will run the product. This is the case when the license involves the identification of one or more machines.

For a Try And Buy, a Full, or a Rental license without machine identification, this step is not necessary.

4.3.2. License verification and associated errors

At launch and every day at 5 AM, the Arcsys Engine checks the license file and raises an error if it cannot verify it. The license file used is the one configured in the -

Darcsys.license.file parameter of the Arcsys Engine launch script. The path to this file is defined during the installation of Arcsys, or during the first update to 5.2 version or later (See Arcsys Installation Manual).

A license verification error consists of a message in the log file and a standard error message in the console.

When a license verification error occurs at Arcsys startup, this error is raised during the Arcsys Engine launch. When the license expires while Arcsys is running, the error occurs during the daily check at 5 AM. In this case, the Arcsys Engine ends proceeding the requests that were already loaded but does not accept any new request and stops.

A license verification error may result from the following causes:

- the license is invalid
- the license file is not found
- the license is expired
- the signature of the license is not correct
- the version of the license file is not correct

4.3.3. Verifying a license file

You can verify a license at any time by using `verify-license-arcsys.sh` (or .cmd) script.

The script is used as follows: `verify-license-arcsys.sh` (or .cmd) `<path to license file>`

When run, the `verify-license-arcsys` scipts displays the status of the license, or an error message. The resulting message can be one of those:

- `HIDI486I THE LICENSE FOUND FOR THE PRODUCT IS A VALID ONE (.. DAYS BEFORE EXPIRATION).`
- `HIDI483E THE LICENSE FOR THE PRODUCT HAS EXPIRED SINCE .. DAYS.`
- `HIDI445W WARNING: YOU ARE STILL USING A TEMPORARY LICENSE. YOU NEED TO ENROLL THE PRODUCT WITHIN .. DAYS.`
- `HIDI446E YOUR TRY AND BUY LICENSE IS DEFINITELY EXPIRED. PLEASE ENROLL THE PRODUCT INSTALLING A FULL LICENSE.`
- `HIDI485E THERE IS NO LICENSE FILE FOUND.`: you may encounter this error when the license file you specified is not found or cannot be read

- `HIDI443E NO LICENSE HAS BEEN FOUND FOR THE PRODUCT`: you may encounter this error when the license file you specified is relative to another machine than the one on which you run the `verify-license-arcsys` script
- `HIDI484E AN ERROR HAS OCCURED DURING THE LICENSE VERIFICATION PROCESS.`: you may encounter this error when the license file specified has been altered, when the signature of the license file is incorrect, or when the version of the license file is incorrect

4.3.4. ArcPAK Option

ArcPAK Option allows to use record compression on ArcMover and native ingestion of compressed files.

The license used must have the ArcPAK Option enabled to:

- Archive native ZIP files
- Write to zones where compression is enabled. This applies to archiving, migration, and copy requests.

If the ArcPAK Option is not enabled:

- archiving, migration, and copy requests to zones where compression in enabled will fail,
- if there are zones with compression enabled attached to a storage profile, all these zones will be logged at WARN level in the engine logs at startup.

5. How to encrypt passwords in Arcsys configuration files

5.1. Generate key file

In order to encrypt the data, it is necessary to generate first a key file with the `generate-key` script present in the Arcsys installer. This script uses a cryptographically secured pseudo-random number generator to create a random 256 bit symmetric key. The key file is used by Arcsys to decrypt the encrypted passwords in the configuration files of different modules. This key file may be shared across modules (as it must be different each time, as required by your security policy).

The following GNU/Linux command generates a file (filename: `secret-key`) in the folder `myFolder`:

```
$ ./generate-key.sh -s -d myFolder
Secret key generated in file myFolder/secret-key
```

5.2. Encrypt a value

The passwords must be encrypted with the generated symmetric key file. All properties can be encrypted too.

The command for using the *encrypt* utility depends on the platform chosen:

- Windows: `bin\encrypt.cmd`
- GNU/Linux: `bin/encrypt.sh`

The following GNU/Linux command encrypts the password `myPassword` with the `secret-key` file:

```
$ ./encrypt.sh -k myFolder/secret-key -e
Please enter the value to be encrypted.

Please enter the value again.

ENC(fRO8S6BlhZY5Ci3Rn8ZJgOXWkwD5jYgaYa3QWBsV52Q=)
```

The blank line corresponds to the entry of the string to encrypt.

Note

If the option `-k myFolder/secret-key` is not specified, the default key stored in `etc/secret-key` is used.

The encrypted values (including "`ENC(`" and "`)`") of the password must then be copied in the configuration file.

The encrypted properties are *salted*, in the cryptographic sense, and associated with the key present in the `secret-key` file. If this key is renewed, all the properties encrypted using a previous key must be re-encrypted manually.

5.3. During installation

The key file must be specified to the installer thanks to the property `SECRET_KEY_PATH` when launching the Arcsys installer (this is detailed in the following "Running the installation module" parts).

Important

This module is installed in the root folder of Arcsys Engine. It may be used to encrypt any value; not only passwords; of all Arcsys modules except Arcsys Transfer Server and Arcsys Transfer Service. When a parameter value beginning with `ENC(` is encountered, Arcsys automatically decrypts it.

Part 2. Installation

1. Introduction

1.1. Installation procedure

Install an Arcsys module by:

- Completing the appropriate sections in the file `INSTALL.properties`. This file is located in the root of the Arcsys Installation and Update Package directory. It should be noted that some sections are shared by a number of modules. To be able to write passwords in the `INSTALL.properties` file, a key file must be generated and a tool called `encrypt` must be used (see « How to encrypt passwords in Arcsys configuration files », page 11).
- Then launching the `ArcsysInstaller` installation script contained in the `ArcsysPackage` Arcsys Installation and Update Package folder with the required options.
- Finally proceeding with a final installation phase if applicable.

1.2. Installation types

The installer allows you to choose the components you wish to install. It can be a preconfigured set of modules or a customized set of modules, and even a single module.

Important

In the case of the installation of a set of modules, the configuration required for the installation of each included module must have been performed beforehand. These configurations are described below in Chapter 2, INSTALL.properties File.

1.2.1. Preconfigured set of module installation

- **Full installation**: All the modules are installed simultaneously. The `COMPONENTS` value of the installation script must be set to `fullInstall` (see « Launching installation », page 53)
- **Minimum server installation**: Only part of the modules are installed. This installation includes **the mandatory server "Arcsys core" modules**.

  The `COMPONENTS` value of the installation script must be set to `minimumServerInstall` (see « Launching installation », page 53).
- **Client installation**: Only part of the modules are installed. This installation includes **the mandatory client "Arcsys core" modules**.

The `COMPONENTS` value of the installation script must be set to `clientInstall` (see « Launching installation », page 53).

The following table summarizes the modules included in each of these partial installation types:

Name of the module	Minimum server installation?	Client installation?
Arcsys core modules		
Arcsys Database	yes	no
Arcsys RMI, TCP/IP and SOAP API	yes	no
Arcsys REST API	yes	no
Arcsys Transfer Server	yes	no
Arcsys Transfer Service	no	yes
Arcsys Engine	yes	no
Arcsys Web Agent	no	no
Arcsys Application Agent	yes	yes
Arcsys Auto-Archive Agent	yes	yes
Optional Arcsys modules associated with the core		
ArcFF format control module	yes	no
CopyRequestManager	yes	no
Arcsys standard Clients	no	no
ArcsysFsComparator File systems comparator	no	no
ArcsysBatchs batch module	no	no
BatchReporting	no	no
MetadataReplacement	no	no
StartRetentionDateAssigner	no	no
ClassAssigner	no	no
Signature validator module	no	no
ArcREF Injector	no	no
ReportingInitializer	no	no
ArcCLI	no	no

Table 1.1. Modules and installation Types

1.2.2. Customized set of module installation

The set of modules to be installed can also be customized. In this case, the `COMPONENTS` value of the installation script must be set to a list of modules to install among:

- `api`
- `restWSApi`

- `applicationAgent`
- `database`
- `engine`
- `transferServer`
- `transferService`
- `webAgent`
- `autoarchiveAgent`
- `copyRequestsManager`
- `fileFormatServer`
- `fsComparator`
- `standardClients`
- `batchs`
- `batchReporting`
- `classAssigner`
- `metadataReplacement`
- `startRetentionDateAssigner`
- `signatureValidator`
- `refInjector`
- `reportingInitializer`
- `cli`

This list can contain a single module or a set of modules of your choice (see « Launching installation », page 53).

1.3. Installation requirements

1.3.1. Before installation: Requirements for starting up the database

Important

If your chosen installation includes database installation, the database must be started up and the installation module must be able to access it.

1.3.2. Before installation: requirement for starting up the application server for JBoss / WildFly

Important

When deploying in automatic mode, the application server must be started up. After installation, the server will be stopped. If you do not want this, you must deploy it manually.

1.3.3. Before installation: requirement for starting up the application server for Apache Tomcat

Important

When deploying in automatic mode, the application server may be started up or not. However, if it started up, you must restart it after the deployment.

2. INSTALL.properties File

Configuration file for the installation of Arcsys.

2.1. Determining the sections to fill

The following table outlines the various installation options available (corresponding to the potential values of the `COMPONENTS` parameter in the installation script). For each installation possibility, this table lists the sections of the `INSTALL.properties` file that need to be completed. Any other sections are ignored.

Value of `COMPONENTS` parameter	**Sections to fill**
Preconfigured set of module installation	
fullInstall	• Database common parameters (DBPARAMS section), page 21 • Common API parameters (APIPARAMS section), page 21 • Database JDBC common parameters (DBJDBCPARAMS section), page 22 • Common API clients parameters (APICLIENTSPARAMS section), page 23 • Common Agent clients parameters (AGENTCLIENTSPARAMS section), page 25 • LDAP authentication common parameters (LDAPPARAMS section), page 26 • Common REST API clients (RESTAPICLIENTSPARAMS parameters), page 30 • Database parameters (REFPARAMS section), page 32 • RMI and TCP/IP API parameters (RMI_TCPIP_APIPARAMS section), page 33 • REST API parameters (RESTAPIPARAMS section), page 35 • Transfer server parameters (TRSPARAMS section), page 35 • Engine parameters (ENGINEPARAMS section), page 37 • ArcFF parameters (ARCFFPARAMS section), page 42 • Application Agent parameters (APPAGENTPARAMS section), page 43 • Web agent generic parameters (WEBGENERICPARAMS section), page 46 • Web agent JEE specific parameters (WEBSPECIFICPARAMS subsection), page 47 • Web agent common automatic deployment parameters (AUTOWEBPARAMS subsection), page 48
minimumServerInstall	• Database common parameters (DBPARAMS section), page 21 • Common API parameters (APIPARAMS section), page 21

Value of `COMPONENTS` parameter	Sections to fill
	• Database JDBC common parameters (DBJDBCPARAMS section), page 22 • Common API clients parameters (APICLIENTSPARAMS section), page 23 • Common Agent clients parameters (AGENTCLIENTSPARAMS section), page 25 • LDAP authentication common parameters (LDAPPARAMS section), page 26 • Database parameters (REFPARAMS section), page 32 • RMI and TCP/IP API parameters (RMI_TCPIP_APIPARAMS section), page 33 • REST API parameters (RESTAPIPARAMS section), page 35 • Transfer server parameters (TRSPARAMS section), page 35 • Engine parameters (ENGINEPARAMS section), page 37 • ArcFF parameters (ARCFFPARAMS section), page 42 • Application Agent parameters (APPAGENTPARAMS section), page 43
clientInstall	• Common API clients parameters (APICLIENTSPARAMS section), page 23 • Common Agent clients parameters (AGENTCLIENTSPARAMS section), page 25 • Application Agent parameters (APPAGENTPARAMS section), page 43 • Transfer service parameters (TRSVCEPARAMS section), page 48
Module installation	
database	• Database common parameters (DBPARAMS section), page 21 • Database JDBC common parameters (DBJDBCPARAMS section), page 22 • Database parameters (REFPARAMS section), page 32
api	• Database common parameters (DBPARAMS section), page 21 • Common API parameters (APIPARAMS section), page 21 • Database JDBC common parameters (DBJDBCPARAMS section), page 22 • LDAP authentication common parameters (LDAPPARAMS section), page 26 • RMI and TCP/IP API parameters (RMI_TCPIP_APIPARAMS section), page 33
restWSApi	• Database common parameters (DBPARAMS section), page 21 • Common API parameters (APIPARAMS section), page 21 • Database JDBC common parameters (DBJDBCPARAMS section), page 22 • LDAP authentication common parameters (LDAPPARAMS section), page 26 • REST API parameters (RESTAPIPARAMS section), page 35

Value of `COMPONENTS` parameter	Sections to fill
engine	• Database common parameters (DBPARAMS section), page 21 • Database JDBC common parameters (DBJDBCPARAMS section), page 22 • LDAP authentication common parameters (LDAPPARAMS section), page 26 • Engine parameters (ENGINEPARAMS section), page 37
webAgent	• Database common parameters (DBPARAMS section), page 21 • Database JDBC common parameters (DBJDBCPARAMS section), page 22 • LDAP authentication common parameters (LDAPPARAMS section), page 26 • Web agent generic parameters (WEBGENERICPARAMS section), page 46 • Web agent JEE specific parameters (WEBSPECIFICPARAMS subsection), page 47 • Web agent common automatic deployment parameters (AUTOWEBPARAMS subsection), page 48
autoarchiveAgent	• Common API clients parameters (APICLIENTSPARAMS section), page 23 • Common Agent clients parameters (AGENTCLIENTSPARAMS section), page 25
standardClients	• Common API clients parameters (APICLIENTSPARAMS section), page 23 • Common Agent clients parameters (AGENTCLIENTSPARAMS section), page 25
fsComparator	• Common API clients parameters (APICLIENTSPARAMS section), page 23
batchs	• Common API clients parameters (APICLIENTSPARAMS section), page 23
copyRequestsManager	• Database common parameters (DBPARAMS section), page 21 • Database JDBC common parameters (DBJDBCPARAMS section), page 22
reportingInitializer	• Database common parameters (DBPARAMS section), page 21 • Database JDBC common parameters (DBJDBCPARAMS section), page 22
batchReporting	• Common REST API clients (RESTAPICLIENTSPARAMS parameters), page 30
cli	• Common REST API clients (RESTAPICLIENTSPARAMS parameters), page 30
classAssigner	• Common REST API clients (RESTAPICLIENTSPARAMS parameters), page 30
metadataReplacement	• Common REST API clients (RESTAPICLIENTSPARAMS parameters), page 30
startRetentionDateAssigner	• Common REST API clients (RESTAPICLIENTSPARAMS parameters), page 30
transferServer	• Transfer server parameters (TRSPARAMS section), page 35
transferService	• Transfer service parameters (TRSVCEPARAMS section), page 48
applicationAgent	• Application Agent parameters (APPAGENTPARAMS section), page 43
refInjector	No sections need to be filled in the `INSTALL.properties file`
signatureValidator	No sections need to be filled in the `INSTALL.properties file`

Table 2.1. Sections to fill for each installation type and for each module installation

2.2. Common parameters

2.2.1. Database common parameters (DBPARAMS section)

This section contains the common parameters for the database.

Configuration filename after the installation: `REFERENTIAL_JDBC.properties`

Note

This section requires completion if the value of `COMPONENTS` parameter of the installation script (`ArcsysInstaller.sh` or `ArcsysInstaller.bat`) is composed of at least one of these values: `minimumServerInstall`, `fullInstall`, `webAgent`, `engine`, `restWSApi`, `api`, `database`, `copyRequestsManager`, `reportingInitializer`

Parameter name in `INSTALL.properties`	**TYPEBASE**
Parameter type	State variable. Possible values: `Oracle`, `MariaDB`, `SQLServer`, `PostgreSQL`
Description	The type of database used for the Database among 'Oracle', 'MariaDB', 'SQLServer', 'PostgreSQL'. Note: Values are case-sensitive.
Shipped value	`Oracle`
Parameter name after installation	`arcsys.typebase`

2.2.2. Common API parameters (APIPARAMS section)

This section contains the common parameters for APIs.

Configuration filename after the installation: `ARCSYS_RMI_API.properties`

Note

This section requires completion if the value of `COMPONENTS` parameter of the installation script (`ArcsysInstaller.sh` or `ArcsysInstaller.bat`) is composed of at least one of these values: `minimumServerInstall`, `fullInstall`, `restWSApi`, `api`

Parameter name in `INSTALL.properties`	**SITE_CODE_FOR_API**
Parameter type	Characters string
Description	Site code to which the module must belong. This site must be created in the Web interface beforehand (by default, the installer creates systematically a PRIMARY_SITE site code)
Shipped value	`PRIMARY_SITE`

2.2.3. Database JDBC common parameters (DBJDBCPARAMS section)

This section contains the common JDBC parameters for the database.

Configuration filename after the installation: `REFERENTIAL_JDBC.properties`.

Note

This section requires completion if the value of `COMPONENTS` parameter of the installation script (`ArcsysInstaller.sh` or `ArcsysInstaller.bat`) is composed of at least one of these values: `minimumServerInstall`, `fullInstall`, `engine`, `restWSApi`, `api`, `database`, `webAgent`, `copyRequestsManager`, `reportingInitializer`

Parameter name in `INSTALL.properties`	**JDBC_URL**
Parameter type	URL
Description	The JDBC access URL to the relational database. Example(s): * Example for Oracle: `jdbc:oracle:thin:@host:1521:ARCSYS_DB` * Example for MariaDB: `jdbc:mariadb://host:3306/ARCSYS_DB` * Example for SQLServer: `jdbc:sqlserver://host:1433;instanceName=ARCSYS;databaseName=ARCSYS_DB`; * Example for PostgreSQL: `jdbc:postgresql://host:5432/arcsys_db`. For PostgreSQL, the database name must be in lowercase. **Important** **For MariaDB, you must specify the serverTimezone explicitly in the connection string: for example `jdbc:mariadb://myhost.mydomain.com:3306/arc_pcl027?serverTimezone=UTC`**
Shipped value	**No shipped value. This parameter must be filled in.**

Parameter name after installation	`arcsys.jdbc_url`

Parameter name in `INSTALL.properties`	**DATABASE_USER**
Parameter type	Characters string
Description	The user name for connecting to the relational database. For PostgreSQL, the database user must be in lowercase. Example(s): Arcsys
Shipped value	`ARCSYS`
Parameter name after installation	`arcsys.jdbc_user`

Parameter name in `INSTALL.properties`	**DATABASE_PASSWORD**
Parameter type	Encrypted characters string. To encrypt a clear characters string, use the encrypt encryption utility (`generate-key` and `encrypt` tools) provided in Arcsys Installation and Update Package or in Arcsys Engine. The encrypted characters string will be surrounded by ENC(...).
Description	The user password for connecting to the relational database.
Shipped value	**No shipped value. This parameter must be filled in.**
Parameter name after installation	`arcsys.jdbc_password`

2.2.4. Common API clients parameters (APICLIENTSPARAMS section)

This section contains the common parameters for connecting to the Arcsys RMI API.

Configuration filename after the installation: `AutoArchive.properties,RestoreClient.properties, SearchClient.properties.`

Note

This section requires completion if the value of `COMPONENTS` parameter of the installation script (`ArcsysInstaller.sh` or `ArcsysInstaller.bat`) is composed of at least one of these values: `minimumServerInstall`, `fullInstall`, `clientInstall`, `standardClients`, `autoarchiveAgent`, `fsComparator`, `batchs`

Parameter name in `INSTALL.properties`	**RMI_API_HOST_NAME_FOR_API_CLIENT**
Parameter type	Characters string
Description	The host name for the connection to the RMI API. If the client is on the same host as the RMI API, the localhost value is authorized
Shipped value	`localhost`
Parameter name after installation	`RMI_API_HOST_NAME`

Parameter name in `INSTALL.properties`	**RMI_API_PORT_NUMBER_FOR_API_CLIENT**
Parameter type	Positive integer
Description	The port number for the connection to the RMI API.
Shipped value	`25040`
Parameter name after installation	`RMI_API_PORT_NUMBER`

Parameter name in `INSTALL.properties`	**RMI_API_SERVICE_NAME_FOR_API_CLIENT**
Parameter type	Characters string
Description	The service name for the connection to the RMI API. (It should be the same as COMPONENT_API_SERVICE_NAME in the RMI API properties file).
Shipped value	`ArcsysRmiAPI`
Parameter name after installation	`RMI_API_PORT_NAME`

2.2.4.1. Authentication and rights administration

Parameter name in `INSTALL.properties`	**LDAP_USER_FOR_API_CLIENT**
Parameter type	Characters string
Description	The LDAP user name for the RMI API connection. The user must exist in the used LDAP directory.
Shipped value	**No shipped value. This parameter must be filled in.**
Parameter name after installation	`RMI_API_USERNAME`

Parameter name in `INSTALL.properties`	**LDAP_PASSWORD_FOR_API_CLIENT**
Parameter type	Encrypted characters string. To encrypt a clear characters string, use the encrypt encryption utility (`generate-key` and `encrypt` tools) provided in Arcsys Installation and Update Package or in Arcsys Engine. The encrypted characters string will be surrounded by ENC(...).
Description	The LDAP password for the RMI API connection.
Shipped value	**No shipped value. This parameter must be filled in.**
Parameter name after installation	`RMI_API_USERPASSWORD`

2.2.5. Common Agent clients parameters (AGENTCLIENTSPARAMS section)

This section contains the common parameters for connecting to the Arcsys Application Agent.

Configuration filename after the installation: `AutoArchive.properties,RestoreClient.properties, SearchClient.properties`

Note

This section requires completion if the value of `COMPONENTS` parameter of the installation script (`ArcsysInstaller.sh` or `ArcsysInstaller.bat`) is composed of at least one of these values: `minimumServerInstall`, `fullInstall`, `clientInstall`, `standardClients`, `autoarchiveAgent`

Parameter name in `INSTALL.properties`	**APP_AGENT_HOST_NAME_FOR_AGENT_CLIENT**
Parameter type	Characters string
Description	The host name for the connection to the Arcsys Application Agent as referenced in the agent host field in the relational database, as shown in the agents list section of the Web interface.
Shipped value	`localhost`
Parameter name after installation	`APP_AGENT_HOSTNAME`

Parameter name in `INSTALL.properties`	**APP_AGENT_PORT_NUMBER_FOR_AGENT_CLIENT**
Parameter type	Positive integer

Description	The port number for the connection to the Arcsys Application Agent.
Shipped value	`25030`
Parameter name after installation	`APP_AGENT_PORT`

Parameter name in `INSTALL.properties`	**APP_AGENT_SERVICE_NAME_FOR_AGENT_CLIENT**
Parameter type	Characters string
Description	The service name of the RMI access interface to the Application Agent.
Shipped value	`ArcsysAgentAPI`
Parameter name after installation	`APP_AGENT_SERVICE_NAME`

2.2.6. LDAP authentication common parameters (LDAPPARAMS section)

This section lists the common parameters for LDAP authentication.

Configuration filename after the installation: `ENGINE.properties,ARCSYS_RMI_API.properties, WEBAGENT_configuration.properties`

Note

This section requires completion if the value of `COMPONENTS` parameter of the installation script (`ArcsysInstaller.sh` or `ArcsysInstaller.bat`) is composed of at least one of these values: `minimumServerInstall`, `fullInstall`, `webAgent`, `engine`, `restWSApi`, `api`

Parameter name in `INSTALL.properties`	**LDAP_TYPE**
Parameter type	State variable. Possible values: `OPEN_DJ`, `OPEN_DS`, `OPEN_LDAP`, `ACTIVE_DIRECTORY`, `TIVOLI_DIRECTORY_SERVER`, `SUN_ONE_DIRECTORY_SERVER`
Description	The LDAP server type.
Shipped value	`OPEN_LDAP`

Parameter name in `INSTALL.properties`	**LDAP_SERVER_HOST**
Parameter type	Characters string

Description	The LDAP server host.
Shipped value	**No shipped value. This parameter must be filled in.**

Parameter name in `INSTALL.properties`	**LDAP_SERVER_PORT**
Parameter type	Positive integer
Description	The LDAP server port.
Shipped value	`389`

Parameter name in `INSTALL.properties`	**LDAP_AUTHENT_DN**
Parameter type	Characters string
Description	The LDAP user DN for authentication (initial bind to create the connection). It is the DN/login of a user with sufficient rights to consult the entire LDAP and, in the case of the Web Agent, to change a password (for example, for Active Directory, this user must be part of the Administrators group). If anonymous is chosen, an anonymous connection is used (under the premise that this is authorized by the specified directory). Such a connection cannot be used to write in the LDAP; therefore an anonymous login cannot be used if you want to enable the password change option
Shipped value	`anonymous`

Parameter name in `INSTALL.properties`	**LDAP_AUTHENT_PASSWORD**
Parameter type	Encrypted characters string. To encrypt a clear characters string, use the encrypt encryption utility (`generate-key` and `encrypt` tools) provided in Arcsys Installation and Update Package or in Arcsys Engine. The encrypted characters string will be surrounded by ENC(...).
Description	The LDAP user password for authentication (initial bind to create the connection).
Shipped value	**No shipped value. This parameter must be filled in.**
Parameter taken into account only if:	`LDAP_AUTHENT_DN` is different from `anonymous`

Parameter name in `INSTALL.properties`	**LDAP_USER_LOGIN_ATTRIBUTE_NAME**
Parameter type	Characters string
Description	The attribute name of an LDAP user entry containing the user login. This attribute must not be changed and be unique for each user because it is saved in the database.

Shipped value	`cn`

Parameter name in `INSTALL.properties`	**LDAP_GROUP_ID_ATTRIBUTE_NAME**
Parameter type	Characters string
Description	The attribute name of an LDAP group entry containing the group name. This attribute must not be changed and be unique for each user because it is saved in the database.
Shipped value	`cn`

Parameter name in `INSTALL.properties`	**LDAP_USER_DISPLAY_NAME_ATTRIBUTE_NAME**
Parameter type	Characters string
Description	The attribute name of an LDAP user entry containing the name to display for a user.
Shipped value	`sn`

Parameter name in `INSTALL.properties`	**LDAP_GROUP_DISPLAY_NAME_ATTRIBUTE_NAME**
Parameter type	Characters string
Description	The attribute name of an LDAP group entry containing the name to display for a group.
Shipped value	`sn`

2.2.6.1. LDAP user search

The users are searched in the LDAP from the search base DN according the search scope and the search filter.

Parameter name in `INSTALL.properties`	**LDAP_USER_SEARCH_BASE_DN**
Parameter type	Characters string
Description	The base DN of the LDAP server for user searches.
Shipped value	`ou=users,dc=infotel,dc=com`

Parameter name in `INSTALL.properties`	**LDAP_USER_SEARCH_SCOPE**

Parameter type	State variable. Possible values: `BASE`, `ONE`, `SUB`, `SUBORDINATE_SUBTREE`
Description	The scope of the user search in the LDAP server. * BASE: only the entry defined by the base DN is considered for the search * ONE: only the entries immediately below the entry defined by the base DN are considered for the search, the entry defined by the base DN is excluded * SUB: all the entries below the entry defined by the base DN and the entry defined by the base DN are considered for the search * SUBORDINATE_SUBTREE: all the entries immediately below the base DN are considered for the search, the entry defined by the base DN is excluded
Shipped value	`SUB`

Parameter name in `INSTALL.properties`	**LDAP_USER_SEARCH_FILTER**
Parameter type	Characters string
Description	The user search filter in the LDAP server.
Shipped value	`(objectClass=inetOrgPerson)`

2.2.6.2. LDAP group search

The groups are searched in the LDAP from the search base DN according the search scope and the search filter.

Parameter name in `INSTALL.properties`	**LDAP_GROUP_SEARCH_BASE_DN**
Parameter type	Characters string
Description	The base DN of the LDAP server for group searches.
Shipped value	`ou=arcsysgroup,ou=groups,dc=infotel,dc=com`

Parameter name in `INSTALL.properties`	**LDAP_GROUP_SEARCH_SCOPE**
Parameter type	State variable. Possible values: `BASE`, `ONE`, `SUB`, `SUBORDINATE_SUBTREE`
Description	The scope of the group search in the LDAP server. * BASE: only the entry defined by the base DN is considered for the search * ONE: only the entries immediately below the entry defined by the base DN are considered for the search, the entry defined by the base DN is excluded * SUB: all the entries below the entry defined by the base DN and the entry defined by the base DN are considered for the search

	* SUBORDINATE_SUBTREE: all the entries immediately below the base DN are considered for the search, the entry defined by the base DN is excluded
Shipped value	`SUB`

Parameter name in `INSTALL.properties`	**LDAP_GROUP_SEARCH_FILTER**
Parameter type	Characters string
Description	The search filter for groups in the LDAP server.
Shipped value	`(objectClass=groupOfNames)`

2.2.7. Common REST API clients (RESTAPICLIENTSPARAMS parameters)

This section contains the common parameters for connecting to the Arcsys REST API.

Configuration filename after the installation: `http-client.properties`

Note

This section requires completion if the value of `COMPONENTS` parameter of the installation script (`ArcsysInstaller.sh` or `ArcsysInstaller.bat`) is composed of at least one of these values: `fullInstall`, `batchReporting`, `classAssigner`, `metadataReplacement`, `startRetentionDateAssigner`, `cli`

Parameter name in `INSTALL.properties`	**REST_API_CLIENT_BASE_URL**
Parameter type	URL
Description	The REST API base URL which includes scheme, host, port and context for the connection to the Arcsys REST API.
Shipped value	`http://localhost:8090/api`
Parameter name after installation	`base.url`

2.2.7.1. OAuth2 settings

This section contains the OAuth2 settings parameters for connecting to the Arcsys REST API.

Parameter name in `INSTALL.properties`	**REST_API_CLIENT_GRANT_TYPE**
Parameter type	State variable. Possible values: `authorization-code`, `implicit`, `password`, `client-credentials`
Description	The OAuth2 authorization grant type.
Shipped value	`password`
Parameter name after installation	`oauth.authorization.grant.type`

Parameter name in `INSTALL.properties`	**REST_API_CLIENT_ID**
Parameter type	Characters string
Description	The OAuth2 client identifier.
Shipped value	**No shipped value. This parameter must be filled in.**
Parameter name after installation	`oauth.client.id`

Parameter name in `INSTALL.properties`	**REST_API_CLIENT_SECRET**
Parameter type	Encrypted characters string. To encrypt a clear characters string, use the encrypt encryption utility (`generate-key` and `encrypt` tools) provided in Arcsys Installation and Update Package or in Arcsys Engine. The encrypted characters string will be surrounded by ENC(...).
Description	The OAuth2 client secret password.
Shipped value	**No shipped value. This parameter must be filled in.**
Parameter name after installation	`oauth.client.secret`

Parameter name in `INSTALL.properties`	**REST_API_CLIENT_USERNAME**
Parameter type	Characters string
Description	The OAuth2 user name.
Shipped value	**No shipped value. This parameter must be filled in.**
Parameter name after installation	`oauth.user.name`

Parameter name in `INSTALL.properties`	**REST_API_CLIENT_USER_PASSWORD**

Parameter type	Encrypted characters string. To encrypt a clear characters string, use the encrypt encryption utility (`generate-key` and `encrypt` tools) provided in Arcsys Installation and Update Package or in Arcsys Engine. The encrypted characters string will be surrounded by ENC(...).
Description	The OAuth2 user password.
Shipped value	**No shipped value. This parameter must be filled in.**
Parameter name after installation	`oauth.user.password`

2.3. Module specific parameters

2.3.1. Database parameters (REFPARAMS section)

This section contains the Arcsys Database parameters.

Note

This section requires completion if the value of `COMPONENTS` parameter of the installation script (`ArcsysInstaller.sh` or `ArcsysInstaller.bat`) is composed of at least one of these values: `minimumServerInstall`, `fullInstall`, `database`

Filling this section implies to fill the following sections: `DBPARAMS`, `DBJDBCPARAMS`

2.3.1.1. Oracle Database Case

The parameters for the case of an Oracle Database.

Section of parameters taken into account only if: `TYPEBASE` = `Oracle`

Parameter name in `INSTALL.properties`	**DATA_TABLESPACE_NAME**
Parameter type	Characters string
Description	The name of the pre-existing tablespace on which the tables will be created.
Shipped value	`ARCSYS_DATA`

Parameter name in `INSTALL.properties`	**INDEX_TABLESPACE_NAME**
Parameter type	Characters string

Description	The name of the pre-existing tablespace on which the indexes will be created.
Shipped value	`ARCSYS_INDEX`

Parameter name in `INSTALL.properties`	**RAC**
Parameter type	Boolean. Possible values: `true`, `false`
Description	Specifies whether Oracle Database is installed in Real Application Clusters mode (`RAC=true`). In this case, the sequences will be defined with the ORDER option.
Shipped value	`false`

2.3.1.2. For all database types

The parameters that need to be informed for all database types.

Parameter name in `INSTALL.properties`	**SITE_CODE_FOR_DATABASE**
Parameter type	Characters string
Description	The code of the site that will be automatically created by the installer in the database.
Shipped value	`PRIMARY_SITE`

Parameter name in `INSTALL.properties`	**SAMPLE_BASE_OWNER**
Parameter type	Characters string
Description	The owner of the sample repository. This is the LDAP user noted as the administration repository and collection creator. The user must exist in the used LDAP directory.
Shipped value	**No shipped value. This parameter must be filled in.**

2.3.2. RMI and TCP/IP API parameters (RMI_TCPIP_APIPARAMS section)

This section contains the Arcsys RMI, TCP/IP and SOAP API parameters. Configuration filename after the installation ARCSYS_RMI_API.properties, ARCSYS_TCPIP_API.properties.

Note

This section requires completion if the value of `COMPONENTS` parameter of the installation script (`ArcsysInstaller.sh` or `ArcsysInstaller.bat`) is composed of at least one of these values: `minimumServerInstall`, `fullInstall`, `api`

Filling this section implies to fill the following sections: `APIPARAMS`, `DBPARAMS`, `DBJDBCPARAMS`, `LDAPPARAMS`

Parameter name in `INSTALL.properties`	**RMI_API_PORT_NUMBER**
Parameter type	Positive integer
Description	The RMI service port number of the RMI API. This port is only used if the RMI API is deployed (`RMI_API_DEPLOY=true`) which is the default case after installation.
Shipped value	`25040`
Configuration filename after the installation	`ARCSYS_RMI_API.properties`
Parameter name after installation	`COMPONENT_API_PORT_NUMBER`

Parameter name in `INSTALL.properties`	**RMI_API_SERVICE_NAME**
Parameter type	Characters string
Description	The name of the RMI API Service.
Shipped value	`ArcsysRmiAPI`
Configuration filename after the installation	`ARCSYS_RMI_API.properties`
Parameter name after installation	`COMPONENT_API_SERVICE_NAME`

Parameter name in `INSTALL.properties`	**TCPIP_API_PORT_NUMBER**
Parameter type	Positive integer
Description	The TCP/IP service port of the Arcsys API. This port is only used if the TCP/IP API is deployed (`TCPIP_API_DEPLOY=true`) which is the default case after installation.
Shipped value	`25050`
Configuration filename after the installation	`ARCSYS_TCPIP_API.properties`

Parameter name after installation	`COMPONENT_API_PORT_NUMBER`

2.3.3. REST API parameters (RESTAPIPARAMS section)

This section contains the Arcsys REST API parameters.

Note

This section requires completion if the value of `COMPONENTS` parameter of the installation script (`ArcsysInstaller.sh` or `ArcsysInstaller.bat`) is composed of at least one of these values: `minimumServerInstall`, `fullInstall`, `restWSApi`

Filling this section implies to fill the following sections: `APIPARAMS`, `DBPARAMS`, `DBJDBCPARAMS`, `LDAPPARAMS`

Parameter name in `INSTALL.properties`	**REST_API_SECURITY_CLIENT_SECRET**
Parameter type	Encrypted characters string. To encrypt a clear characters string, use the encrypt encryption utility (`generate-key` and `encrypt` tools) provided in Arcsys Installation and Update Package or in Arcsys Engine. The encrypted characters string will be surrounded by ENC(...).
Description	The OAuth2 client secret for Rest API, encrypted with `encrypt.sh(.cmd)`.
Shipped value	**No shipped value. This parameter must be filled in.**

2.3.4. Transfer server parameters (TRSPARAMS section)

This section contains the Arcsys Transfer Server parameters.

Configuration filename after the installation: `transferServer.conf`

Note

This section requires completion if the value of `COMPONENTS` parameter of the installation script (`ArcsysInstaller.sh` or `ArcsysInstaller.bat`) is composed of at least one of these values: `minimumServerInstall`, `fullInstall`, `transferServer`

Parameter name in `INSTALL.properties`	**TRANSFER_SERVER_CODE**
Parameter type	Characters string
Description	The code (name) of this component, to avoid duplication.This name must be unique (on all sites). The code must identify the module in the installation. It is recommended to use a logical name distinct from a physical host name for example. It must not include the following characters: ; , / , \ , " , ' and space.
Shipped value	`ArcsysTransferServerPrimary`
Parameter name after installation	`COMPONENT_CODE`

Parameter name in `INSTALL.properties`	**SITE_CODE_FOR_TRANSFER_SERVER**
Parameter type	Characters string
Description	Site code to which the module must belong. This site must be created in the Web interface beforehand (by default, the installer creates systematically a PRIMARY_SITE site code)
Shipped value	`PRIMARY_SITE`
Parameter name after installation	`SITE_CODE`

Parameter name in `INSTALL.properties`	**TRANSFER_SERVER_PORT_NUMBER**
Parameter type	Characters string
Description	The port number on which the transfer server is deployed. For compatibility, this parameter also accepts a service name defined in `/etc/services`.
Shipped value	`25010`
Parameter name after installation	`component_port_number`

Parameter name in `INSTALL.properties`	**TCP_HOST_NAME_FOR_TRANSFER_SERVER**
Parameter type	Characters string
Description	The host name for the connection to the TCP/IP API.
Shipped value	`localhost`
Parameter name after installation	`api_host_name`

Parameter name in `INSTALL.properties`	**TCP_PORT_NUMBER_FOR_TRANSFER_SERVER**
Parameter type	Characters string
Description	The port number for the connection to the TCP/IP API. For compatibility, this parameter also accepts a service name defined in `/etc/services`.
Shipped value	`25050`
Parameter name after installation	`api_port_number`

2.3.5. Engine parameters (ENGINEPARAMS section)

This section contains the Arcsys Engine parameters.

Configuration filename after the installation: `ENGINE.properties`

Note

This section requires completion if the value of `COMPONENTS` parameter of the installation script (`ArcsysInstaller.sh` or `ArcsysInstaller.bat`) is composed of at least one of these values: `minimumServerInstall`, `fullInstall`, `engine`

Filling this section implies to fill the following sections: `DBPARAMS`, `DBJDBCPARAMS`, `LDAPPARAMS`

Parameter name in `INSTALL.properties`	**ENGINE_CODE**
Parameter type	Characters string
Description	The code (name) of this component, to avoid duplication.This name must be unique (on all sites). The code must identify the module in the installation. This code is used as the RMI service name under which the engine is deployed. It is recommended to use a logical name distinct from a physical host name for example. It must not include the following characters: ; , / , \ , " , ' and space.
Shipped value	`ArcsysEnginePrimary`
Parameter name after installation	`COMPONENT_CODE`

Parameter name in `INSTALL.properties`	**SITE_CODE_FOR_ENGINE**

Parameter type	Characters string
Description	Site code to which the module must belong. This site must be created in the Web interface beforehand (by default, the installer creates systematically a PRIMARY_SITE site code)
Shipped value	`PRIMARY_SITE`
Parameter name after installation	`SITE_CODE`

Parameter name in `INSTALL.properties`	**ENGINE_PORT_NUMBER**
Parameter type	Positive integer
Description	The port number on which the Engine API is deployed.
Shipped value	`25020`
Parameter name after installation	`COMPONENT_API_PORT_NUMBER`

2.3.5.1. Engine LDAP

Authentication and rights administration.

Parameter name in `INSTALL.properties`	**LDAP_USER_FOR_ENGINE**
Parameter type	Characters string
Description	The LDAP user for the ArcsysEngine RMI interface. It is the LDAP user used to start the engine. This parameter is not required in SSO mode. The user must exist in the used LDAP directory
Shipped value	**No shipped value. This parameter must be filled in.**
Parameter name after installation	`USERNAME`

Parameter name in `INSTALL.properties`	**LDAP_PASSWORD_FOR_ENGINE**
Parameter type	Encrypted characters string. To encrypt a clear characters string, use the encrypt encryption utility (`generate-key` and `encrypt` tools) provided in Arcsys Installation and Update Package or in Arcsys Engine. The encrypted characters string will be surrounded by ENC(...).
Description	The LDAP password for the ArcsysEngine RMI interface. It is the LDAP user password used to start the engine. This parameter is not required in SSO mode.
Shipped value	**No shipped value. This parameter must be filled in.**

Parameter name after installation	`USERPASSWORD`

2.3.5.2. Requests processing

Parameter name in `INSTALL.properties`	**ARCHIVING_ENGINE**
Parameter type	Boolean. Possible values: `true`, `false`
Description	Specifies whether this engine will process archiving requests.
Shipped value	**No shipped value. This parameter must be filled in.**
Option(s)	`ArcHP`: If the ArcHP option is not enabled, one engine per site with this specialization is authorized. If the ArcHP Option is enabled, you can have as many engines per site as you want with this specialization.

Parameter name in `INSTALL.properties`	**RETRIEVAL_ENGINE**
Parameter type	Boolean. Possible values: `true`, `false`
Description	Specifies whether this engine will process retrieval requests.
Shipped value	**No shipped value. This parameter must be filled in.**
Option(s)	`ArcHP`: If the ArcHP option is not enabled, one engine per site with this specialization is authorized. If the ArcHP Option is enabled, you can have as many engines per site as you want with this specialization.

Parameter name in `INSTALL.properties`	**RESTITUTION_ENGINE**
Parameter type	Boolean. Possible values: `true`, `false`
Description	Specifies whether this engine will process archive restitution requests.
Shipped value	**No shipped value. This parameter must be filled in.**
Option(s)	`ArcHP`: If the ArcHP option is not enabled, one engine per site with this specialization is authorized. If the ArcHP Option is enabled, you can have as many engines per site as you want with this specialization.

Parameter name in `INSTALL.properties`	**MIGRATION_ENGINE**
Parameter type	Boolean. Possible values: `true`, `false`
Description	Specifies whether this engine will manage the migration process (migration of records from one storage zone to another storage zone). This involves both creating and processing these requests on the storage zones belonging to

	the site of this engine. There must be an engine managing these requests per site where the storage zones are defined.
Shipped value	**No shipped value. This parameter must be filled in.**
Option(s)	`ArcHP`: If ArcHP Option is not enabled, it is possible to have several engines per site managing these requests, but the time slots must not overlap. If ArcHP Option is enabled, it is possible to have several engines per site managing this processing, even with shared time slots.

Parameter name in `INSTALL.properties`	**CHECKZONES_ENGINE**
Parameter type	Boolean. Possible values: `true`, `false`
Description	Specifies whether the engine will process zone checking (random checking of envelope integrity). This involves both creating and processing these requests on the storage zones belonging to the site of this engine. There must be an engine managing these requests per site where the storage zones are defined.
Shipped value	**No shipped value. This parameter must be filled in.**
Option(s)	`ArcHP`: If ArcHP Option is not enabled, it is possible to have several engines per site managing these requests, but the time slots must not overlap. If ArcHP Option is enabled, it is possible to have several engines per site managing this processing, even with shared time slots.

Parameter name in `INSTALL.properties`	**RECOVERY_ENGINE**
Parameter type	Boolean. Possible values: `true`, `false`
Description	Specifies whether this engine will process recovery (re-establishment of copy showing an error). This involves both creating and processing these requests on the storage zones belonging to the site of this engine. There must be an engine managing these requests per site where the storage zones are defined.
Shipped value	**No shipped value. This parameter must be filled in.**
Option(s)	`ArcHP`: If ArcHP Option is not enabled, it is possible to have several engines per site managing these requests, but the time slots must not overlap. If ArcHP Option is enabled, it is possible to have several engines per site managing this processing, even with shared time slots.

Parameter name in `INSTALL.properties`	**POSTPONE_ENGINE**
Parameter type	Boolean. Possible values: `true`, `false`
Description	Specifies whether this engine will process postponed requests (requests created for potentially long operations in the web interface). This means both creating and processing these requests on all storage zones. It is possible to have several engines per site managing these requests but the time slots must not overlap.

	 **Important** **Even if ArcHP Option is enabled, it is forbidden to have several engines managing these requests on shared time slots**
Shipped value	**No shipped value. This parameter must be filled in.**

Parameter name in `INSTALL.properties`	**COPY_ENGINE**
Parameter type	Boolean. Possible values: `true`, `false`
Description	Specifies whether this engine will handle the copy process of archived lots from one zone in a zone list to a zone in another zone list of a storage profile. This involves only processing these requests on the storage zones belonging to the site of this engine. There must be an engine managing these requests per site where the storage zones are defined.
Shipped value	**No shipped value. This parameter must be filled in.**
Parameter name after installation	`COPY_REQUESTS_PROCESSING`
Option(s)	`ArcHP`: If ArcHP Option is not enabled, it is possible to have several engines per site managing these requests, but the time slots must not overlap. If ArcHP Option is enabled, it is possible to have several engines per site managing this processing, even with shared time slots.

Parameter name in `INSTALL.properties`	**SYNC_ENGINE**
Parameter type	Boolean. Possible values: `true`, `false`
Description	Specifies whether this engine will process manifest synchronization. This means both creating and processing these requests on all storage zones. It is possible to have several engines per site managing these requests but the time slots must not overlap.  **Important** **Even if ArcHP Option is enabled, it is forbidden to have several engines managing these requests on shared time slots**
Shipped value	**No shipped value. This parameter must be filled in.**

Parameter name in `INSTALL.properties`	**MEDIA_MIGRATION_ENGINE**
Parameter type	Boolean. Possible values: `true`, `false`

Description	Specifies whether this engine will process media expiry. This involves both creating and processing these requests on the storage zones belonging to the site of this engine. There must be an engine managing these requests per site where the storage zones are defined.
Shipped value	**No shipped value. This parameter must be filled in.**
Option(s)	`ArcMover Tape`: You must have this option to use this parameter. `ArcHP`: If ArcHP Option is not enabled, it is possible to have several engines per site managing these requests, but the time slots must not overlap. If ArcHP Option is enabled, it is possible to have several engines per site managing this processing, even with shared time slots.

Parameter name in `INSTALL.properties`	**RECYCLING_ENGINE**
Parameter type	Boolean. Possible values: `true`, `false`
Description	Specifies whether this engine will process tape recycling. This involves both creating and processing these requests on the storage zones belonging to the site of this engine. There must be an engine managing these requests per site where the storage zones are defined.
Shipped value	**No shipped value. This parameter must be filled in.**
Option(s)	`ArcMover Tape`: You must have this option to use this parameter. `ArcHP`: If ArcHP Option is not enabled, it is possible to have several engines per site managing these requests, but the time slots must not overlap. If ArcHP Option is enabled, it is possible to have several engines per site managing this processing, even with shared time slots.

Parameter name in `INSTALL.properties`	**DISPOSAL_ENGINE**
Parameter type	Boolean. Possible values: `true`, `false`
Description	Specifies whether this engine will process disposal by retention schedule. This means both creating and processing these requests on all storage zones. It is possible to have several engines per site managing these requests but the time slots must not overlap.  **Important** **Even if ArcHP Option is enabled, it is forbidden to have several engines managing these requests on shared time slots**
Shipped value	**No shipped value. This parameter must be filled in.**

2.3.6. ArcFF parameters (ARCFFPARAMS section)

This section contains the parameters for ArcFF format control module.

Configuration filename after the installation: `arcff.properties`

Note

This section requires completion if the value of `COMPONENTS` parameter of the installation script (`ArcsysInstaller.sh` or `ArcsysInstaller.bat`) is composed of at least one of these values: `minimumServerInstall`, `fullInstall`, `fileFormatServer`

Parameter name in `INSTALL.properties`	**ARCFF_SECURITY_CLIENT_SECRET**
Parameter type	Encrypted characters string. To encrypt a clear characters string, use the encrypt encryption utility (`generate-key` and `encrypt` tools) provided in Arcsys Installation and Update Package or in Arcsys Engine. The encrypted characters string will be surrounded by ENC(...).
Description	The secret shared by identifier of the client application for OAuth2 authentication on the ArcFF format control module. All application agents connecting to this module must have the same shared secret.
Shipped value	**No shipped value. This parameter must be filled in.**
Parameter name after installation	`SECURITY_CLIENT_SECRET`

2.3.7. Application Agent parameters (APPAGENTPARAMS section)

This section contains the Arcsys Application Agent parameters.

Configuration filename after the installation: `APPAGENT.properties`

Note

This section requires completion if the value of `COMPONENTS` parameter of the installation script (`ArcsysInstaller.sh` or `ArcsysInstaller.bat`) is composed of at least one of these values: `minimumServerInstall`, `fullInstall`, `clientInstall`, `applicationAgent`

Parameter name in `INSTALL.properties`	**APP_AGENT_CODE**
Parameter type	Characters string

Description	The name of this component, to avoid duplication. This name must be unique (on all sites). The code must identify the module in the installation. It is recommended to use a logical name distinct from a physical host name for example.
Shipped value	`ArcsysApplicationAgentPrimary`
Parameter name after installation	`COMPONENT_CODE`

Parameter name in `INSTALL.properties`	**APP_AGENT_PORT_NUMBER**
Parameter type	Positive integer
Description	The port number on which the Application Agent API is deployed.
Shipped value	`25030`
Parameter name after installation	`COMPONENT_API_PORT_NUMBER`

Parameter name in `INSTALL.properties`	**APP_AGENT_SERVICE_NAME**
Parameter type	Characters string
Description	The name of the Application Agent Service
Shipped value	`ArcsysAgentAPI`
Parameter name after installation	`COMPONENT_API_SERVICE_NAME`

Parameter name in `INSTALL.properties`	**ENGINE_API_LIST**
Parameter type	List of string separated by commas
Description	The list of URLs of Engines API, the first URL being the first to be contacted when starting. The pattern of a URL is: `rmi://Host or IP or localhost:Port/COMPONENT_CODE` of the Engine. List of addresses used for contacting the engine API. Each address has the following format `rmi://Engine host:Engine port/Engine code`. The parameter Engine host must contain an address from the application agent machine. It does not necessarily correspond to the value of the engine's physical host registered in the relational database and visible in the engine interface. The value localhost is accepted, for instance. The engine code can be found in the COMPONENT_CODE parameter of the engine configuration file; the engine port is specified in the

	COMPONENT_API_PORT_NUMBER parameter of the engine configuration file. If an engine in the list cannot be contacted, the agent will not start. An engine is considered unavailable if it does not exist in the repository or if it has not been updated for more than two minutes. This time can be set with the SEC_ENGINE_AVAILABLE_TIMEOUT parameter of the engine. Example(s): An example of Engine API URL is `rmi://192.168.43.222:25020/ArcsysEnginePrimary` where 25020 is the COMPONENT_API_PORT_NUMBER of the Engine and ArcsysEnginePrimary is the COMPONENT_CODE of the Engine. Other example: `rmi://localhost:25020/ArcsysPrimaryEngine`
Shipped value	`rmi://localhost:25020/ArcsysEnginePrimary/`

Parameter name in `INSTALL.properties`	**PRIORITY_SITE_LIST**
Parameter type	List of string separated by commas
Description	The list of site codes. The engines of the first site in the list are the most priority for processing archiving / retrieval requests of this agent. All engines of each site have the same priority. If a site is not defined in the list, the request will not be processed by an engine of this site.
Shipped value	`PRIMARY_SITE`

Parameter name in `INSTALL.properties`	**LOCAL_TRANSFER_SERVICES_FOR_APP_AGENT**
Parameter type	List of string separated by commas
Description	The list of the codes of the local transfer services (NFS mount included) for the application agent. The transfer servers are also transfer services: they can be included in this list. If a local code is not given, the ARCH_REPOSITORY_DIR and REST_REPOSITORY_DIR directories must contain the directories of a transfer service from which data will be transferred to a transfer server managing the storage zone. If a local code is given, the data will simply be moved from the ARCH_REPOSITORY_DIR and REST_REPOSITORY_DIR directories to the cache directories of the transfer servers. **Important** **The list must not be empty.**
Shipped value	`ArcsysTransferServerPrimary`
Parameter name after installation	`LOCAL_TRANSFER_SERVICES`

Parameter name in `INSTALL.properties`	**ARCH_REPOSITORY_PATH**

Parameter type	File path
Description	The absolute path of a directory used to store the files to be archived. This path corresponds to the output_stage_dir directory of the transfer service. The path must correspond to an existing directory, with adequate access rights (read, write and execute). ${TARGET_FOLDER} is a special variable that automatically takes the value of the parameter of the -DTARGET_FOLDER command line. Absolute file path. Use Unix like separators ('/'), even on Windows.
Shipped value	`${TARGET_FOLDER}/ArcsysTransferServer/stage/cli/out`

Parameter name in `INSTALL.properties`	**REST_REPOSITORY_PATH**
Parameter type	File path
Description	The absolute path of a directory used to store the files to be retrieved. This path corresponds to the input_stage_dir directory of the transfer service. The path must correspond to an existing directory, with adequate access rights (read, write and execute). ${TARGET_FOLDER} is a special variable that automatically takes the value of the parameter of the -DTARGET_FOLDER command line. Absolute file path. Use Unix like separators ('/'), even on Windows.
Shipped value	`${TARGET_FOLDER}/ArcsysTransferServer/stage/cli/in`

2.3.8. Web agent generic parameters (WEBGENERICPARAMS section)

This section lists the general parameters (non-specific for a JEE application server) of the Arcsys Web Agent.

Configuration filename after the installation: `WEBAGENT_configuration.properties`

Note

This section requires completion if the value of `COMPONENTS` parameter of the installation script (`ArcsysInstaller.sh` or `ArcsysInstaller.bat`) is composed of at least one of these values: `fullInstall`, `webAgent`

Filling this section implies to fill the following sections: `WEBSPECIFICPARAMS`, `DBPARAMS`, `LDAPPARAMS`

Parameter name in `INSTALL.properties`	**SITE_CODE_FOR_WEB_AGENT**
Parameter type	Characters string
Description	Site code to which the module must belong. This site must be created in the Web interface beforehand (by default, the installer creates systematically a PRIMARY_SITE site code).
Shipped value	`PRIMARY_SITE`
Parameter name after installation	`SITE_CODE`

2.3.9. Web agent JEE specific parameters (WEBSPECIFICPARAMS subsection)

Application server and installation mode to use.

Note

This section requires completion if the value of `COMPONENTS` parameter of the installation script (`ArcsysInstaller.sh` or `ArcsysInstaller.bat`) is composed of at least one of these values: `fullInstall`, `webAgent`

Parameter name in `INSTALL.properties`	**WEB_APPLICATION_SERVER**
Parameter type	State variable. Possible values: `TOMCAT-AUTO`, `TOMCAT`, `WILDFLY-AUTO`, `WILDFLY`, `OTHER`
Description	The application server and installation mode to use. The possible values on a JBoss/WildFly application server are: * WILDFLY_AUTO: Automatic deployment of the Web application. Automatic deployment must not be used in the case of specific configuration of the application server. After installation, the server will be stopped. Do not forget to run it again with the provided startarcsys.sh script in ArcsysWebAgent/server/wildfly. * WILDFLY: Manual installation of the Web application. Manual deployment of the Web will then be required. After installation, the server will be stopped. Do not forget to run it again, either with the provided startarcsys.sh script or in service mode. The possible values on a Apache Tomcat application server are: * TOMCAT-AUTO: Automatic deployment of the Web application. Automatic deployment must not be used in the case of specific configuration of the application server.

	* TOMCAT: Manual installation of the Web application. Manual deployment of the Web will then be required. * OTHER: If you specify OTHER, manual deployment of the Web will then be required
Shipped value	`TOMCAT-AUTO`

2.3.10. Web agent common automatic deployment parameters (AUTOWEBPARAMS subsection)

When you want to install in automatic deployment mode (application, environment variables) in JBoss / WildFly or in Apache Tomcat

Note

This section requires completion if the value of `COMPONENTS` parameter of the installation script (`ArcsysInstaller.sh` or `ArcsysInstaller.bat`) is composed of at least one of these values: `fullInstall`, `webAgent`

Section of parameters taken into account only if: `WEB_APPLICATION_SERVER` = `WILDFLY-AUTO` or `TOMCAT-AUTO`

Parameter name in `INSTALL.properties`	**WEB_APPLICATION_SERVER_PATH**
Parameter type	Directory path
Description	The web application server absolute path directory. Only necessary in case of automatic deployment. Absolute path of a directory. The path must correspond to an existing directory, with adequate access rights (read, write and execute). Use Unix-like separators ('/'), even on Windows.
Shipped value	**No shipped value. This parameter must be filled in.**
Parameter taken into account only if:	`WEB_APPLICATION_SERVER` = `TOMCAT-AUTO` or `WILDFLY-AUTO`

2.3.11. Transfer service parameters (TRSVCEPARAMS section)

This section contains the Arcsys Transfer Service parameters.

Configuration filename after the installation: `transferService.conf`

Note

This section requires completion if the value of `COMPONENTS` parameter of the installation script (`ArcsysInstaller.sh` or `ArcsysInstaller.bat`) is composed of at least one of these values: `clientInstall`, `transferService`

Parameter name in `INSTALL.properties`	**TRANSFER_SERVICE_CODE**
Parameter type	Characters string
Description	The code (name) of this component, to avoid duplication. This name must be unique (on all sites). It is recommended to use a logical name distinct from a physical host name for example. It must not include the following characters: ; , / , \ , " , ' and space.
Shipped value	`ArcsysTransferServicePrimary`
Parameter name after installation	`component_code`

Parameter name in `INSTALL.properties`	**TRANSFER_SERVICE_PORT_NUMBER**
Parameter type	Positive integer
Description	The transfer service port number on which the transfer service is deployed.
Shipped value	`25010`
Parameter name after installation	`component_port_number`

Parameter name in `INSTALL.properties`	**TRANSFER_SERVER_LIST**
Parameter type	List of string separated by commas
Description	The list of addresses on which to contact the transfer server on startup of the transfer service. The transfer service contacts a transfer server to register the transfer server in the relational database. This parameter can list all the transfer servers installed, irrespective of the site to which they are attached. The transfer servers will be contacted in their order of definition, until a connection leads to registering the service in the repository (addition or update). Each address has the following format `Transfer server host:Transfer server port`. The parameter Transfer server host must contain an address from the transfer service machine. It does not necessarily correspond to the value of

	the physical host of the transfer server registered in the relational database and visible in the transfer servers. The value localhost is accepted, for instance. The transfer server port is specified in the component_port_number parameter of the configuration file of the transfer server. Example(s): An example of Transfer Server URL is `192.168.43.222:25010`.
Shipped value	**No shipped value. This parameter must be filled in.**
Parameter name after installation	`transfer_server_list`

3. Deploying Arcsys Web Agent on a server (with manual deployment)

3.1. Java environment variables: overview

The following environment variables are currently required for Arcsys Web Agent operation:

- *arcsys.alert.manager.factory*: Java class to instantiate the alert manager (internal parameter). The value must be the following: `com.infotel.arcsys.alert.manager.ArcsysWebAgentAlertManagerFactory`
- *log4j.configurationFile*: Full path of the `log4j2.xml` properties file (found by default in the `etc` folder of the module)
- *arcsys.trace.folder*: Full path of the directory which must contain the trace files.
- *arcsys.bdd*: Full path of the `REFERENTIAL_JDBC.properties` properties file (found by default in the `etc` folder of the module)
- *arcsys.web.config*: Full path of the `WEBAGENT_configuration.properties` properties file (found by default in the `etc` folder of the module)
- *arcsys.third.party.search.engine*: Full path of the third party search engine properties file `THIRD_PARTY_SEARCH_ENGINE.properties` (found by default in the `etc` folder of the module)
- *secret.key.file*: Full path of the secret key file to encrypt the passwords thanks to the `encrypt` mechanism.
- *java.awt.headless=true*: Enables "headless mode" mode to view the web interface graphics in the absence of server X11.

  Requirement If you want to use the X11 graphics server on a Unix machine, you must first install the `libxp6` and `libxtst6` libraries.

3.2. Suggested update tool in JBoss / WildFly environment

The Java environment variables can be updated by the `jboss-cli` utility (using `system-property`). This script is run with the application server started up.

Here is an example:

```
$ ./bin/jboss-cli.sh --connect controller=IP_ADDRESS
                        [standalone@IP_ADDRESS:9990 /] /system-property=log4j.
configurationFile:add(value=/opt/Infotel/ArchivingProduct/ArcsysWebAgent/etc/log4j2.xml)
                        [standalone@IP_ADDRESS:9990 /] /system-property=arcsys.
bdd:add(value=/opt/Infotel/ArchivingProduct/ArcsysWebAgent/etc/REFERENTIAL_JDBC.properties)
                        [standalone@IP_ADDRESS:9990 /] /system-property=arcsys.web.config
:add(value=/opt/Infotel/ArchivingProduct/ArcsysWebAgent/etc/WEBAGENT_configuration.propert
ies)
                        [standalone@IP_ADDRESS:9990 /] /system-property=arcsys.alert.
manager.factory:add(value=com.infotel.arcsys.alert.manager.ArcsysWebAgentAlertManagerFacto
ry)
                        [standalone@IP_ADDRESS:9990 /] /system-property=arcsys.third.p
arty.search.engine:add(value=/opt/Infotel/ArchivingProduct/ArcsysWebAgent/etc/THIRD_PARTY_
SEARCH_ENGINE.properties)
                        [standalone@IP_ADDRESS:9990 /] /system-property=secret.key.
file:add(value=/opt/Infotel/ArchivingProduct/secretkeyfile)
```

3.3. Suggested update tool in Apache Tomcat environment

The Java environment variables can be updated by the `CATALINA_OPTS` variable in the `setenv.sh` file.

3.4. Deploying the application

3.4.1. WAR file to deploy

The WAR file to deploy is the `webarcsys.war` file found in the `war` folder of the Arcsys Web Agent. If this file is already deployed (for interrupted installation or update), we recommend to uninstall it first. When updating from a version prior to 5.2.0.0, uninstall the EAR file(s) first.

3.4.2. Deployment procedure

The deployment procedure depends on the application server used.

For JBoss / WildFly, you must deploy the WAR file in the `deployments` folder of the JBoss / WildFly instance (for example, `standalone/deployments`). You can also use the `jboss-cli` utility to deploy the file with the `deploy` command.

For Apache Tomcat, you must deploy the WAR file in the `webapps` folder of the Apache Tomcat instance.

4. Launching installation

Use the following command line to run the Arcsys installer:

```
./ArcsysInstaller.sh (or ArcsysInstaller.bat on Windows system) <Parameters>
```

Note

For a Windows system, the installer is not compliant with PowerShell.

The installation parameters are the following:

- *-DCOMPONENTS*: The type of installation or the list of modules you want to install.

  For a full installation, `-DCOMPONENTS=fullInstall`

  For a partial installation with preconfired set of module, `-DCOMPONENTS=minimumServerInstall` or `-DCOMPONENTS=clientInstall`.

  The installer also allows you to choose the components you wish to install. To do so, you must indicate these components, separated by a comma, via the *COMPONENTS* parameter. It is also possible to install only one component. The aliases of the components to be used in the parameter are the same as the one defined for the individual installation of each component (Chapter 2, INSTALL.properties File): `api,restWSApi,applicationAgent,database,engine,transferServer,transferService,webAgent,autoarchiveAgent,copyRequestsManager,fileFormatServer,fsComparator,standardClients,batchs,batchReporting,classAssigner,metadataReplacement,startRetentionDateAssigner,refInjector,signatureValidator,reportingInitializer,cli`

  For the installation of `refInjector` and `signatureValidator` components, it is not necessary to edit the `INSTALL.properties` file (Chapter 2, INSTALL.properties File). None of `INSTALL.properties` file parameters have any impact on the installation of ArcREF Injector and Signature validator module.

  Example: `-DCOMPONENTS=api,applicationAgent,database,engine,transferServer`

Important

On a Windows system, you must put the component list in quotes.

- *-DTARGET_FOLDER*: directory in which the module will be installed. For a Windows system, double the "\" character. For example: `-DTARGET_FOLDER=C:\\Arcsys`
- *-DSECRET_KEY_PATH*: path to the secret key file generated by the encrypt tool. Example: `-DSECRET_KEY_PATH=/opt/Infotel/Arcsys/secret-key`

- *-DINSTALL_CONFIG_PATH*: optional parameter which can be used to specify the path to the `INSTALL.properties` file. If not specified, the `INSTALL.properties` file in the installer folder is used. Example: `-DINSTALL_CONFIG_PATH=/home/Arcsys/install/INSTALL_CONFIG.properties`

- *-DJAVA_PATH*: optional parameter which can be used to specify the path to the java executable. This parameter is used for the systemd services definition files. If you do not use systemd to start Arcsys you do not need to specify *-DJAVA_PATH*. If not specified, `/usr/bin/java` is used. Example: *-DJAVA_PATH*=`/usr/lib64/jvm/jre-11-openjdk/bin/java`

- *-DLICENSE_PATH*: Path to the license file provided by Infotel. This parameter is required only for these types of installations: `fullInstall`, `minimumServerInstall` and for installations involving these modules: `api`, `applicationAgent`, `database`, `engine`, `transferServer`

5. Finalizing installation

After installation, it is recommended that you review the configuration of all installed components, using Arcsys Administration Manual for more information.

More specifically, the following chapter describes certain parameters that require attention.

5.1. Finalizing installation for Arcsys RMI, TCP/IP and SOAP API, Arcsys Transfer Server, Arcsys Engine, Arcsys Application Agent and Arcsys REST API

For Arcsys Transfer Server revise particularly:

- Work space sizes: parameters *online_max_size* and *cache_max_size*;
- Number of connections to the TCP/IP API: parameters *referential_**;
- Configuring an external media manager for Cloud or Generic

For Arcsys Engine revise particularly:

- Activation of the various engine specializations (**_ENGINE*) and time ranges associated
- Configuration of mail sending (*MAIL_**).

5.1.1. Registering the Windows service

For a Windows system, where a Windows service must be created, a service installation script must be launched for the server components.

The available installation scripts are:

- `installService_api.cmd` (for Arcsys RMI, TCP/IP and SOAP API)
- `installService_transferServer.cmd` (for Arcsys Transfer Server)
- `installService_transferService.cmd` (for Arcsys Transfer Service)
- `installService_engine.cmd` (for Arcsys Engine)
- `installService_applicationAgent.cmd` (for Arcsys Application Agent)
- `installService_apiRestWS.cmd` (for Arcsys REST API)
- `installService_fileFormatServer.cmd` (for ArcFF format control module)

Note

Note that installing the Windows service for Arcsys Engine does not automatically start the engine. You must either launch it manually after installation or integrate it into a custom start-up script.

Each script installs the corresponding Windows service using `Apache Procrun`. The usage is as follows:

```
<scriptName> [<service_id> [<service_name>]]
```

Where:

- `<service_id>` is the internal identifier of the service.

  If not specified, a default is applied (e.g., arcsys-apis, arcsys-transfer-server, arcsys-engine, etc.).

- `<service_name>` is the display name of the service in the Windows Services list. If not specified, it defaults to the same value as `<service_id>`.

The scripts configure all necessary environment variables and Procrun parameters internally. These typically include:

- The path to the application JAR file (`--Classpath`)
- JVM options such as logging configuration, security key path, and module specific settings (`++JvmOptions`)
- Logging configuration: log path, log prefix, and log level (`--LogPath`, `--LogPrefix`, `--LogLevel`)
- The main class to launch the service (`--StartClass`) and the corresponding stop class

To specify the JVM to use for the service, set the `--Jvm` parameter to the full path of the desired `jvm.dll` (e.g., `--Jvm="C:\Java\jdk-17\bin\server\jvm.dll"`). If not specified, Procrun attempts to detect a suitable JVM automatically.

After running the script, you must reboot the machine and check that the service has started up properly. Otherwise, you must start it up manually.

Under normal conditions, the installation scripts are fully configured and require no manual modification.

If the service fails to start and no error is logged, the Java Virtual Machine may not have been detected correctly by Procrun. This can happen if no `JAVA_HOME` environment variable is defined, or if the system registry does not point to a valid Java installation.

In such cases, you can explicitly specify the path to the JVM using the --Jvm option, as explained earlier in this section.

For further customization or advanced usage of Apache Procrun, refer to the official documentation: https://commons.apache.org/proper/commons-daemon/procrun.html

5.1.2. Using the Linux systemd service

For a Linux system, if you want to use Arcsys as systemd services, the systemd configuration for Arcsys RMI, TCP/IP and SOAP API, Arcsys Transfer Server, Arcsys Engine and Arcsys Application Agent, must be defined. See Arcsys Administration Manual for more information.

5.1.3. Additional configurations for Arcsys Transfer Server

5.1.3.1. Configuring the library path

In the launch script of the transfer server, you may need to specify in the variable `LIB` the location of external libraries needed by the transfer server.

5.1.3.2. Configuring external media managers

The configuration files for external media managers are located in the `etc` directory of the Arcsys Transfer Server installation directory. During the installation process, the default configuration files are created, resulting in the following directory structure:

```
etc/
├── cloud
│   └── s3
│       └── default
│           └── cloud.conf
└── generic
    └── default
        └── generic.conf
```

The last directory level, named `default`, matches to the configuration name of the concerned media manager (identified by upper directories). The configuration name is the name used to associate a configuration to a zone. This name coincides with what is set in the Arcsys Web Agent in the `Configuration` field of the zone. If the default configuration is kept for the Arcsys Transfer Server, the configuration name is then `default`.

Different configurations can be used for a particular manager, allowing it to access to the media manager with various users for example.

A configuration name is limited to 32 characters. Since this configuration name is used as a directory name, we recommend it to be made of alphanumerical characters, "-"

and "_". There is no limit to the number of configurations that can be managed, except for the one resulting from the 32-character name limit.

For more information on the detailed configuration of each media manager, see Arcsys Administration Manual.

Best practices for the configuration of external media managers

When the external media manager does not support concurrent accesses, only one Transfer Server by site should handle archiving requests.

To avoid any ambiguity and any anomaly resulting from incorrect configuration, it is recommended to set a different configuration name for the zones belonging to different sites.

It is also strongly recommended that each of the configurations points to a different storage space so that there is real replication.

5.1.3.3. Configuring the ArcMover Tape Option

Where ArcMover Tape Option is used, you must configure this option.

5.1.3.4. Configuring the ArcHP Option

When ArcHP Option is used, you must configure this option. In particular, you must activate the cluster by setting *`cluster_enabled`* as `true` and registering the transfer server in the relational database.

Important

This registration step is mandatory if you want to use the transfer server in cluster mode (ArcHP Option).

It must be performed before staring up the transfer server in cluster mode: either after installation or after changing the configuration to pass the transfer server in cluster mode.

To register the transfer server in the relational database:

- the Arcsys Database must be installed,
- the TCP/IP API to which the server is connected must be started up,
- the transfer server port must be specified in the relational database.

Once all these conditions are met, launch the `register_transferServer.sh` (or .bat) script found in the installation directory of the transfer server.

5.2. Finalizing installation of the Arcsys Database

You may choose to add specific indexes depending on your use of the product. These indexes are not installed by default to avoid consuming unnecessary database space if they are not needed.

Use case	Table	Column	Example
Search archive by label value (see ticket #72591 in the changelog)	ARO002_ LOT	LOTLIBELLE	For Oracle: `CREATE INDEX ARO002_LOT_LABEL ON ARO002_LOT (OBJHASHVALUE);`
Search document by hash value (see ticket #75015 in the changelog)	ARO005_ OBJECT	OBJHASHVALUE	For Oracle: `CREATE INDEX ARO005_OBJECT_OBJHASH ON ARO005_OBJECT (OBJHASHVALUE);`
For ArcREF Option, to improve performance when *`object.validation.unicity`* is `true`	ARO005_ OBJECT	OBJNAME	For Oracle: `CREATE INDEX ARO005_OBJECT_OBJNAME ON ARO005_OBJECT (OBJNAME);`

Table 5.1. List of recommended indexes by use case

5.3. Finalizing installation of the Arcsys Web Agent

You can check that the Arcsys Web Agent is installed by going to the web URL http://localhost:port/webarcsys/index.jsp. The connection screen should then be displayed.

The port to use depends on the chosen JEE application server. For example, the default is 8080 for JBoss / WildFly, etc.

For performance reasons, when installing the Arcsys Web Agent for the first time, it is recommended to access the following page to generate the Java code for the JSP:

http://localhost:port/webarcsys/PrecompJSP

It is recommended you to revise the configuration of Arcsys Web Agent, in particular:

- Customization: parameters *`WEB_NOM_CLIENT`*, *`WEB_APPLI_NOM`*
- Security: parameter *`USE_LOGIN_AUTOCOMPLETE`*
- Mail settings: parameters *`MAIL_*`*
- Activation of the web interface for ArcAFP, ArcPAK options: parameter *`ENABLE_NATIVE_MOVER`*

For more information, see Arcsys Administration Manual.

Part 3. Updates

1. Introduction

1.1. Procedure

Update an Arcsys module by:

- Launching the `ArcsysUpdater` update script located in the Arcsys Installation and Update Package with the required options
- Then proceed with a final update phase if applicable

1.2. Minimum Arcsys Version for an Update

This version is an STS version. Therefore, updates are only possible from the previous STS version or from 2024.1.LTS version. An LTS version is compatible with both STS and LTS update paths.

For any other combination, please contact Arcsys Technical Support.

1.3. Update Order

There are no specific ordering restrictions when updating the modules.

1.4. Updating by Module

As opposed to the installation procedure, update may be performed only module by module and not on all modules simultaneously.

1.5. Obtaining Prior Information

Important

Updates require preparation. Before updating, you should read the update information and check that it does not require information that must be obtained while Arcsys is still running (for example, you must obtain the list of site codes in the administration web interface).

2. Updating by Module

2.1. Updating Arcsys Database

2.1.1. Prior Shutdown

All specific Arcsys modules, connectors and clients must be stopped.

2.1.2. Prior Backup

Perform a backup of the relational database (structure and data) before updating it.

2.1.3. Manual SQL updates

When upgrading to 5.2 version or later, there are SQL requests which must be executed on the database. Depending on the amount of archives to process, they might take a long time to execute. For this reason, they are not included in the automatic update. These SQL requests are the following:

```
UPDATE MOVER_MEDIA SET MOVDEFLATEHASHVALUE = NULL WHERE MOVDEFLATEHASHVALUE = 'null';
UPDATE MOVER_MEDIA_BKP SET MOVDEFLATEHASHVALUE = NULL WHERE MOVDEFLATEHASHVALUE = 'null';
```

2.1.4. Running the updater

Important

The database must be started up and the update module must be able to access it.

Use the following command line to run the Arcsys updater:

```
./ArcsysUpdater.sh (or ArcsysUpdater.bat on Windows system) -DCOMPONENT=database <Update
parameters>
```

Note

For a Windows system, the updater is not compliant with PowerShell.

The update parameter(s) is/are the following:

- *-DTYPEBASE*: It can be Oracle, MariaDB, SQLServer or PostgreSQL
- *-DJDBC_URL*: JDBC URL of the database, in quotation marks; found in parameter *arcsys.jdbc_url* from the `REFERENTIAL_JDBC.properties` configuration file in the Arcsys Engine. If the URL contains ";" character it needs to be escaped

with a "\" character before it: for example `jdbc:jtds:sqlserver://myhost:1433/sql_lat191\;instance=SQL2012`.

When using a MariaDB Database, you must specify the `serverTimezone` explicitly in the connection string: for example `jdbc:mariadb://myhost.mydomain.com:3306/arc_pcl027?serverTimezone=UTC`

- *`-DJDBC_USER`*: JDBC user of the database; found in parameter *`arcsys.jdbc_user`* from the `REFERENTIAL_JDBC.properties` configuration file in the Arcsys Engine.
- *`-DJDBC_PASSWORD`*: JDBC password (alphanumeric) of the database.
- *`-DARCSYS_INDEX_TABLESPACE_NAME`*: name of the Oracle tablespace of the indexes. On Arcsys versions prior to 4.4 installed initially, the default tablespace for Oracle was used (value USERS); therefore this tablespace name must be used. For later versions of Arcsys, the recommended value was `ARCSYS_DATA` for data and `ARCSYS_INDEX` for indexes. Please check with a DBA where in doubt.

  This parameter can be used in Oracle only; if the tablespace name of the index is not defined by default, i.e., ARCSYS_INDEX
- *`-DRAC`*: Specify if Oracle is installed in RAC mode (`-DRAC=true`). In this case, the sequences will be defined with the ORDER option.

  This parameter can be used in Oracle only, when you want to use RAC mode (Oracle Real Application Clusters).
- *`-DARCSYS_DATA_TABLESPACE_NAME`*: name of the Oracle tablespace of the data.

  This parameter can be used in Oracle only; if the tablespace name of the data is not defined by default, i.e., ARCSYS_DATA
- *`-DTRIM_ARCHIVE_CODES`*: The possible values are `true` or `false`. This parameter is used to specify if, when migrating length of archive codes from 32 to 64 characters (from 2024.1.LTS update), the existing archive codes are automatically trimmed or not. This operation can take a long time depending on the database architecture and the involved volumetrics. That is why it is specified as a parameter. If you specify `false`, the trim of existing archive codes must be done externally before using the product, otherwise the product will misfunction (searching for an archive by its code will not return the archive).

  If the parameter is not specified, the default value taken is `true`.

2.1.5. Checking Indexes

If you want to check the indexes found against those recommended by Infotel, generate a file containing the indexes for the Arcsys Database (an example of the

`listIndexes.sql` script is supplied in the `data/Referential` folder of the updater; if you have insufficient database administration knowledge, please consult Arcsys technical support for more details). Then launch the updater with the option `-DINPUT_INDEX_FILE` =file path of the index generated. The updater logs display the comparison status.

2.2. Updating Arcsys RMI, TCP/IP and SOAP API

2.2.1. Prior Shutdown

All these modules must be stopped:

- Arcsys RMI, TCP/IP and SOAP API
- Arcsys Transfer Server
- Arcsys Engine
- Arcsys Application Agent

2.2.2. Windows: Reinstalling the Windows service

Before performing the update, you must uninstall the Windows service using the `removeService_<Module name>` script found in the module folder. Then reinstall the Windows service using the corresponding `installService_<Module name>` script.

The service identifier may have been changed from its default value during the installation of the Windows service. When changed in the XML information file, this change is automatically considered when removing the Windows service. However, when changed in the command line while launching the `installService_<Module name>` script, it has to be specified the same way when uninstalling.

After running the script, you must reboot the machine.

2.2.3. Linux: Update the systemd service

If you want to use Arcsys as a systemd service, please refer to installation procedure to set it up. See the Arcsys Administration Manual for more information.

2.2.4. Backing up the module automatically

The updater automatically backs up the modules being updated (binaries, libraries, configuration files) in a folder `backup<Year><Month><Day><Hour><Minute>` .

2.2.5. Updating from a version earlier than 5.0.0.0: log4j migration

For an update from a version earlier than 5.0.0.0, switching to version 5.0.0.0 or later updates the log4j library version from 1 to 2, using an XML-format properties file.
The old properties file `log4j.properties` is replaced in the backup directory. The new properties file `log4j2.xml` does not take the values (log level, log file location) informed in the old properties file.

Therefore, after update, you need to inform these values again if they were modified after installation.

2.2.6. Updating from a version earlier than 2024.1.LTS: reference to Java policy file

Starting from Arcsys 2024.1.LTS, the Java policy files are no longer in use.

These files are automatically deleted from the `etc` folder, but the references to these files, made with `-Djava.security.policy=etc/policy`, are not removed from the launch scripts.

The scripts will continue to function even if this reference is retained, but it is advisable to remove it.

To do so, simply delete the line from the script.

2.2.7. Running the updater

Use the following command line to run the Arcsys updater:

```
./ArcsysUpdater.sh (or ArcsysUpdater.bat on Windows system) -DCOMPONENT=api <Update
parameters>
```

Note

For a Windows system, the updater is not compliant with PowerShell.

The update parameter(s) is/are the following:

- *`-DTARGET_FOLDER`*: Component installation directory, in quotation marks if there are spaces.
- *`-DSECRET_KEY_PATH`*: See page 11, « How to encrypt passwords in Arcsys configuration files »: to encrypt the passwords thanks to the encrypt mechanism, this property must be specified in the update command line, giving the path to the secret key file generated by generate-key. The property `secret.key.file` will automatically

be added to the scripts of the component by the updater. You may use different secret key file for each component. For example, you may have one key file for the Arcsys components on a primary site and a different key file on the secondary site.

- *-DJAVA_PATH*: Path to the java executable. This parameter is used for the systemd services definition files. If you do not use systemd to start Arcsys you do not need to specify *-DJAVA_PATH*. If not specified, `/usr/bin/java` is used. Example: *-DJAVA_PATH*=`/usr/lib64/jvm/jre-11-openjdk/bin/java`

2.2.8. Updating from a version earlier than 5.2.0.0 with MariaDB: specification of the timezone

For an update from a version earlier than 5.2.0.0, switching to version 5.2.0.0 or later updates the JDBC driver. This may cause errors of unrecognized timezone when using a MariaDB.

Therefore you must specify the `serverTimezone` explicitly at the end of the connection string, in the *arcsys.jdbc_url* parameter. This parameter is located in the `REFERENTIAL_JDBC.properties` file of each module that accesses the database (Arcsys RMI, TCP/IP and SOAP API, Arcsys REST API, Arcsys Engine and Arcsys Web Agent).

For example: *arcsys.jdbc_url*=`jdbc:mariadb://myhost.mydomain.com:3306/arc_pcl027?serverTimezone=UTC`

2.3. Updating Arcsys REST API

2.3.1. Windows: Reinstalling the Windows service

Before performing the update, you must uninstall the Windows service using the `removeService_<Module name>` script found in the module folder. Then reinstall the Windows service using the corresponding `installService_<Module name>` script.

The service identifier may have been changed from its default value during the installation of the Windows service. When changed in the XML information file, this change is automatically considered when removing the Windows service. However, when changed in the command line while launching the `installService_<Module name>` script, it has to be specified the same way when uninstalling.

After running the script, you must reboot the machine.

2.3.2. Linux: Update the systemd service

If you want to use Arcsys as a systemd service, please refer to installation procedure to set it up. See the Arcsys Administration Manual for more information.

2.3.3. Backing up the module automatically

The updater automatically backs up the modules being updated (binaries, libraries, configuration files) in a folder `backup<Year><Month><Day><Hour><Minute>` .

2.3.4. Updating from a version earlier than 2024.1.LTS: reference to Java policy file

Starting from Arcsys 2024.1.LTS, the Java policy files are no longer in use.

These files are automatically deleted from the `etc` folder, but the references to these files, made with `-Djava.security.policy=etc/policy`, are not removed from the launch scripts.

The scripts will continue to function even if this reference is retained, but it is advisable to remove it.

To do so, simply delete the line from the script.

2.3.5. Running the updater

Use the following command line to run the Arcsys updater:

```
./ArcsysUpdater.sh (or ArcsysUpdater.bat on Windows system) -DCOMPONENT=restWSApi <Update parameters>
```

Note

For a Windows system, the updater is not compliant with PowerShell.

The update parameter(s) is/are the following:

- *`-DTARGET_FOLDER`*: Component installation directory, in quotation marks if there are spaces.
- *`-DSECRET_KEY_PATH`*: See page 11, « How to encrypt passwords in Arcsys configuration files »: to encrypt the passwords thanks to the encrypt mechanism, this property must be specified in the update command line, giving the path to the secret key file generated by generate-key. The property `secret.key.file` will automatically be added to the scripts of the component by the updater. You may use different secret key file for each component. For example, you may have one key file for the Arcsys components on a primary site and a different key file on the secondary site.
- *`-DJAVA_PATH`*: Path to the java executable. This parameter is used for the systemd services definition files. If you do not use systemd to start Arcsys you do not need to

specify `-DJAVA_PATH`. If not specified, `/usr/bin/java` is used. Example: `-DJAVA_PATH=/usr/lib64/jvm/jre-11-openjdk/bin/java`

2.3.6. Updating from a version earlier than 5.2.0.0 with MariaDB: specification of the timezone

For an update from a version earlier than 5.2.0.0, switching to version 5.2.0.0 or later updates the JDBC driver. This may cause errors of unrecognized timezone when using a MariaDB.

Therefore you must specify the `serverTimezone` explicitly at the end of the connection string, in the *`arcsys.jdbc_url`* parameter. This parameter is located in the `REFERENTIAL_JDBC.properties` file of each module that accesses the database (Arcsys RMI, TCP/IP and SOAP API, Arcsys REST API, Arcsys Engine and Arcsys Web Agent).

For example: *`arcsys.jdbc_url`*`=jdbc:mariadb://myhost.mydomain.com:3306/arc_pcl027?serverTimezone=UTC`

2.4. Updating Arcsys Transfer Server

2.4.1. Prior Shutdown

All these modules must be stopped:

- Arcsys Transfer Server
- Arcsys Engine
- Arcsys Application Agent

2.4.2. Windows: Reinstalling the Windows service

Before performing the update, you must uninstall the Windows service using the `removeService_<Module name>` script found in the module folder. Then reinstall the Windows service using the corresponding `installService_<Module name>` script.

The service identifier may have been changed from its default value during the installation of the Windows service. When changed in the XML information file, this change is automatically considered when removing the Windows service. However, when changed in the command line while launching the `installService_<Module name>` script, it has to be specified the same way when uninstalling.

After running the script, you must reboot the machine.

2.4.3. Linux: Update the systemd service

If you want to use Arcsys as a systemd service, please refer to installation procedure to set it up. See the Arcsys Administration Manual for more information.

2.4.4. Backing up the module automatically

The updater automatically backs up the modules being updated (binaries, libraries, configuration files) in a folder `backup<Year><Month><Day><Hour><Minute>` .

2.4.5. Running the updater

Use the following command line to run the Arcsys updater:

```
./ArcsysUpdater.sh (or ArcsysUpdater.bat on Windows system) -DCOMPONENT=transferServer
<Update parameters>
```

Note

For a Windows system, the updater is not compliant with PowerShell.

The update parameter(s) is/are the following:

- *-DTARGET_FOLDER*: Component installation directory, in quotation marks if there are spaces.
- *-DSECRET_KEY_PATH*: See page 11, « How to encrypt passwords in Arcsys configuration files »: to encrypt the passwords thanks to the encrypt mechanism, this property must be specified in the update command line, giving the path to the secret key file generated by generate-key. The property `secret.key.file` will automatically be added to the scripts of the component by the updater. You may use different secret key file for each component. For example, you may have one key file for the Arcsys components on a primary site and a different key file on the secondary site.

2.4.6. Updating the startup script

Important

When updating from a version below 5.0 to version 5.0 or higher, the transfer server startup script is replaced by a new script. If the previous script contains specific instructions, you must resume these

instructions in the new startup script. The old startup script is found in the update backup directory.

2.4.7. Updating the configuration of S3 Cloud media manager (ArcMOVS3 Option)

When new parameters are added to the S3 Cloud configuration, they are only updated in the `default` cloud configuration file (`etc/cloud/s3/default/cloud.conf`). All other configurations are left unchanged. Therefore, the default value of the new parameters applies.

If you do not want the default value to apply, you have to manually update the corresponding configuration file (`etc/cloud/s3/<configuration_name>/cloud.conf`), adding the parameter and its expected value.

You can find below the list of parameters added to the S3 cloud media manager configuration. The table shows the version concerned by these parameters, as well as their default value, that will apply if no manual update is done.

Parameter	Updating from version	Updating to version	Default value
sse_type	below 5.2	5.2 or higher	NONE
request_type	below 2024.1.LTS	2024.1.LTS or higher	PATH

Table 2.1. List of parameters added to S3 Cloud provider configuration

2.4.8. Updating the configuration of key manager

Since version 2024.1.LTS, the key manager configuration file includes a new parameter: *out_of_validity_range_behaviour*. When updating from a version below 5.2.1 to a version 5.2.1 or higher, this parameter has to be added manually. This step can be omitted if the default value of this parameter (`WARNING`) is used.

As a reminder, the key manager configuration file corresponds to the path specified by the *key_managers_conf* parameter in the Arcsys Transfer Server configuration file. You should add the parameter to all the sections of the key manager configuration file corresponding the key managers for which it must be customized.

2.5. Updating Arcsys Engine

2.5.1. Prior Shutdown

All these modules must be stopped:

- Arcsys Engine

- Arcsys Application Agent

2.5.2. Windows: Reinstalling the Windows service

Before performing the update, you must uninstall the Windows service using the `removeService_<Module name>` script found in the module folder. Then reinstall the Windows service using the corresponding `installService_<Module name>` script.

The service identifier may have been changed from its default value during the installation of the Windows service. When changed in the XML information file, this change is automatically considered when removing the Windows service. However, when changed in the command line while launching the `installService_<Module name>` script, it has to be specified the same way when uninstalling.

After running the script, you must reboot the machine.

2.5.3. Linux: Update the systemd service

If you want to use Arcsys as a systemd service, please refer to installation procedure to set it up. See the Arcsys Administration Manual for more information.

2.5.4. Backing up the module automatically

The updater automatically backs up the modules being updated (binaries, libraries, configuration files) in a folder `backup<Year><Month><Day><Hour><Minute>` .

2.5.5. Updating from a version earlier than 5.0.0.0: log4j migration

For an update from a version earlier than 5.0.0.0, switching to version 5.0.0.0 or later updates the log4j library version from 1 to 2, using an XML-format properties file. The old properties file `log4j.properties` is replaced in the backup directory. The new properties file `log4j2.xml` does not take the values (log level, log file location) informed in the old properties file.

Therefore, after update, you need to inform these values again if they were modified after installation.

2.5.6. Updating from a version earlier than 2024.1.LTS: reference to Java policy file

Starting from Arcsys 2024.1.LTS, the Java policy files are no longer in use.

These files are automatically deleted from the `etc` folder, but the references to these files, made with `-Djava.security.policy=etc/policy`, are not removed from the launch scripts.

The scripts will continue to function even if this reference is retained, but it is advisable to remove it.

To do so, simply delete the line from the script.

2.5.7. Running the updater

Use the following command line to run the Arcsys updater:

```
./ArcsysUpdater.sh (or ArcsysUpdater.bat on Windows system) -DCOMPONENT=engine <Update
parameters>
```

Note

For a Windows system, the updater is not compliant with PowerShell.

The update parameter(s) is/are the following:

- *-DTARGET_FOLDER*: Component installation directory, in quotation marks if there are spaces.
- *-DSECRET_KEY_PATH*: See page 11, « How to encrypt passwords in Arcsys configuration files »: to encrypt the passwords thanks to the encrypt mechanism, this property must be specified in the update command line, giving the path to the secret key file generated by generate-key. The property `secret.key.file` will automatically be added to the scripts of the component by the updater. You may use different secret key file for each component. For example, you may have one key file for the Arcsys components on a primary site and a different key file on the secondary site.
- *-DLICENSE_PATH*: Path to the license file provided by Infotel. During an update, if this parameter is already present in the scripts, its value will not be modified. The license file path can be modified manually in the scripts.
- *-DJAVA_PATH*: Path to the java executable. This parameter is used for the systemd services definition files. If you do not use systemd to start Arcsys you do not need to specify *-DJAVA_PATH*. If not specified, `/usr/bin/java` is used. Example: *-DJAVA_PATH*=`/usr/lib64/jvm/jre-11-openjdk/bin/java`

2.6. Updating Arcsys Application Agent

2.6.1. Prior Shutdown

All these modules must be stopped:

- Arcsys Application Agent

2.6.2. Windows: Reinstalling the Windows service

Before performing the update, you must uninstall the Windows service using the `removeService_<Module name>` script found in the module folder. Then reinstall the Windows service using the corresponding `installService_<Module name>` script.

The service identifier may have been changed from its default value during the installation of the Windows service. When changed in the XML information file, this change is automatically considered when removing the Windows service. However, when changed in the command line while launching the `installService_<Module name>` script, it has to be specified the same way when uninstalling.

After running the script, you must reboot the machine.

2.6.3. Linux: Update the systemd service

If you want to use Arcsys as a systemd service, please refer to installation procedure to set it up. See the Arcsys Administration Manual for more information.

2.6.4. Backing up the module automatically

The updater automatically backs up the modules being updated (binaries, libraries, configuration files) in a folder `backup<Year><Month><Day><Hour><Minute>` .

2.6.5. Updating from a version earlier than 5.0.0.0: log4j migration

For an update from a version earlier than 5.0.0.0, switching to version 5.0.0.0 or later updates the log4j library version from 1 to 2, using an XML-format properties file. The old properties file `log4j.properties` is replaced in the backup directory. The new properties file `log4j2.xml` does not take the values (log level, log file location) informed in the old properties file.

Therefore, after update, you need to inform these values again if they were modified after installation.

2.6.6. Updating from a version earlier than 2024.1.LTS: reference to Java policy file

Starting from Arcsys 2024.1.LTS, the Java policy files are no longer in use.

These files are automatically deleted from the `etc` folder, but the references to these files, made with `-Djava.security.policy=etc/policy`, are not removed from the launch scripts.

The scripts will continue to function even if this reference is retained, but it is advisable to remove it.

To do so, simply delete the line from the script.

2.6.7. Windows

If the application agent is used with ArcPAK Option or ArcAFP Option options, before update, you must check there are no DLL files(s) in the directory root of the application agent (at the same level as the startup scripts). If there are, you must delete this(these) files(s).

2.6.8. Running the updater

Use the following command line to run the Arcsys updater:

```
./ArcsysUpdater.sh (or ArcsysUpdater.bat on Windows system) -DCOMPONENT=applicationAgent
<Update parameters>
```

Note

For a Windows system, the updater is not compliant with PowerShell.

The update parameter(s) is/are the following:

- *`-DTARGET_FOLDER`*: Component installation directory, in quotation marks if there are spaces.
- *`-DPRIORITY_SITE_LIST`*: List of site codes, separated by commas. The order of the list is important: the engines of the first site in the list take priority for processing the archiving or retrieval requests for this agent. All the engines of each site have the same priority.
- *`-DLOCAL_TRANSFER_SERVICE_CODES`*: List of codes of "local" transfer servers for this application agent. An application agent transfers its data to Arcsys either via a transfer service or via a transfer server (which is then used as a transfer service). In both cases, the transfer module code must be entered.
- *`-DSECRET_KEY_PATH`*: See page 11, « How to encrypt passwords in Arcsys configuration files »: to encrypt the passwords thanks to the encrypt mechanism, this property must be specified in the update command line, giving the path to the secret key file generated by generate-key. The property `secret.key.file` will automatically be added to the scripts of the component by the updater. You may use different

secret key file for each component. For example, you may have one key file for the Arcsys components on a primary site and a different key file on the secondary site.

- *-DJAVA_PATH*: Path to the java executable. This parameter is used for the systemd services definition files. If you do not use systemd to start Arcsys you do not need to specify *-DJAVA_PATH*. If not specified, `/usr/bin/java` is used. Example: *-DJAVA_PATH*=`/usr/lib64/jvm/jre-11-openjdk/bin/java`

2.7. Updating Arcsys Transfer Service

2.7.1. Prior Shutdown

All these modules must be stopped:

- Arcsys Transfer Service

2.7.2. Windows: Reinstalling the Windows service

Before performing the update, you must uninstall the Windows service using the `removeService_<Module name>` script found in the module folder. Then reinstall the Windows service using the corresponding `installService_<Module name>` script.

The service identifier may have been changed from its default value during the installation of the Windows service. When changed in the XML information file, this change is automatically considered when removing the Windows service. However, when changed in the command line while launching the `installService_<Module name>` script, it has to be specified the same way when uninstalling.

After running the script, you must reboot the machine.

2.7.3. Linux: Update the systemd service

If you want to use Arcsys as a systemd service, please refer to installation procedure to set it up. See the Arcsys Administration Manual for more information.

2.7.4. Backing up the module automatically

The updater automatically backs up the modules being updated (binaries, libraries, configuration files) in a folder `backup<Year><Month><Day><Hour><Minute>` .

2.7.5. Running the updater

Use the following command line to run the Arcsys updater:

```
./ArcsysUpdater.sh (or ArcsysUpdater.bat on Windows system) -DCOMPONENT=transferService
<Update parameters>
```

Note

For a Windows system, the updater is not compliant with PowerShell.

The update parameter(s) is/are the following:

- *-DTARGET_FOLDER*: Component installation directory, in quotation marks if there are spaces.
- *-DCOMPONENT_CODE*: Desired code of the module. This code must be unique compared to other components of same type.

  This parameter is mandatory only for an update from a version prior to the version: 5.0.0.0
- *-DTRANSFER_SERVER_LIST*: List of addresses on which to contact the transfer server on startup of the transfer service, separated by commas. Each address has the format `<Transfer server host>:<Transfer server port>`.

  This parameter is mandatory only for an update from a version prior to the version: 5.0.0.0
- *-DSECRET_KEY_PATH*: See page 11, « How to encrypt passwords in Arcsys configuration files »: to encrypt the passwords thanks to the encrypt mechanism, this property must be specified in the update command line, giving the path to the secret key file generated by generate-key. The property `secret.key.file` will automatically be added to the scripts of the component by the updater. You may use different secret key file for each component. For example, you may have one key file for the Arcsys components on a primary site and a different key file on the secondary site.

2.8. Updating Arcsys Web (when changing application server)

If you want to change application servers, for example, because an application server is no longer supported (e.g. in version 5.0, JOnAS is no longer supported), you must perform a special update procedure.

Follow this procedure:

- Follow the Arcsys web update phase for the new application server chosen.

2.9. Updating Arcsys Web

Updating is not completely automatic and takes place in several stages.

2.9.1. Prior shutdown

Arcsys Web Agent must be stopped before updating.

2.9.2. Backing up the module

The updater automatically backs up a part of the module being updated (libraries, configuration files) in a `backup<Year><Month><Day><Hour><Minute>` folder. However, web application files should be backed up manually (WAR file).

2.9.3. Updating from a version earlier than 5.0.0.0: log4j migration

For an update from a version earlier than 5.0.0.0, switching to version 5.0.0.0 or later updates the log4j library version from 1 to 2, using an XML-format properties file.
The old properties file `log4j.properties` is replaced in the backup directory. The new properties file `log4j2.xml` does not take the values (log level, log file location) informed in the old properties file.

Therefore, after update, you need to inform these values again if they were modified after installation.

2.9.4. Updating from a version earlier than 5.2.0.0: changing the datasource part

From Arcsys 5.2, there is no more EJB data source and the link to the database is made via JDBC, in the same way as the following modules: Arcsys Engine, Arcsys RMI, TCP/IP and SOAP API or Arcsys REST API.

To change the datasource part:

1. Remove the `REFERENTIAL_EJB.properties` file from the `etc` folder
2. Add an adequate `REFERENTIAL_JDBC.properties` file in the `etc` folder. You may copy this file from the Arcsys Engine, Arcsys RMI, TCP/IP and SOAP API or Arcsys REST API modules
3. Change the Java environment variable `arcsys.bdd` to the path of the `REFERENTIAL_JDBC.properties` file and not the `REFERENTIAL_EJB.properties` file
4. Undeploy the initial Arcsys data source

2.9.5. Running the updater

Use the following command line to run the Arcsys updater:

```
./ArcsysUpdater.sh (or ArcsysUpdater.bat on Windows system) -DCOMPONENT=webAgent <Update
parameters>
```

Note

For a Windows system, the updater is not compliant with PowerShell.

The update parameter(s) is/are the following:

- *-DTARGET_FOLDER*: Component installation directory, in quotation marks if there are spaces.

2.9.6. Deploying the application

2.9.6.1. WAR file to deploy

The WAR file to deploy is the `webarcsys.war` file found in the `war` folder of the Arcsys Web Agent. If this file is already deployed (for interrupted installation or update), we recommend to uninstall it first. When updating from a version prior to 5.2.0.0, uninstall the EAR file(s) first.

2.9.6.2. Deployment procedure

The deployment procedure depends on the application server used.

On JBoss / WildFly you must deploy the WAR file in the `deployments` folder of the JBoss / WildFly instance (for example, `standalone/deployments`). You can also use the `jboss-cli` utility to deploy the file with the `deploy` command.

2.9.7. Java environment variables: overview

The following environment variables are currently required for Arcsys Web Agent operation:

- *arcsys.alert.manager.factory*: Java class to instantiate the alert manager (internal parameter). The value must be the following: `com.infotel.arcsys.alert.manager.ArcsysWebAgentAlertManagerFactory`
- *log4j.configurationFile*: Full path of the `log4j2.xml` properties file (found by default in the `etc` folder of the module)
- *arcsys.trace.folder*: Full path of the directory which must contain the trace files.
- *arcsys.bdd*: Full path of the `REFERENTIAL_JDBC.properties` properties file (found by default in the `etc` folder of the module)

- *arcsys.web.config*: Full path of the WEBAGENT_configuration.properties properties file (found by default in the etc folder of the module)

- *arcsys.third.party.search.engine*: Full path of the third party search engine properties file THIRD_PARTY_SEARCH_ENGINE.properties (found by default in the etc folder of the module)

- *secret.key.file*: Full path of the secret key file to encrypt the passwords thanks to the encrypt mechanism.

- *java.awt.headless=true*: Enables "headless mode" mode to view the web interface graphics in the absence of server X11.

  Requirement If you want to use the X11 graphics server on a Unix machine, you must first install the libxp6 and libxtst6 libraries.

2.9.8. Suggested update tool

In JBoss / WildFly, the Java environment variables can be updated by the jboss-cli utility (using system-property). This script is run with the application server started up.

Here is an example:

```
$ ./bin/jboss-cli.sh --connect controller=IP_ADDRESS
                    [standalone@IP_ADDRESS:9990 /] /system-property=log4j.
configurationFile:add(value=/opt/Infotel/ArchivingProduct/ArcsysWebAgent/etc/log4j2.xml)
                    [standalone@IP_ADDRESS:9990 /] /system-property=arcsys.bdd:add(value=/
opt/Infotel/ArchivingProduct/ArcsysWebAgent/etc/REFERENTIAL_JDBC.properties)
                    [standalone@IP_ADDRESS:9990 /] /system-property=arcsys.web.
config:add(value=/opt/Infotel/ArchivingProduct/ArcsysWebAgent/etc/WEBAGENT_configuration.
properties)
                    [standalone@IP_ADDRESS:9990 /] /system-property=arcsys.alert.manager.
factory:add(value=com.infotel.arcsys.alert.manager.ArcsysWebAgentAlertManagerFactory)
                    [standalone@IP_ADDRESS:9990 /] /system-property=arcsys.third.
party.search.engine:add(value=/opt/Infotel/ArchivingProduct/ArcsysWebAgent/etc/TH
IRD_PARTY_SEARCH_ENGINE.properties)
                    [standalone@IP_ADDRESS:9990 /] /system-property=secret.key.file:
add(value=/opt/Infotel/ArchivingProduct/secretkeyfile)
```

2.9.9. Updating from a version earlier than 5.2.0.0 with MariaDB: specification of the timezone

For an update from a version earlier than 5.2.0.0, switching to version 5.2.0.0 or later updates the JDBC driver. This may cause errors of unrecognized timezone when using a MariaDB.

Therefore you must specify the serverTimezone explicitly at the end of the connection string, in the *arcsys.jdbc_url* parameter. This parameter is located in the REFERENTIAL_JDBC.properties file of each module that accesses the database

(Arcsys RMI, TCP/IP and SOAP API, Arcsys REST API, Arcsys Engine and Arcsys Web Agent).

For example: *arcsys.jdbc_url*=jdbc:mariadb://myhost.mydomain.com:3306/arc_pcl027?serverTimezone=UTC

2.10. Updating Arcsys Auto-Archive Agent

2.10.1. Backing up the module automatically

The updater automatically backs up the modules being updated (binaries, libraries, configuration files) in a folder backup<Year><Month><Day><Hour><Minute> .

2.10.2. Updating from a version earlier than 5.0.0.0: log4j migration

For an update from a version earlier than 5.0.0.0, switching to version 5.0.0.0 or later updates the log4j library version from 1 to 2, using an XML-format properties file.
The old properties file log4j.properties is replaced in the backup directory. The new properties file log4j2.xml does not take the values (log level, log file location) informed in the old properties file.

Therefore, after update, you need to inform these values again if they were modified after installation.

2.10.3. Running the updater

Use the following command line to run the Arcsys updater:

```
./ArcsysUpdater.sh (or ArcsysUpdater.bat on Windows system) -DCOMPONENT=autoarchiveAgent
<Update parameters>
```

Note

For a Windows system, the updater is not compliant with PowerShell.

The update parameter(s) is/are the following:

- *-DTARGET_FOLDER*: Component installation directory, in quotation marks if there are spaces.
- *-DSECRET_KEY_PATH*: See page 11, « How to encrypt passwords in Arcsys configuration files »: to encrypt the passwords thanks to the encrypt mechanism, this property must be specified in the update command line, giving the path to the secret key file generated by generate-key. The property secret.key.file will automatically

be added to the scripts of the component by the updater. You may use different secret key file for each component. For example, you may have one key file for the Arcsys components on a primary site and a different key file on the secondary site.

2.11. Updating CopyRequestManager

2.11.1. Backing up the module automatically

The updater automatically backs up the modules being updated (binaries, libraries, configuration files) in a folder `backup<Year><Month><Day><Hour><Minute>` .

2.11.2. Running the updater

Use the following command line to run the Arcsys updater:

```
 ./ArcsysUpdater.sh (or ArcsysUpdater.bat on Windows system) -DCOMPONENT=
copyRequestsManager <Update parameters>
```

Note

For a Windows system, the updater is not compliant with PowerShell.

The update parameter(s) is/are the following:

- *`-DTARGET_FOLDER`*: Component installation directory, in quotation marks if there are spaces.

2.12. Updating ArcFF format control module

2.12.1. Linux: Update the systemd service

If you want to use Arcsys as a systemd service, please refer to installation procedure to set it up. See the Arcsys Administration Manual for more information.

2.12.2. Backing up the module automatically

The updater automatically backs up the modules being updated (binaries, libraries, configuration files) in a folder `backup<Year><Month><Day><Hour><Minute>` .

2.12.3. Running the updater

Use the following command line to run the Arcsys updater:

```
./ArcsysUpdater.sh (or ArcsysUpdater.bat on Windows system) -DCOMPONENT=fileFormatServer
<Update parameters>
```

Note

For a Windows system, the updater is not compliant with PowerShell.

The update parameter(s) is/are the following:

- *-DTARGET_FOLDER*: Component installation directory, in quotation marks if there are spaces.
- *-DSECRET_KEY_PATH*: See page 11, « How to encrypt passwords in Arcsys configuration files »: to encrypt the passwords thanks to the encrypt mechanism, this property must be specified in the update command line, giving the path to the secret key file generated by generate-key. The property `secret.key.file` will automatically be added to the scripts of the component by the updater. You may use different secret key file for each component. For example, you may have one key file for the Arcsys components on a primary site and a different key file on the secondary site.
- *-DJAVA_PATH*: Path to the java executable. This parameter is used for the systemd services definition files. If you do not use systemd to start Arcsys you do not need to specify *-DJAVA_PATH*. If not specified, `/usr/bin/java` is used. Example: *-DJAVA_PATH*=`/usr/lib64/jvm/jre-11-openjdk/bin/java`

2.13. Updating ArcsysFsComparator File systems comparator

2.13.1. Backing up the module automatically

The updater automatically backs up the modules being updated (binaries, libraries, configuration files) in a folder `backup<Year><Month><Day><Hour><Minute>` .

2.13.2. Updating from a version earlier than 5.0.0.0: log4j migration

For an update from a version earlier than 5.0.0.0, switching to version 5.0.0.0 or later updates the log4j library version from 1 to 2, using an XML-format properties file.
The old properties file `log4j.properties` is replaced in the backup directory. The new properties file `log4j2.xml` does not take the values (log level, log file location) informed in the old properties file.

Therefore, after update, you need to inform these values again if they were modified after installation.

2.13.3. Running the updater

Use the following command line to run the Arcsys updater:

```
./ArcsysUpdater.sh (or ArcsysUpdater.bat on Windows system) -DCOMPONENT=fsComparator
<Update parameters>
```

Note

For a Windows system, the updater is not compliant with PowerShell.

The update parameter(s) is/are the following:

- *-DTARGET_FOLDER*: Component installation directory, in quotation marks if there are spaces.
- *-DSECRET_KEY_PATH*: See page 11, « How to encrypt passwords in Arcsys configuration files »: to encrypt the passwords thanks to the encrypt mechanism, this property must be specified in the update command line, giving the path to the secret key file generated by generate-key. The property `secret.key.file` will automatically be added to the scripts of the component by the updater. You may use different secret key file for each component. For example, you may have one key file for the Arcsys components on a primary site and a different key file on the secondary site.

2.14. Updating Arcsys standard Clients

2.14.1. Backing up the module automatically

The updater automatically backs up the modules being updated (binaries, libraries, configuration files) in a folder `backup<Year><Month><Day><Hour><Minute>` .

2.14.2. Updating from a version earlier than 5.0.0.0: log4j migration

For an update from a version earlier than 5.0.0.0, switching to version 5.0.0.0 or later updates the log4j library version from 1 to 2, using an XML-format properties file.
The old properties file `log4j.properties` is replaced in the backup directory. The new properties file `log4j2.xml` does not take the values (log level, log file location) informed in the old properties file.

Therefore, after update, you need to inform these values again if they were modified after installation.

2.14.3. Updating from a version earlier than 2024.1.LTS: reference to Java policy file

Starting from Arcsys 2024.1.LTS, the Java policy files are no longer in use.

These files are automatically deleted from the `etc` folder, but the references to these files, made with `-Djava.security.policy=etc/policy`, are not removed from the launch scripts.

The scripts will continue to function even if this reference is retained, but it is advisable to remove it.

To do so, simply delete the line from the script.

2.14.4. Running the updater

Use the following command line to run the Arcsys updater:

```
./ArcsysUpdater.sh (or ArcsysUpdater.bat on Windows system) -DCOMPONENT=standardClients
<Update parameters>
```

Note

For a Windows system, the updater is not compliant with PowerShell.

The update parameter(s) is/are the following:

- *`-DTARGET_FOLDER`*: Component installation directory, in quotation marks if there are spaces.
- *`-DSECRET_KEY_PATH`*: See page 11, « How to encrypt passwords in Arcsys configuration files »: to encrypt the passwords thanks to the encrypt mechanism, this property must be specified in the update command line, giving the path to the secret key file generated by generate-key. The property `secret.key.file` will automatically be added to the scripts of the component by the updater. You may use different secret key file for each component. For example, you may have one key file for the Arcsys components on a primary site and a different key file on the secondary site.

2.15. Updating ArcsysBatchs batch module

2.15.1. Backing up the module automatically

The updater automatically backs up the modules being updated (binaries, libraries, configuration files) in a folder `backup<Year><Month><Day><Hour><Minute>` .

2.15.2. Running the updater

Use the following command line to run the Arcsys updater:

```
./ArcsysUpdater.sh (or ArcsysUpdater.bat on Windows system) -DCOMPONENT=batchs <Update
parameters>
```

Note

For a Windows system, the updater is not compliant with PowerShell.

The update parameter(s) is/are the following:

- *-DTARGET_FOLDER*: Component installation directory, in quotation marks if there are spaces.

2.16. Updating BatchReporting

2.16.1. Backing up the module automatically

The updater automatically backs up the modules being updated (binaries, libraries, configuration files) in a folder `backup<Year><Month><Day><Hour><Minute>` .

2.16.2. Running the updater

Use the following command line to run the Arcsys updater:

```
./ArcsysUpdater.sh (or ArcsysUpdater.bat on Windows system) -DCOMPONENT=batchReporting
<Update parameters>
```

Note

For a Windows system, the updater is not compliant with PowerShell.

The update parameter(s) is/are the following:

- *-DTARGET_FOLDER*: Component installation directory, in quotation marks if there are spaces.
- *-DSECRET_KEY_PATH*: See page 11, « How to encrypt passwords in Arcsys configuration files »: to encrypt the passwords thanks to the encrypt mechanism, this property must be specified in the update command line, giving the path to the secret key file generated by generate-key. The property `secret.key.file` will automatically be added to the scripts of the component by the updater. You may use different secret key file for each component. For example, you may have one key file for the Arcsys components on a primary site and a different key file on the secondary site.

2.17. Updating MetadataReplacement

2.17.1. Backing up the module automatically

The updater automatically backs up the modules being updated (binaries, libraries, configuration files) in a folder `backup<Year><Month><Day><Hour><Minute>` .

2.17.2. Running the updater

Use the following command line to run the Arcsys updater:

```
 ./ArcsysUpdater.sh (or ArcsysUpdater.bat on Windows system) -DCOMPONENT=
metadataReplacement <Update parameters>
```

Note

For a Windows system, the updater is not compliant with PowerShell.

The update parameter(s) is/are the following:

- *`-DTARGET_FOLDER`*: Component installation directory, in quotation marks if there are spaces.
- *`-DSECRET_KEY_PATH`*: See page 11, « How to encrypt passwords in Arcsys configuration files »: to encrypt the passwords thanks to the encrypt mechanism, this property must be specified in the update command line, giving the path to the secret key file generated by generate-key. The property `secret.key.file` will automatically be added to the scripts of the component by the updater. You may use different secret key file for each component. For example, you may have one key file for the Arcsys components on a primary site and a different key file on the secondary site.

2.18. Updating StartRetentionDateAssigner

2.18.1. Backing up the module automatically

The updater automatically backs up the modules being updated (binaries, libraries, configuration files) in a folder `backup<Year><Month><Day><Hour><Minute>` .

2.18.2. Running the updater

Use the following command line to run the Arcsys updater:

```
./ArcsysUpdater.sh (or ArcsysUpdater.bat on Windows system) -DCOMPONENT=
startRetentionDateAssigner <Update parameters>
```

Note

For a Windows system, the updater is not compliant with PowerShell.

The update parameter(s) is/are the following:

- *-DTARGET_FOLDER*: Component installation directory, in quotation marks if there are spaces.
- *-DSECRET_KEY_PATH*: See page 11, « How to encrypt passwords in Arcsys configuration files »: to encrypt the passwords thanks to the encrypt mechanism, this property must be specified in the update command line, giving the path to the secret key file generated by generate-key. The property `secret.key.file` will automatically be added to the scripts of the component by the updater. You may use different secret key file for each component. For example, you may have one key file for the Arcsys components on a primary site and a different key file on the secondary site.

2.19. Updating ClassAssigner

2.19.1. Backing up the module automatically

The updater automatically backs up the modules being updated (binaries, libraries, configuration files) in a folder `backup<Year><Month><Day><Hour><Minute>` .

2.19.2. Running the updater

Use the following command line to run the Arcsys updater:

```
./ArcsysUpdater.sh (or ArcsysUpdater.bat on Windows system) -DCOMPONENT=classAssigner
<Update parameters>
```

Note

For a Windows system, the updater is not compliant with PowerShell.

The update parameter(s) is/are the following:

- *-DTARGET_FOLDER*: Component installation directory, in quotation marks if there are spaces.
- *-DSECRET_KEY_PATH*: See page 11, « How to encrypt passwords in Arcsys configuration files »: to encrypt the passwords thanks to the encrypt mechanism, this

property must be specified in the update command line, giving the path to the secret key file generated by generate-key. The property `secret.key.file` will automatically be added to the scripts of the component by the updater. You may use different secret key file for each component. For example, you may have one key file for the Arcsys components on a primary site and a different key file on the secondary site.

2.20. Updating Signature validator module

2.20.1. Backing up the module automatically

The updater automatically backs up the modules being updated (binaries, libraries, configuration files) in a folder `backup<Year><Month><Day><Hour><Minute>` .

2.20.2. Running the updater

Use the following command line to run the Arcsys updater:

```
./ArcsysUpdater.sh (or ArcsysUpdater.bat on Windows system) -DCOMPONENT=signatureValidator
<Update parameters>
```

Note

For a Windows system, the updater is not compliant with PowerShell.

The update parameter(s) is/are the following:

- *`-DTARGET_FOLDER`*: Component installation directory, in quotation marks if there are spaces.

2.21. Updating ArcREF Injector

2.21.1. Backing up the module automatically

The updater automatically backs up the modules being updated (binaries, libraries, configuration files) in a folder `backup<Year><Month><Day><Hour><Minute>` .

2.21.2. Running the updater

Use the following command line to run the Arcsys updater:

```
./ArcsysUpdater.sh (or ArcsysUpdater.bat on Windows system) -DCOMPONENT=refInjector
<Update parameters>
```

Note

For a Windows system, the updater is not compliant with PowerShell.

The update parameter(s) is/are the following:

- *-DTARGET_FOLDER*: Component installation directory, in quotation marks if there are spaces.

2.22. Rolling Back in the Event of Error

2.22.1. Overview

For some reason, the update may end in error (typical message: `BUILD FAILED` at the end of the script). In this case, before relaunching the update, you should restore the module to its original status before update.

2.22.2. Procedure by Module

2.22.2.1. Arcsys Database

For the Arcsys Database, restore the data with the backup made before starting the update.

2.22.2.2. Arcsys Web Agent

For the Arcsys Web Agent, redeploy the WAR backed up before starting the update.

2.22.2.3. ArcsysBatchs batch module

For ArcsysBatchs batch module, restore the data with the backup made before starting the update.

2.22.2.4. All modules except Arcsys Database, Arcsys Web Agent, and ArcsysBatchs batch module

For other modules, you must use the Arcsys updater `Restore` option:

```
 >./ArcsysUpdater.sh ( or ArcsysUpdater.bat on Windows) -DCOMPONENT=<Component> -DACTION=
restore -DRESTORE_DATE=<Date with the pattern yyyyMMddHHmm> -DTARGET_FOLDER=<Module
 installation folder>
```

Glossary

Access Zone

An access zone is an independent entity within Arcsys that defines a controlled network area from which resources can be accessed. These entities can then be attached to permissions (at the repository, collection, lot, or class level) to restrict or grant access based on the client's IP address when authenticating to the Arcsys REST API, the Arcsys Web Agent or ArcWeb Module.

API *(Application Programming Interface)*

The APIs provided by Arcsys enable the product holder to fully customize a new application or user interface according to the specific ergonomic needs of their use case. Arcsys exposes several types of APIs:

- REST APIs are the recommended interface. They offer broad coverage of Arcsys's functionalities, including administration, operations, archiving, search, and archive retrieval.
- Legacy APIs based on RMI and SOAP protocols are still available for compatibility purposes but are deprecated and should no longer be used in new developments.

Application Agent

There are two different types of agents at archiving level: application interface agents and user interface agents. An **application agent** can archive all the objects specific to an application (files, RDBMS table records, etc.), whereas a **web agent** performs both administration functions and manual archiving functions initiated by the user.

Archiving By Reference

Archiving by reference is a method in which data remains in its original storage location when added to an archive system, and the system generates references and metadata entries for the files. Eventually, the files are transferred to the archive system's defined storage using the copy and migration mechanism.

Archive Restitution

Archive restitution is the return and transfer of archived documents to their originator, or to a duly appointed person or organization. An Archive Restitution is in Arcsys an Archive Retrieval operation that ends with a Destruction. An Archive restitution operation can only be created through the appropriate operation in the REST API, or by using ArcEP module.
See Also **Archive Retrieval**, **Destruction**.

Archive Retrieval

Archive retrieval is an operation that makes a copy of a record available to a record requester. This term takes precedence over the term *restore*, which has another meaning at archiving level (restore in the sense of handing back the documents to the organization that created them or to its representatives, then destroying them). Archive retrieval can be

complete (misleadingly called a "complete retrieval") or partial (*Partial Archive Retrieval*, misleadingly called a "partial retrieval").
See Also **Archive Restitution**.

Arcsys
ERM published by Infotel. Arcsys refers to both the Arcsys Core product and all of its connectors and options.

Arcsys Connector
An Arcsys connector is an operational module generally requiring an additional license used to interface with an external software package (ECM, ERP, Mail) for archiving and/or archive retrieval to and from Arcsys.

Arcsys Core
The Arcsys Core represents all "essential" Arcsys modules, which are: Arcsys Database, the Arcsys RMI, TCP/IP and SOAP API, the Arcsys REST API, the Arcsys Transfer Server, the Arcsys Transfer Service, the Arcsys Engine, the Arcsys Web Agent, the Arcsys Application Agent, the Arcsys Auto-Archive Agent, the ArcFF format control module, the CopyRequestManager, the Arcsys standard Clients, the ArcsysFsComparator File systems comparator, the ArcProofFolder Proof Folder module and the ArcsysBatchs batch module.
See Also **Arcsys**.

Arcsys Engine
Central archiving platform on which synchronous and asynchronous archiving, indexing and retrieval processes operate. The engine can spread threads over multiple processors. This guarantees dialogue and traceability between the agents that are associated to it.

Arcsys Option
Arcsys options are added to the Arcsys Core for additional functionalities. They do not necessarily require an additional architectural module. They may be subject to a separate license. The main options are:

- ArcAFP Option (AFP format management)
- ArcMover Tape Option (media manager managing file systems and tape libraries)
- ArcIP (record ingestion)
- ArcEP (record extractor)
- ArcPAK Option (record compression on ArcMover and native ingestion of compressed files)
- ArcRFT Option (full text search)
- ArcSIGN Option (internal digital signature generation) and ArcVERIF (external digital signature verification)

- ArcCrypt Option (encryption of data at rest)
- ArcCFN (digital vault)
- ArcREF Option (record ingestion by reference)
- ArcMOVS3 Option (media manager allowing to archive and retrieve data on any Cloud media compatible with the Amazon S3 REST API)

Attestation policy

An attestation policy allows to define which type of attestation can be generated as well as a set of parameters concerning their generation.

Classification Scheme

A classification scheme in archiving and digital preservation refers to an organized framework for categorizing records and archival materials based on a hierarchical structure. It facilitates systematic retrieval, management, and preservation of information. In the context of Arcsys, the classification scheme is defined as the structural entity that contains a hierarchy of classes. These classes are used for organizing archives and records and for implementing specific archival policies such as retention schedules and format management. Within Arcsys, a classification scheme is linked to a specific repository, providing an organizational backbone for multiple collections. It also serves as a navigational tool for end users, enabling them to explore archives through the hierarchical structure of classes, alongside navigation by repository and collection.

Collection

Set of rules that a record must comply with. The collection is defined via the Web agent or Arcsys API, and comprises information contained in the relational database tables. A collection always refers to two rules: one concerning the **storage policy** and one relating to the **indexing mask**. A collection is assigned to the record on the initial record request.
See Also **Storage policy**, **Indexing mask**.

Deletion

MOREQ2010 provides the following definition for this concept: the act of deleting data from the relational database so that no trace remains. Generally speaking, an entity can only be deleted if is not used in a stored record. Otherwise, it can only be destroyed and not deleted, thus leaving a residual entity.
See Also **Destruction**.

Destruction

Irreversible action that deletes the documents by applying disposal criteria. It can be associated with the retention of residual information in the relational database.

Disposal

Outcome of archived documents when the retention period ends, i.e. generally, destruction or transfer.
See Also **Destruction**, **Transfer**.

Disposal due date (or retention end date)
Scheduled end of retention date.

Disposal Hold
Arcsys can be used to place a "disposal hold" on one or more lots archived in the application. This prevents certain sensitive operations, such as transitioning the lots to end-of-life status or migrating them to a different storage medium. All other operations remain authorized. The disposal hold guarantees that no irreversible change affecting the archival integrity or status of the lot can occur while the hold is active.

Electronic Attestation
Document produced to attest that an action or an electronic transaction has occurred.

Envelope
Arcsys groups documents stored in the system in envelopes, either created by Arcsys during the archiving process (in this case, files in TAR format), or created prior to Arcsys processing by the user or third-party processes (*native envelopes* in AFP or ZIP format, for example). The representation of an envelope in the Arcsys Database is called a **logical envelope**. Its technical implementation is also called *MoverReference*. Last but not least, the representation of information of where the envelope is physically stored in the optional ArcMover module is called *MoverMedia*.

Event
In Arcsys, a retention schedule can associate the start of record retention with an event with a known or unknown date. This event, created in an Arcsys repository, can thus be attached to a number of different retention schedules.
See Also **Retention schedule**.

Feature preview
A Preview status on a feature enables early access to non-production features, allowing users to explore and provide feedback for improvement.

Features in Preview status should not be used in production environment, as they are not fully implemented yet.

Fixity
The quality of a document that has not been subject to intentional or accidental destruction, alteration or modification.

Format policy
A format policy is used to define a set of rules concerning format checks for a given file type. These rules are used to specify the action that will be performed, the expected results of these actions, as well as the error cases, triggering archiving failure.

Hash value
Also called an "integrity certificate" in cryptography, the hash value is the digest of a message which guarantees a practically unique result that is impossible to reverse calculate. The most commonly used algorithms are MD5 (128 bits), SHA-1 (160 bits), SHA256 (256

bits) and SHA512 (512 bits). Arcsys includes a module that is capable of dynamically calling several algorithms. The choice of an algorithm type is valid for all archived objects within the same Arcsys product version; compatibility with algorithms from the previous version is guaranteed. The associated term *hash function* is also used.

Indexing mask

As is the case with the storage policy, an indexing mask is a rule that is referenced by a collection. An indexing mask can be referenced by several collections. An indexing mask refers to the use of a set of Keyword = Value pairs. The keyword component is set to make sense in a specific business application (e.g. Accounting Day, Department, Account No., Account Holder, etc.). The value component can be either unrestricted, or restricted to a set of acceptable values (e.g. A, B or C), or in date format, or restricted by an input mask. Some pairs are defined as mandatory whereas others may be optional.

An application which uses an indexing mask through a collection must supply all Keyword=Value pairs as they are defined using this mask. Any indexing-related errors lead to the record being rejected for conformity. This record is then added to the list of records with errors.

The indexing mask is defined by an administrator via the Arcsys interface or APIs. It is comprised of a set of metadata element definitions.

Journal

A journal is an XML file which contains a list of PREMIS events.

Lot

Arcsys can consolidate several different objects that form a functional set in a client application in the same physical record. It is comprised of different types of objects: files, databases, or any other type of object managed by Arcsys. It is possible to retrieve the entire lot or one of the objects contained in the lot. The MOREQ2010 record is translated in Arcsys implementation by a lot; the lot, as opposed to a MOREQ2010 record, can represent documents that are not yet archived.

Lot enrichment

The process of adding new objects to an existing archive.

Manifest

The manifest is an XML file that defines precisely the content of a record. The manifest contains: metadata associated with the record, structure metadata, a description of the physical files of the record(s) that follow, the object-by-object content of the record, object formats, object names, their size, hash value, the algorithm used to calculate the hash value, etc. The manifest is a type of complete ID card for the record.

The manifest is always written on the storage media and precedes the record that it describes. This process is used to automatically describe storage media (irrespective of the medium). With this system, users can understand media content and metadata without installing the software that generated the records.

Metadata element definition (or keyword)
Component of an indexing mask. We use the term "metadata element definition" rather than the term "keyword" as it is closer to MOREQ2010. The metadata element definition in particular defines the type of metadata (date, string, digital, controlled) and its input mask, for example.
See Also **Indexing mask**.

Object
The object is a basic archived unit that can be retrieved via Arcsys. Lots contain one or more objects. An object can be: a file, a directory, a table, a relational table, etc. The MOREQ2010 component is implemented by this object concept; the object, as opposed to a MOREQ2010 component, can represent a document that has not yet been archived.

Online
Storage level, which must be disk type, that makes records permanently available within an extremely reduced time period.

Permissions
Permissions refer to the user profiles or groups authorized to access documents or sets of documents archived in the system.

Program exit
Place in the standard workflow for picking up and executing specific code.
See Also **Workflow**.

Proof folder
A proof folder consists of a record, a proof slip, and, where appropriate, additional items (common signature or timestamp response, for example) that are used, by demonstrating the fixity and the authenticity of a document, for admission as proof. A proof slip can be generated using Arcsys Web Agent, ArcWeb Module, or Arcsys REST API. A proof folder can only be generated using ArcEP.

Record
A record is an evidential document that is deemed sufficiently important by the creator to be managed by an ERM that will manage its life cycle (retention, disposal, etc.). A record represents an archived lot. A record is archived via a *record request*. Archiving a document *creates a record*.

Relational database (or referential)
Essential component of the system, it contains all the data (excluding archived data) used by Arcsys for its operation. It includes logical entities called "repositories" (see definition).

Repository
Logical entity in the Arcsys relational database. The company can define as many repositories as it wants, either to define a test set, to isolate an application, or for any other reason. These repositories are entirely independent of each other. They all have their own pattern and all have the same structure.

Restore(or retrieval)
This term is used misleadingly in Arcsys to refer to the concept of archive retrieval. It is not accepted in archiving terminology as to mean transfer and then destruction.
See Also **Archive Retrieval**.

Retention and disposal schedule
This comprises all the rules defining the record retention period for a company or an organization, according to risks of unavailability and information system access requirements. It specifies the disposal after these time periods.
See Also **Retention schedule**.

Retention period
A duration expressed in days, months or years of object retention. The retention period is a concept used notably in MOREQ2010.

Retention schedule
A retention schedule defines the start and the end of the retention of records that are attached to it, either directly or through their class.

Retention start date
Date from which a retention period must be taken into account. The retention start date is a concept used notably in MOREQ2010.

Security
An ERMS requirement that involves including documents whose use (confidentiality, risk of exposure) and/or fixity (non modification of content, non-alteration of media) should be closely monitored.

Storage policy
A storage policy is a rule that is referenced by a collection. The policy dictates the storage media which are successively implemented to hold a record, as well as the retention period for each media. The storage policy is defined through the graphical interface. Applications or business users use it indirectly through the reference to a collection. A storage policy can be changed over time to reflect new retention periods or new storage media. The policy covers storage units by logical pool.

Storage pool
Logical storage pool, characterized in particular by its time period (e.g. 10 years). The storage policy assigns a "zone" to a "policy".

Storage zone
The storage zone is a logical entity representing a physical storage space (e.g. set of file systems, tape libraries, cloud storage).

Synchronous retrieval
Archive retrieval that takes place in the form of a direct retrieval of a document (for direct viewing or downloading) in a Web browser.

See Also **Archive Retrieval**.

Time stamping

Time stamping is a technique used to associate a document with a certain date in reference to a given and recognized time system. The date set in this way is an essential element for document authentication. Time stamping can be performed internally or by a third-party time stamp.

Tracking

Result of continuously creating, capturing and maintaining information about the movement and use of the system and objects (ISO 15489-1:2001, 3.19).

Transfer

In an archival sense, this operation sends an archived object to another IT system. Once the transfer is performed, the object can be removed from the ERMS as needed. In OAIS terminology, a transfer represents more specifically the physical transmission of a record or a set of records by a service supplying an archive service. Not to be confused with the transfer of data in the purely technical sense, as with the Arcsys Transfer Server module.

Transit Zone

Entity logically connecting an application agent and a directory, along with additional configuration.

Workflow

A set of operations carried out from the beginning to the end of a process. In Arcsys, this refers to all actions carried out on archives and objects, either directly or indirectly, in the case of archives, from their pre-archiving or preparation to their removal from the system (after they have reached end-of-life). There are other workflows in Arcsys besides the archiving workflow, which are more administration-oriented. Customized workflow involves the use of at least one drop-off point to carry out customer processing.

Registered Trademarks

Firefox is a registered trademark of the Mozilla Foundation.

Linux is a registered trademark of Linus Torvalds.

UNIX is a registered trademark of the Open Group.

Microsoft Windows, Windows NT, Windows Server, SQL Server, Internet Explorer are registered trademarks of Microsoft Corporation in the United States and/or other countries.

SAP is a registered trademark of SAP AG in Germany and other countries.

MySQL is a registered trademark of Oracle and/or its subsidiaries. MariaDB is a registered trademark of Monty Program Ab.

Java is a registered trademark of Oracle and/or its subsidiaries in the United States and other countries.

Infotel is a registered trademark of Infotel SA.

All trademarks mentioned are the property of their respective owners.

Infotel Technical Support

https://techsupport.infotel.com