Active Dashboard 2

Interactive Performance Insight

Installation Guide

2021-03-25

SARTORIUS

Contents

1 Introduction and architecture 4

1.1 Limiting the scope 4
1.2 System requirements 4
2 Installation 4

2.1 Install Node.js 4
2.2 Install MongoDB 5
2.3 Create folder and unzip files 5
2.4 Set up MongoDB to run as a service 5
2.5 Apply Active Dashboard license file 5
2.6 Set up Active Dashboard to run as a service 5
2.7 Set passwords for the Administrator and the user Test 5
2.8 Configure Active Dashboard to connect to data sources 6
2.8.1 SIMCA-online 7
2.8.2 OSIsoft PI 7
3 After installation 9

3.1 Test your installation 9
3.2 Troubleshooting 9
4 Uninstalling Active Dashboard 10

Appendix A Server scripts 11

Appendix B Using HTTPS / TLS with Active Dashboard 12

1 Introduction 12
Requirements 12
2 Outline of the procedure 12
Enabling HTTPS for Active Dashboard server 12
HTTPS for connecting to data source servers 13
4.1.1 Using OpenSSL for certificate management 13
Appendix C Authenticating users with Microsoft Active Directory 15

1 Introduction 15
Further information 15
2 Configuration 15

1 Introduction and architecture

Active Dashboard, part of the Umetrics™ Suite of Data Analytics Solutions, is a client/server solution with a server that connects to multiple data sources to obtain and analyze data, and with client web browsers to view and interact with dashboards. Active Dashboard currently supports SIMCA-online and OSIsoft PI as data sources.

Technically, the server part is a Node.js application that stores its data in a MongoDB database.

Note: By installing Active Dashboard you agree to the End User License Agreement found among the installation files, and on https://webshop.umetrics.com/pages/terms-and-conditions. Active Dashboard uses cookies, like most web sites and applications. For more information, see the **About** page in Active Dashboard, once logged in.

1.1 Limiting the scope

This document describes how to set up an Active Dashboard server environment on a Windows computer, where both the Node.js server and the MongoDB database are installed on the same computer. Installing Active Dashboard on other platforms or hosting the Node.js and MongoDB services on separate computers are supported but out of scope for this document.

This document does not describe how to install the data sources SIMCA-online or OSISoft PI. Refer to the documentation for those products for more information about that.

1.2 System requirements

System requirements are listed in the knowledge base articles:

Active Dashboard 2 https://www.sartorius.com/en/products/process-analytical-technology/data-analytics-software/support/knowledge-base/active-dashboard-2-550706.

Active Dashboard 2.0.3 https://www.sartorius.com/en/products/process-analytical-technology/data-analytics-software/support/knowledge-base/active-dashboard-203-602684.

2 Installation

The installation consists of these major steps, all performed on the server computer (details for each step follows further down):

1. Installation of Node.js.
2. Installation of MongoDB.
3. Creating an Active Dashboard folder and unzipping files into it.
4. Setting up MongoDB to run as a service.
5. Apply Active Dashboard license.
6. Setting up Active Dashboard to run as a service.
7. Testing that you can connect with a web browser and log in.
8. **Optionally:** Enabling HTTPS and/or Windows Active Directory authentication. Both of these topics are described separately in Appendix B and Appendix C.

2.1 Install Node.js

1. Go to https://nodejs.org/ and download version 8.10.0 LTS. This is the most recent version that works with Active Dashboard. You can find it here https://nodejs.org/download/release/v8.10.0/.
2. Click through the installer; leave the defaults at each step (You can change installation directory if you wish).

2.2 Install MongoDB

1. Go to https://www.mongodb.com/ and download version 3.4.4 This is the most recent version that works. You find version 3.4.4 in the archive here https://www.mongodb.com/download-center/community/releases/archive. The zip file is called mongodb-win32-x86_64-3.4.4.zip.

2. Create a folder **C:\MongoDB** and unzip the contents to it. You can create the folder elsewhere if you wish, but you'll need to adjust the path to it in step 2.4 in that case.

2.3 Create folder and unzip files

1. Create the folders **C:\ActiveDashboard** and **C:\ActiveDashboard\MongoDB**.

2. Unzip the **ActiveDashboard.zip** file in **C:\ActiveDashboard**.

3. On a command prompt, type (or paste) the following to create the default collector configuration file:

```
copy ^
  C:\ActiveDashboard\server\config\collectors.config.sample.json ^
  C:\ActiveDashboard\server\config\collectors.config.json
```

4. If you need to make settings, such as change the port number from the default 80, do it in the **server\config\environment\production.js** file using a text editor.

2.4 Set up MongoDB to run as a service

1. Start an Administrative Command Prompt and on it type (or paste):

```
"C:\MongoDB\bin\mongod" ^
  --dbpath=C:\ActiveDashboard\MongoDB --bind_ip localhost ^
  --logpath=C:\ActiveDashboard\MongoDB\log.txt ^
  --install
```

2. Start the MongoDB service in the **Services** tool found in the Windows **Control Panel**.

2.5 Apply Active Dashboard license file

Together with your delivery, you should have gotten a license file. Place it in the following folder:

```
C:\ActiveDashboard\server\config
```

2.6 Set up Active Dashboard to run as a service

Start an **Administrative** Command Prompt and type:

```
cd C:\ActiveDashboard
npm --production run service install
```

This install and automatically starts the Active Dashboard service (which you can verify in the **Services** tool in the **Control Panel** if you like).

2.7 Set passwords for the Administrator and the user Test

When Active Dashboard is started for the first time, it has generated a randomized password for the **Administrator** and **Test** users. Since you do not know these passwords, you need to replace them with new ones:

1. Start a Command Prompt (not an Administrative one) and type:

```
cd C:\ActiveDashboard
npm --production run set-admin-password [password]
```

(replacing **[password]** with a good password of your choice).

2. In the same way, run the following script to set the password for **Test** user:

```
npm --production run set-user-password [password]
```

2.8 Configure Active Dashboard to connect to data sources

Active Dashboard supports fetching data from several different data sources, via configured **data collectors**. The following data collectors are supported:

Data collector	What Active Dashboard fetches	Described in
SIMCA-online	Batches, alarms, CQA variable values	section 2.8.1
OSIsoft PI	Batches, alarms, as configured in the PI Asset Framework	section 2.8.2

By default, the Active Dashboard server is not configured to fetch any data. Here follows a general description of the data collector configuration, with collector-specific details in the subsections below. Follow these steps to make Active Dashboard fetch data:

1. In a text editor, open the file **C:\ActiveDashboard\server\config\collectors.config.json**. If it does not exist, create it or copy the example file **collectors.config.sample.json** and use as starting point.
2. Each data collector is configured with an element ("{ ... }") inside the top-level brackets ("[...]"):

```
[
  {
    "type": "sol or pi",
    "enabled": true,
    "settings": {
      "url": "[server-url]",
      "interval": 60,
      "auth": {
        "username": "[username]",
        "password": "[topsecret]"
      },
      ... more settings
    }
  },
  ... more collectors
]
```

Replace **[server-url]** with the URL to your SIMCA-online or OSIsoft PI server. Also replace **[username]** and **[topsecret]** with the username and password to connect to the server. Each specific collector also has its own configuration parameters, described below.

Configure as many collectors as needed, for example one for each SIMCA-online server you want to access.

3. Restart the Active Dashboard server, either via the **Services** tool in the **Control Panel** or by typing the following in an **Administrative** Command Prompt:

```
cd C:\ActiveDashboard
npm --production run service restart
```

Note: it is not advised to use "localhost" as host name in the server URLs, as Active Dashboard uses the URLs to identify data sources. Instead use the real computer name of the server such as "MyServer".

Note: mixing **SIMCA-online** and **OSIsoft** PI data collectors in the same Active Dashboard installation is not supported.

2.8.1 SIMCA-online

The following notes are relevant for SIMCA-online data collectors:

- We recommend that you create a special user account in SIMCA-online for use from Active Dashboard. This account needs to have access rights to the folders containing project configurations you want to see in Active Dashboard, but need not hold the Administrator role in SIMCA-online.
- If you plan to use the **Open in SIMCA-online** feature in the dashboards and the SIMCA-online server runs at another RPC port than the default 2731, you need to specify this port in the configuration file. This setting is configured with the parameter "serverRPCPort"; set it to SIMCA-online server's port number, which can be seen in SIMCA-online **Server Options** on the server computers. For more information on the **Open in SIMCA-online** feature, see the User Guide.

2.8.2 OSIsoft PI

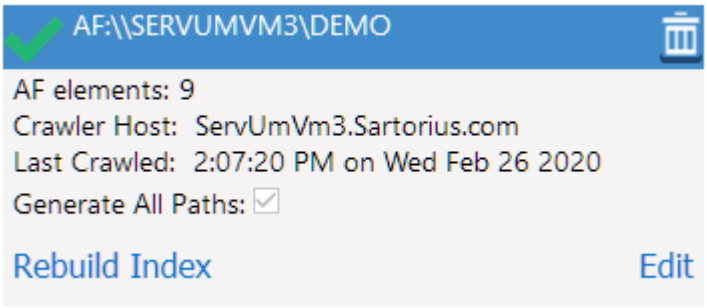
The following notes are relevant for OSIsoft PI data collectors:

- Active Dashboard uses the **PI Asset Framework** (AF) to define what **assets** (**units** in SIMCA-online) to fetch data for. AF elements represents assets, with element attributes defining static information (asset name, location, etc.) as well as batch data (batches, alarms, etc.).
- In the collector configuration, the parameter "elementQueryString" is used to select which elements Active Dashboard should fetch. The syntax is described in the following page (use "*" to match all elements): https://techsupport.osisoft.com/Documentation/PI-Web-API/help/controllers/search/actions/query.html
- The configuration parameter "startDate" can be set to determine how far back Active Dashboard will fetch data.
- For Active Dashboard to be able to discover the elements in the PI database you need to first configure the PI Web API Indexed Search Crawler so that Active Dashboard can query the database. Learn more on the Crawler in https://techsupport.osisoft.com/Documentation/PI-Web-API/help/topics/getting-started.html

  You can do this by following the steps below:
  1. Open up a web browser on the PI server computer that hosts the web API.
  2. Navigate to the address <PI server>/piwebapi. Log in as a user with administrative permissions in P).
  3. Now you should see a page with a few links:

  4. Click the link for 'Search' (2nd last link in the screenshot).
  5. Then click 'Admin'

6. Now you should see a **Search Service Administration** dashboard
    a. In the top right, click 'Databases'
    b. Now click 'Add Database' and enter the address to the database to be indexed of the form af:\\<PI server>\<database name> For example: af:\\servumvm3\demo
    c. Click 'Save and close', now the database will be indexed, this may take a while depending on the size of the database. When it is finished it will get a green checkmark in the top left (see image below)



- Active Dashboard expects to find a set of PI attributes on the elements it will query. These attributes need to be configured correctly in order for Active Dashboard to be able to run.

  The expected attributes are as follows. Note that names are case sensitive.

Attribute name (case sensitive)	Data type	Required	Description	Examples
Type	String	Yes	Machine type	"Fermentation tank"
Name	String	Yes	Machine name or id	"Fermenter 01"
Region name	String	No	Physical region that encompasses one or more locations	"Europe"
Location name	String	No	Physical location	"Stockholm"
Location coordinates **OR** LocationCoordinates	Single array	No	GPS coordinates in decimal form, coordinates to the west and south are described with negative values. [latitude, longitude]	[59.332, 18.064] (Stockholm) [-22.89, -43.17] (Rio de Janeiro)
LocationLatitude **AND** LocationLongitude	Single	No	GPS coordinates in decimal form, coordinates to the west and south are described with negative values.	Lat: 59.332 and Long: 18.064 (Stockholm) Lat: -22.89 and Long: -43.17 (Rio de Janeiro)
Product type	String	No	A type name that encompasses one or more product names	"Gel capsule"
Product name	String	No	Specific product name	"Drug A"

Notes:

- Attributes listed as not required in the table above are still recommended. Without these attributes some measures in Active Dashboard will not work.
- It is important that required attributes exists in PI. Otherwise Active Dashboard will not be able to get any data and will show errors in the log such as:
  - Required attributes 'Name' (<name or 'missing'>), 'Type' (<type or 'missing'>), for uniqueId: <element uniqueId>
  - Location coordinates must be decimal degrees ex. [63.8, 20.1], for Name: <element Name attribute if available> uniqueId: <element uniqueId>
- The location coordinates have different forms to give some flexibility. Only one of the attributes will be used if many variants are provided by you. The priority order is: Location coordinates (highest priority) > LocationCoordinates > LocationLatitude and LocationLongitude.

3 After installation

As soon as the Active Dashboard server is started, it will connect to the data sources and begin fetching its data. This can take a while so allow time after starting the service before expecting all data to be available.

3.1 Test your installation

Start a web browser on your server computer and type the address http://**server**/ (replace **server** with the name of your server computer). It should connect and display the Active Dashboard login page.

Now, log in with one of the local users configured during the installation (**Administrator** or **Test**). To be able to set user permissions, you need to be logged in as Administrator; regular users have permissions set via their user groups (list, view, interact with, and create, edit, move dashboards, respectively). If Active Directory authentication is configured (see appendix c), you can also log in with user accounts from the Windows Active Directory and administrators can configure user permissions via Active Directory user groups (see the User Guide).

3.2 Troubleshooting

To monitor what the server does, consult the log files in the **C:\ActiveDashboard\daemon** folder.

Active Dashboard will by default use TCP port 80. If that port is in use the service will not be able to accept any web browsers connecting to it. This will result in errors such as the following in the **activedashboard.err.log** file:

```
Unhandeled rejection: { [Error: listen EACCES 0.0.0.0:80]
  code: 'EACCES',
  errno: 'EACCES',
  syscall: 'listen',
  address: '0.0.0.0',
  port: 80 }
```

To fix this, stop any other service on the computer that is using port 80 or configure Active Dashboard to use another port.

It is possible to turn on debug logging in the log file by going to server/config and then copying and renaming the file local.env.sample.js to local.env.js.
Edit the renamed file and change the line DEBUG='' to DEBUG: 'activedashboard:*' Save the file, and restart the Active Dashboard service (if it was running when the file was changed). After this, the log file in the daemon folder (*.err.log) will contain detailed information.

Note: It is possible to set it to DEBUG='*' to output even more information, but this is mostly from third party libraries which most of the time will not be useful and is therefore normally not recommended)

With the debug logging turned on some information about what has gone wrong is sometimes provided with the error. For example uniqueId, this id can be used in the PI web API query to find the offending item.

4 Uninstalling Active Dashboard

If you want to remove Active Dashboard from a server, follow these steps:

1. Stop and uninstall the MongoDB and Active Dashboard services by starting an Administrative Command Prompt and type:

```
cd C:\ActiveDashboard
npm --production run service uninstall
"C:\MongoDB\bin\mongod" --remove
```

2. Delete the **C:\ActiveDashboard** folder.
3. Uninstall Node.js in **Programs and Features** in the Windows **Control Panel**.
4. Remove the C:\MongoDB folder

Appendix A Server scripts

Server scripts are run on a command prompt on the Active Dashboard server computer. Available scripts are:

Script	Purpose
seed	Empty the Active Dashboard database, and re-create an empty database. Useful if you want to start from scratch or have deleted configurations from SIMCA-online server, or an entire SIMCA-online server that you do not want to appear in Active Dashboard. Note that the password for the users Administrator and Test becomes randomized and have to be set using the scripts below. **Note:** Stop the Active Dashboard service before running this script.
seed dashboards	Removes all existing dashboards and re-creates the default dashboards. **Note:** Stop the Active Dashboard service before running this script.
seed dataset	Removes all data fetched from data sources; the data will be fetched again via the configured data collectors. **Note:** Stop the Active Dashboard service before running this script.
reset-aggregation **[level]**	Reset the aggregation for the select level and the levels above it that represent larger timespans, so that the data will be re-aggregated. if **[level]** is empty all data will be reset. Possible levels 'years', 'months', 'days', 'hours', 'raw'. Example: reset-aggregation days, this will re-aggregate the data that is aggregated to days, months and years. **Note:** Stop the Active Dashboard service before running this script.
set-password admin **[password]** set-password user **[password]**	Sets the password for the Administrator / Test user to **[password]**. (Omit password to generate a new one, printed to the console.)
service install service uninstall	Install / uninstall Active Dashboard as a Windows service.
service start service restart service stop	Start / restart / stop the already installed Active Dashboard service.

To run one of these scripts, start a Windows Command Prompt on the server computer and type:

```
cd C:\ActiveDashboard
npm --production run [script]
Replace [script] with one of the scripts in the table above.
```

Appendix B Using HTTPS / TLS with Active Dashboard

1 Introduction

Active Dashboard can be configured to use encrypted connections and thus increase security in two ways:

1. HTTPS enabled for Active Dashboard, providing secure connections to connected web browsers; and
2. between Active Dashboard and the data sources (SIMCA-online and / or OSIsoft PI servers) it is connected to.

These items are independent; one can be enabled without the other.

Requirements

The following things are required for used HTTPS with Active Dashboard:

- A **TLS (SSL) certificate issued for the common name** of the Active Dashboard server computer. The certificate must be a PEM file (with the extension ".pem").
- The common name is the name you want to use when connecting to the server, such as http://dashboard.example.com.
- **Tip:** Creating a certificate for the common name **localhost** is meaningless: it only allows connecting to that server from the same computer.
- **A private key for your certificate**, in the form of another PEM file.
- An optional passphrase needed to decrypt the private key. Only required if your private key was encrypted when it was created.

If you do not know what certificates are, you need to learn more about this to be able to use HTTPS with Active Dashboard. Your IT or web department will be able to assist.

You can also see below for information how to use OpenSSL to create a certificate-signing request or how to create a self-signed certificate for **testing purposes**. To use OpenSSL, you need at least some experience with command line tools.

2 Outline of the procedure

When you fulfil the above requirements and have a certificate, you need to complete the following steps:

1. Edit the **server/config/environment/production.js** configuration file, to specify your certificate and private key files, and to enable HTTPS. Note that if you enable HTTPS for Active Dashboard, the server will not allow unencrypted HTTP at the same time.
2. Optionally change the port number of the Active Dashboard to 443 if you want to use the standard HTTPS port number. Otherwise the server will use the same port number as without HTTPS.
3. Restart the Active Dashboard service.
4. In a web browser, connect to the server with HTTPS: https://servername.domain.com.

As always, if you have any problems, consult the log files.

Enabling HTTPS for Active Dashboard server

Copy the certificate and private key files (**cert.pem** and **key.pem** in the example below) to the **server/config** folder of the Active Dashboard server.

Edit the **server/config/environment/production.js** file with a text editor. In it, locate or create the "useHttpsTls" settings:

```
// HTTPS / TLS between backend and frontend
useHttpsTls: {
  enabled: true,
  cert: 'cert.pem',
  key: 'key.pem',
  // passphrase: 'required if the key.pem file has a passphrase'
},
```

Change “enabled” to “true”, and specify the names of the certificate file and the key file respectively. If your key file uses encryption to protect its contents, you need to provide the passphrase too.

HTTPS for connecting to data source servers
To configure Active Dashboard to connect to data source servers (such as SIMCA-online or OSIsoft PI) using HTTPS, you need to edit the file **server/config/collectors.config.json**. There you change the URL from “http” to “https” to tell the Active Dashboard to use HTTPS.

Note: These changes can be done independently, and does not require a certificate.

For example, the following would tell Active Dashboard to connect to the SIMCA-online server with HTTPS:

```
[
  {
    "type": "sol",
    "enabled": true,
    "settings": {
      "url": "https://dashboard.example.com",
      "auth": {
        "username": "Administrator",
        "password": ""
      }
    }
  }
]
```

If you're using a self-signed certificate for https you might have to turn off strict ssl in order for it to work, this is not a problem with a proper certificate and should only be turned off for testing purposes. The setting is found in the server/config/collectors.config.js file, type "strictSsl": false in the settings section for a collector.

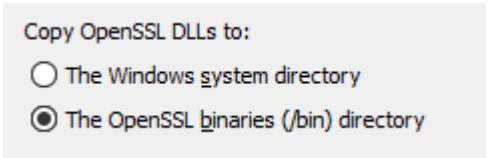
4.1.1 Using OpenSSL for certificate management

The OpenSSL (https://openssl.org/) toolkit can be used to create certificate-signing requests and self-signed certificates for testing. Here is some short guidance for this. For more information, read the full documentation on the OpenSSL website.

Installation

1. Download and install OpenSSL 1.0.2k or later from https://slproweb.com/products/Win32OpenSSL.html

Install to **C:\OpenSSL-Win32**, and select to copy OpenSSL DLLs to the OpenSSL binaries directory:



2. Open a Command Prompt and type the commands:

```
set OPENSSL_CONF=C:\OpenSSL-Win32\bin\openssl.cfg
cd c:\OpenSSL-Win32\bin
```

3. Leave the command prompt open; you will use it for all OpenSSL commands.

Create a Certificate Signing Request - for production

In the command prompt you left open (see above), execute the following:

```
openssl.exe req -new -newkey rsa:2048 -nodes -out server.csr -keyout server.key
```

You will then be prompted to provide details about your certificate. This includes the location of the server, and most importantly the common name of your server. The common name should match what your web browsers will use to connect to the server, such as dashboard.example.com.

This creates a key file that is secret and a CSR-file which you send to a Certificate Authority. The CA then will sign the CSR and provide you with a certificate file in PEM format.

Create a self-signed certificate with OpenSSL – for testing purposes

In the command prompt you left open (see above), execute the following:

```
openssl req -x509 -newkey rsa:2048 -keyout key.pem -out cert.pem -days 365 -nodes
```

You will then be prompted to provide details about your certificate. This includes the location of the server, and most importantly the common name of your server. The common name should match what your web browsers will use to connect to the server, such as dashboard.example.com.

This creates two files, **cert.pem** and **key.pem**, which you then can use in Active Dashboard.

A note about passphrases: If you remove the "-nodes" flag, you will be prompted for a passphrase for your key. This passphrase must then be put in the "useHttpsTls" settings object (see above).

Appendix C Authenticating users with Microsoft Active Directory

1 Introduction

Apart from the local user account(s) set up during installation, Active Dashboard also supports accounts from an LDAP directory service such as a Microsoft Active Directory server commonly used in Windows domains. This is the recommended way of authenticating users, because the user then can log in as "DOMAIN\user" and then type her regular windows password. A user that has logged in this way is a non-administrative user by default. To perform administrative tasks, log on as the built-in **Administrator** account or configure an Active Directory group to have the administrator role, see **Configuration** of this section.

In this section, the Active Dashboard server is acting as an LDAP client when connecting to an LDAP server and "(authenticating) user" represents the user trying to log in to use the Active Dashboard application.

Active Dashboard authentication via LDAP, if enabled in the configuration, is done with the following steps:

1. The Active Dashboard server connects (**binds** in LDAP terminology) to the LDAP server (optional, depending on the LDAP server configuration);
2. the LDAP directory is searched for the entry for the authenticating user and the correct user identifier (**distinguished name**) is extracted; and
3. the Active Dashboard server connects (binds) again to the LDAP server with the user identifier (**distinguished name**) and the user-given password.

If all three steps pass, the user is authenticated and logged in.

Further information

This LDAP Guide is a good general introduction to what LDAP is and how it works.

The SIMCA-online knowledge base article regarding Active Directory authentication can also be of help: umetrics.com/kb/active-directory-authentication-users-simca-online.

2 Configuration

LDAP authentication is **not** enabled by default in Active Dashboard, so to use it several configuration parameters must be set:

1. In a text editor, open the file **C:\ActiveDashboard\server\config\environment\production.js**. It contains a section similar to this:

```
ldapAuthentication: {
  enabled: false,
  // url: 'ldap://ldap-server:389',   // URL to LDAP / AD server
  // url: 'ldaps://ldap-server:389',  // URL to LDAP / AD server (TLS)
  // url: 'ldaps://ldap-server:636',  // URL to LDAP / AD server (SSL)

  // useCredentialsFromAuthenticatingUserOnBind: true,   // use the authenticating
user's credentials for binding (default)
  // useCredentialsFromAuthenticatingUserOnBind: false,  // use the credentials
below for binding

  /// Default AD domain to use when binding with the authenticating user's
credentials
```

```
  /// and no domain is specified. (Can be left undefined to not prepending any
  /// domain to the username.)
  // defaultAdDomain: '...',

  /// Bind credentials if the authenticating user's credentials are not used for
  /// binding, as well as for group fetching.
  // bindDn: '...',
  // bindCredentials: '...',

  /// The distinguished name (DN) of the base subtree entry to search within, often
  /// on the form 'dc=company,dc=com'.
  // searchBase: 'dc=example,dc=com',

  /// Default, suitable for Active Directory; can be overridden. All occurences
  /// of "{{username}}" is replaced with the authenticating user's username.
  // searchFilter:
'(|(userPrincipalName={{username}})(sAMAccountName={{username}})(mail={{username}}))
',

  /// TLS / SSL options, if using LDAPS
  tlsOptions: {
    // rejectUnauthorized: true,   // require the LDAP server to offer a certificate
signed by a trusted CA (default)
    // rejectUnauthorized: false,  // accept any certificate the LDAP server offers

    // more options described here:
    // https://nodejs.org/api/tls.html#tls_tls_connect_options_callback
  },
},
```

The following parameters must be set: “enabled”, “url”, “searchBase”, “bindDn”, and “bindCredentials”. The rest can possibly be left as default, depending on your LDAP server configuration.

2. Set or change the following parameters in the configuration file (lines beginning with "//" are just examples; for them to take effect, remove the double slashes):
    - Change “enabled” to “true”.
    - Set “url” to the correct URL for your network. Your IT department should know what the address is. In a Windows domain you can run “nslookup DOMAIN” on a command prompt to see your nearest domain controller which can be used as an LDAP server. Using “ldaps” instead of “ldap” will make the communication between Active Dashboard and the LDAP server encrypted with SSL / TLS.
    - The “searchBase” parameter must be set to the **distinguished name** of the root directory entry to search within. In a Windows domain with an Active Directory , this is often based on the company domain name, for example “dc=DOMAIN,dc=com”. Your IT department can tell you the correct value. You can also run “whoami /fqdn” on a command prompt to see this information for the logged on user.
    - Active Dashboard needs a service account to login with when fetching Active Directory groups; set “bindDn” and “bindCredentials” to the **user principal name** and password, respectively for this account.
3. The default value for “searchFilter” is adequate for most LDAP configurations; common variants of Windows usernames will be accepted by Active Directory (AD) servers, as well as user e-mail addresses and the other common way of storing usernames (“uid”). If you have different requirements for what identifiers to accept

or have a custom LDAP configuration, the value can be changed to any valid LDAP search query. Occurrences of "{{username}}" in the search filter will be replaced by the username entered by the authenticating user. For example, to accept both usernames and e-mail addresses in the common LDAP attributes "uid" and "mail", the search filter would be "(|(uid={{username}})(mail={{username}}))".

4. The rest of the parameters determines how Active Dashboard binds with the LDAP server, as described in step 1 in the introduction. By default, the authenticating user's credentials are also used for the initial bind. The parameter "defaultAdDomain" can be set to the name of a Windows domain (for example "DOMAIN") if using Active Directory; then users can enter their username as "user" instead of "DOMAIN\user".

5. If regular users don't have permission to search the directory, the LDAP server accepts anonymous searching (no binding required before searching), or you prefer to use a specific system account, then set "useCredentialsFromAuthenticatingUserOnBind" to "false". Set "bindDn" and "bindCredentials" to the user identifier and password of the system account to bind with, or leave them undefined for anonymous directory searching. Active Directory accepts identifiers in the forms **distinguished name**, **user principal name** (**user**@example.com), and "example\username". The recommendation is to first try authenticating with "useCredentialsFromAuthenticatingUserOnBind" enabled (default, "true"), and only if it does not work, set the flag to "false".

6. If using encrypted LDAP (an "ldaps://" URL) there are relevant options that can be set in the configuration group "tlsOptions". Especially, "rejectUnauthorized" must be set to "false" if the LDAP server has a self-signed certificate; by default, Active Dashboard rejects the LDAP server if its certificate is not signed by a trusted certificate authority (CA).

Note: Some general information about TLS / SSL can be found in the section **Using HTTPS / TLS with Active Dashboard**.

Active directory groups can be given admin priviliges by adding them to activeDirectoryAdminGroups property of the config.

```
activeDirectoryAdminGroups: ['DN_OF_ADMIN_GROUP_1', 'DN_OF_ADMIN_GROUP_2'],
```

Special characters like **#**, **>**, **<**, ****, need to be escaped with double backslash see https://social.technet.microsoft.com/wiki/contents/articles/5312.active-directory-characters-to-escape.aspx for the complete list.