

MDCMSCmdService Installation Instructions for Linux

These instructions provide the customer with the necessary information to install and configure the MDCMS Command Service Application as a Linux service to automatically process Pre- and Post-Deployment scripts delivered by MDCMS.

MDCMSCmdService Installation Instructions for Linux	1
1. Version History	1
2. General Information	1
3. Installation and Configuration of MDCMSCmdService for Linux	1
..3.1 Prerequisites.....	1
..3.2 Configure the Service using systemd on Debian Linux 11 x64 / 1 GB Memory / 25 GB Disk / Debian 11 x64	3
..3.2.1 Install java.....	3
..3.2.2 Configure the service – systemd / systemctl	4
..3.2.3 Configure logging for the MDCMSCmdService	9
..3.2.4 Starting the service for the first time	10
..3.3 Configure the Service with upstart on Ubuntu Linux 14.04 x64, 16.04 x64 (deprecated and no longer supported).....	11
..3.3.1 Install java.....	11
..3.3.2 Configure the service.....	12
4. Testing and Using the Service	20
..4.1 Testing locally	20
..4.2 Test example.....	20
5. Usage in RFP	23

1. Version History

1.2.2022	Former version of document	René Unternährer
15.2.2022	Updated for permission addition	René Unternährer

2. General Information

The MDCMSCmdService Application runs as a service on any Linux server. The service processes script files in a designated folder. The script files are placed in the folder via FTP during the MDCMS RFP deployment process of Linux components. The contents of the script files must be a set of syntactically correct Linux commands that will be executed by MDCMSCmdService.

If all of the commands in the script execute without exception, the file is moved to the ok folder for positive confirmation to MDCMS. If a command in the script fails, the file is moved to the nok folder for negative confirmation to MDCMS.

3. Installation and Configuration of MDCMSCmdService for Linux

..3.1 Prerequisites

- Java JDK 6 or newer (with Java EE)
- Linux Operating System

Before you install the Linux service, you need to install a JAVA JDK Version 5 or newer (Recommended).



Note that with different Linux systems there are different techniques to install the service. In further sections of this document you will find (maybe) a explanation for your Linux system.

Create/Designate a folder as the destination for the contents of MdCmdServiceLinux.zip. It is recommended to have full access rights to that folder.

Unzip the contents into the folder.

The MdCmdServiceLinux.zip file consists of several files:

Filename	Description
mdcmd.jar	The runtime java application for executing the script commands
mdcmcmd.service	Service file to create the service in the etc/system/system folder
mdcmcmd.service.permission	Service file to create the service in the etc/system/system folder This file has one more parameter with the permission parameter 776 -> this is the permission to be set for all created folders and files during the service run in order for the service to be able to read, write and delete the files for the process. This file needs to be renamed to mdcmcmd.service though before use
mdcmcmd.conf	Service file to create the service in the etc/init (upstart folder) -> deprecated
mdcmcmd.conf.permission	Service file to create the service in the etc/init (upstart folder) -> deprecated This file has one more parameter with the permission parameter 776 -> this is the permission to be set for all created folders and files during the service run in order for the service to be able to read, write and delete the files for the process. This file needs to be renamed to mdcmcmd.conf though before use
testscripts-linux	Folder with testscripts for linux
➔ ls.txt -> ls1-4.txt	Directory listing example scripts that executes successfully
➔ ls-wrong.txt -> ls-wrong1-4.txt	Example script that does not execute successfully
MDCMSCmdService Installation Instructions Linux.docx	This document with the installation instructions - systemd
MDCMSCmdService Installation Instructions Linux-old.docx	This document with the installation instructions Upstart (deprecated)



..3.2 Configure the Service using systemd on Debian Linux 11 x64 / 1 GB Memory / 25 GB Disk / Debian 11 x64

.3.2.1 Install java

This instructions show you how to install the service using the mdcmd.jar on Debian Linux 11 x64 using systemdctl. Systemd is the ancestor of upstart previously used but deprecated for long.

First you need to install java.

First logon to your system using PUTTY for example (<https://www.putty.org/>)

```
161.35.66.216 - PuTTY
login as: root
root@161.35.66.216's password:
Linux debian-s-lvcpu-lgb-fral-01 5.10.0-7-amd64 #1 SMP Debian 5.10.40-1 (2021-05-28) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Jan 31 08:00:08 2022 from 178.192.31.107
root@debian-s-lvcpu-lgb-fral-01:~# sudo apt update
Hit:1 http://security.debian.org/debian-security bullseye-security InRelease
Hit:2 http://deb.debian.org/debian bullseye InRelease
Get:3 http://deb.debian.org/debian bullseye-updates InRelease [39.4 kB]
Get:4 http://deb.debian.org/debian bullseye-backports InRelease [44.2 kB]
Hit:5 https://repos-droplet.digitalocean.com/apt/droplet-agent main InRelease
Fetched 83.6 kB in 1s (138 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
85 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@debian-s-lvcpu-lgb-fral-01:~#
```

After signing in
it is suggested to do the update first:

➤ `sudo apt update`

To check if java is installed you can
Type command `> java- version`

Usually java is installed in:
`/usr/lib/jvm/`

To install java you can follow these steps or refer to other recommendations.
<https://www.digitalocean.com/community/tutorials/how-to-install-java-with-apt-on-debian-10>



Now we can install java. Easiest for that is to install the default-jre and default-jdk

Java default-jre

- `sudo apt install default-jre`

After installation execute command again:

- `Java -version`

```
root@debian-s-lvcpu-lgb-fral-01:~# java -version
openjdk version "11.0.14" 2022-01-18
OpenJDK Runtime Environment (build 11.0.14+9-post-Debian-1deb11u1)
OpenJDK 64-Bit Server VM (build 11.0.14+9-post-Debian-1deb11u1, mixed mode, sharing)
root@debian-s-lvcpu-lgb-fral-01:~#
```

Java default-jdk

- `sudo apt-get install default-jdk`

After installation execute command again:

- `Javac -version`

```
root@debian-s-lvcpu-lgb-fral-01:~# javac -version
javac 11.0.14
root@debian-s-lvcpu-lgb-fral-01:~#
```

Now we also see the installations in the directory mentioned before:

```
Server: /usr/lib/jvm
├── ? gnupg2
├── ? groff
├── ? grub
├── ? ifupdown
├── ? init
├── ? jvm
│   ├── ? default-java
│   ├── ? java-1.11.0-openjdk-amd64
│   ├── ? java-11-openjdk-amd64
│   ├── ? openjdk-11
│   └── ? kernel
└── ? kernel
```

.3.2.2 Configure the service – systemd / systemctl

Next is we have to create a folder where we want our service to run in.

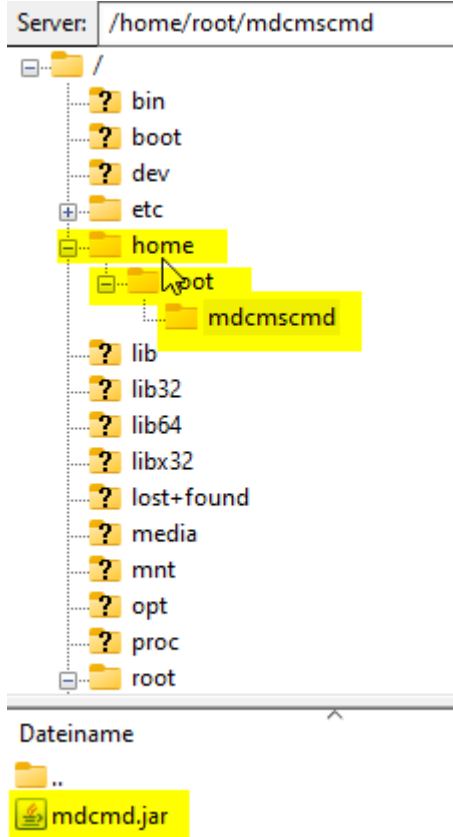
For that we usually use the home directory as starting point. For our example we create the following directory structure:

`/home/root/mdcmscmd`

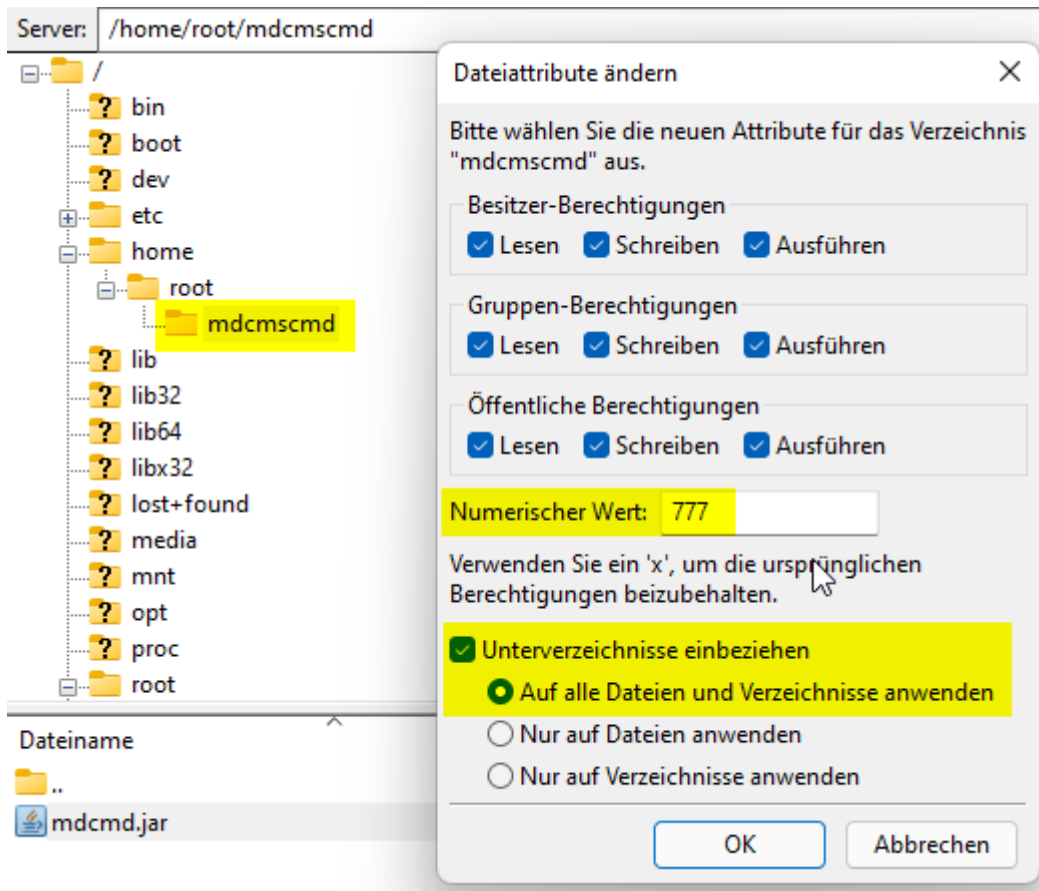
To create the directories under home we either use putty or we also could use an SFTP client like fillezilla to create these directories (<https://filezilla-project.org/download.php?type=client>)

An important thing is that the folder mdcmscmd has full write and execute rights.

We copy the mdcmd.jar into the mdcmscmd folder.



Now we make sure that the mdcmscmd folder has full right (777).



To change it from command line e.g. putty:

- `Sudo chmod -R 777 /home/root/mdcmscmd`



It's time now to create the systemd service.

The jar file is located in **home/root/mdcmcmd** directory.

We created a `mdcmcmd.service` and `mdcmcmd.service.permission` file already with the following content:

`mdcmcmd.service`

`[Unit]`

`Description=MDCMS Command Service`

`After=network.target`

`StartLimitIntervalSec=30`

`StartLimitBurst=2`

`[Service]`

`SuccessExitStatus=143`

`User=root`

`Type=simple`

`WorkingDirectory=/home/root/mdcmcmd`

`ExecStart=java -jar mdcmd.jar start /home/root/mdcmcmd/drops`

`Restart=always`

`[Install]`

`WantedBy=multi-user.target`

`mdcmcmd.service.permission`

`[Unit]`

`Description=MDCMS Command Service`

`After=network.target`

`StartLimitIntervalSec=30`

`StartLimitBurst=2`

`[Service]`

`SuccessExitStatus=143`

`User=root`

`Type=simple`

`WorkingDirectory=/home/root/mdcmcmd`

`ExecStart=java -jar mdcmd.jar start /home/root/mdcmcmd/drops 777`

`Restart=always`

`[Install]`

`WantedBy=multi-user.target`

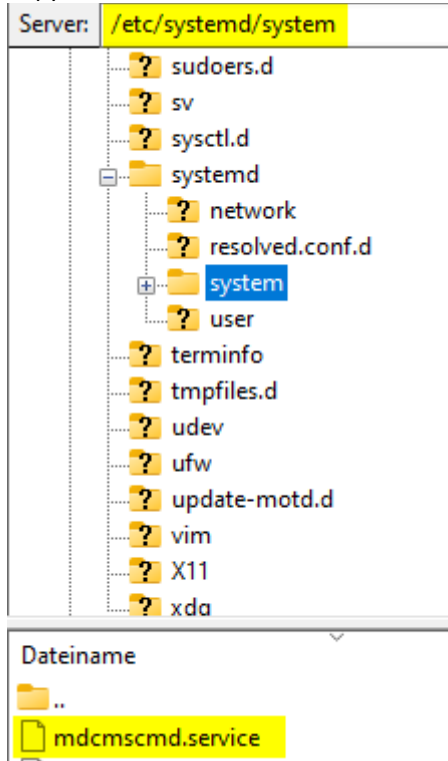
In the `*.permission` file there is an additional parameter **777**. This is the permission to set on all the files created in the `ok`, `nok`, `okResult` and `nokResult` folder in order for the FTP to have access to these files. This is usually not needed.

If needed then use this file but rename it to `mdcmcmd.service`.



Feel free to customize this file to your own needs. This is just a basic file.

Copy the **mdcmcmd.service** file into the **/etc/systemd/system** folder:



Before we can start the service we need to reload systemd in order for it to know about the new service.

- Sudo `systemctl daemon-reload`

```
root@debian-s-lvcpu-lgb-fral-01:~# sudo systemctl daemon-reload
root@debian-s-lvcpu-lgb-fral-01:~#
```

Once reloaded, we can start the service. Please be aware that the daemon-reload needs to occur after every change of the `*.service` file to have the change loaded.

- Sudo `systemctl start mdcmcmd.service`

Anytime we can find out about the status of the service

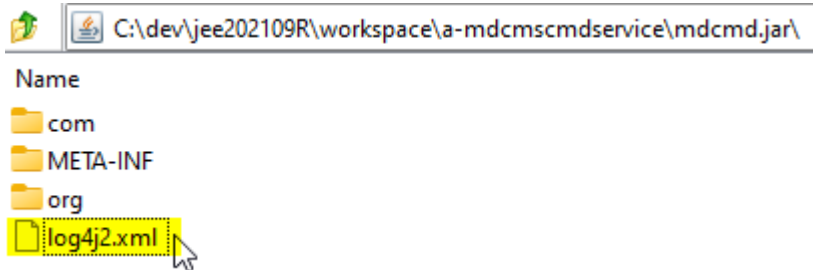
- Sudo `systemctl status mdcmcmd.service`

```
root@debian-s-lvcpu-lgb-fral-01:~# sudo systemctl status mdcmcmd.service
● mdcmcmd.service - MDCMD Command Service
   Loaded: loaded (/etc/systemd/system/mdcmcmd.service; disabled; vendor preset: enabled)
   Active: activating (start) since Mon 2022-01-31 14:52:44 UTC; 18s ago
     Ctrl PID: 22480 (java)
       Tasks: 15 (limit: 1132)
      Memory: 26.4M
         CPU: 769ms
    CGroup: /system.slice/mdcmcmd.service
            └─22480 java -jar mdcmd.jar start /home/root/mdcmcmd/drops && /var/log/mdcmcmd.log

Jan 31 14:52:44 debian-s-lvcpu-lgb-fral-01 systemd[1]: Starting MDCMD Command Service...
root@debian-s-lvcpu-lgb-fral-01:~#
```


.3.2.3 Configure logging for the MDCMSCmdService

The MDCMSCmdService (mdcmd.jar) uses log4j2 logging. Therefore a log4j2.xml file is configured in the root of the mdcmd.jar.



The file can be viewed by unpacking the jar file. The file can also be edited and saved back within the jar to customize the logfile location, minLevel, maxLevel, root level.

To change the logfile location change the APP_LOG_ROOT value. Can be absolut or relative to the location of the mdcmd.jar

Relativ:

```
<Property name="APP_LOG_ROOT">logs</Property>
<Property name="APP_LOG_ROOT_HIST">logshist</Property>
```

Location of mdcmd.jar is /home/root/mdcmcmd/mdcmd.jar

Logfiles typically created in /home/root/mdcmcmd/logs/...

Loghistory files typically created in /home/root/mdcmcmd/logshist/...

Absolut:

```
<Property name="APP_LOG_ROOT">/home/root/mdcmcmd/logs</Property>
<Property name="APP_LOG_ROOT_HIST">/home/root/mdcmcmd/logshist</Property>
```

Location of mdcmd.jar is /home/root/mdcmcmd/mdcmd.jar

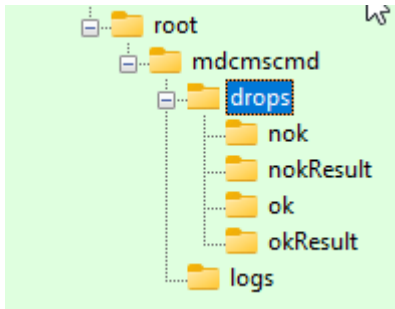
Logfiles typically created in /home/root/mdcmcmd/logs/...

Loghistory files typically created in /home/root/mdcmcmd/logshist/...

.3.2.4 Starting the service for the first time

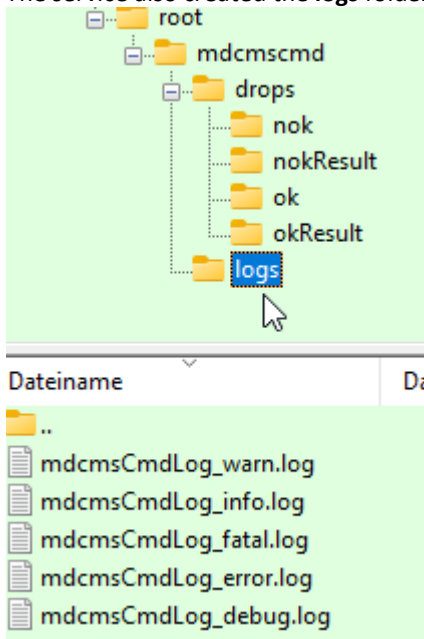
After we started the job the first time. We have a look at the directories and where they were created:

Server: /home/root/mdcmscmd/drops



At the beginning the service created the drops folder with the nok, nokResult, ok and okResult folders. In the drops folder the scripts are copied that have to be executed. We simulate that later.

The service also created the **logs** folder:



In the logs we can track what the service was doing so far.



..3.3 Configure the Service with upstart on Ubuntu Linux 14.04 x64, 16.04 x64 (deprecated and no longer supported)

.3.3.1 Install java

This instructions show you how to install the service using the mdcmd.jar on Ubuntu Linux 14.04 x64 using upstart. Usually upstart is already installed on Ubuntu. So no need to install it.

First you need to install java.

First logon to your system using PUTTY for example
(<http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>)

```
root@Ubuntu: ~
login as: root
root@188.166.120.54's password:
You are required to change your password immediately (root enforced)
Welcome to Ubuntu 14.04.2 LTS (GNU/Linux 3.13.0-52-generic x86_64)

* Documentation:  https://help.ubuntu.com/

System information as of Mon Jul 13 15:58:09 EDT 2015

System load: 0.0          Memory usage: 5%    Processes:      53
Usage of /:  5.0% of 29.40GB  Swap usage:  0%    Users logged in: 0

Graph this data and manage this system at:
  https://landscape.canonical.com/

Changing password for root.
(current) UNIX password:
Enter new UNIX password:
Retype new UNIX password:
root@Ubuntu:~# java
The program 'java' can be found in the following packages:
 * default-jre
 * gcj-4.8-jre-headless
 * openjdk-7-jre-headless
 * gcj-4.6-jre-headless
 * openjdk-6-jre-headless
Try: apt-get install <selected package>
root@Ubuntu:~#
```

After signing in you can verify if java might be already installed or has to be installed or configured.

Type command > java- version

Might be that the following list as above appears and you can try installing it. If you would like another version of java you can use the following commands (you can refer to the following page:

<https://www.digitalocean.com/community/tutorials/how-to-install-java-on-ubuntu-with-apt-get>)

Before installing java it's recommended to update the configuration anyways with:

- Sudo apt-get update



Now we can install java. Easiest for that is to install the default-jre and default-jdk

Java default-jre

- `sudo apt-get install default-jre`

Java default-jdk

- `sudo apt-get install default-jdk`

After installation execute command again:

- `Java -version`

Ubuntu 14.04.01

```
root@ubuntu-64-14:~# java -version
java version "1.7.0_151"
OpenJDK Runtime Environment (IcedTea 2.6.11) (7u151-2.6.11-2ubuntu0.14.04.1)
OpenJDK 64-Bit Server VM (build 24.151-b01, mixed mode)
root@ubuntu-64-14:~#
```

Ubuntu 16.04.02-b12

```
root@ubuntu-64-16:~# java -version
openjdk version "1.8.0_151"
OpenJDK Runtime Environment (build 1.8.0_151-8u151-b12-0ubuntu0.16.04.2-b12)
OpenJDK 64-Bit Server VM (build 25.151-b12, mixed mode)
root@ubuntu-64-16:~#
```

.3.3.2 Configure the service

Next is we have to create a folder where we want our service to run in.

For that we usually use the home directory as starting point. For our example we create the following directory structure:

```
/home/root/mdcmcmd
```

To create the directories under home we either use putty or we also could use an SFTP client like fillezilla to create these directories (<https://filezilla-project.org/download.php?type=client>)

An important thing is that the folder mdcmscmd has full write and execute rights.



We copy the mdcmd.jar into the mdcmscmd folder.

Server: /home/root/mdcmscmd

- /
- bin
- boot
- dev
- etc
- home
 - root
 - mdcmscmd
- initrd.img
- initrd.img.old
- lib

Dateiname	Dateigröße	Dateityp	Zuletzt geändert
..			
mdcmd.jar	572'648	Executable ...	13.07.2015 22:26:1



Now we make sure that the mdcmscmd folder has full right (777).

The screenshot shows a file manager window with the server path `/home/root/mdcmscmd`. The directory tree on the left shows the following structure:

- bin
- boot
- dev
- etc
- home
 - root
 - mdcmscmd
- initrd.img
- initrd.img.old
- lib

The main pane shows the contents of the `mdcmscmd` directory:

Dateiname	Dateigröße	Dateityp	Zuletzt geändert
..			
mdcmd.jar	572'648	Executable ...	13.07.2015 22:26:1

A dialog box titled "Dateiattribute ändern" is open, showing the following settings:

- Bitte wählen Sie die neuen Attribute für das Verzeichnis "mdcmscmd" aus.
- Besitzer-Berechtigungen: Lesen, Schreiben, Ausführen
- Gruppen-Berechtigungen: Lesen, Schreiben, Ausführen
- Öffentliche Berechtigungen: Lesen, Schreiben, Ausführen
- Numerischer Wert:
- Verwenden Sie ein 'x', um die ursprünglichen Berechtigungen beizubehalten.
- Unterverzeichnisse einbeziehen
- Auf alle Dateien und Verzeichnisse anwenden
- Nur auf Dateien anwenden
- Nur auf Verzeichnisse anwenden

Buttons: OK, Abbrechen



Next we copy the mdcmscmd.conf file into the /etc/init folder:

Dateiname	Dateigröße	Dateityp	Zuletzt geändert
hostname.conf	284	CONF-Datei	23.07.2013 11:2
hwclock-save.conf	444	CONF-Datei	16.04.2014 18:0
hwclock.conf	557	CONF-Datei	16.04.2014 18:0
irqbalance.conf	579	CONF-Datei	26.08.2014 15:3
kmod.conf	689	CONF-Datei	10.04.2014 15:3
mdcmscmd.conf	252	CONF-Datei	13.07.2015 22:2
mountall-bootclean.sh.conf	268	CONF-Datei	22.02.2014 02:4

The mdcmscmd.conf file looks the following:

```
description "MDCMSCmdService"  
author "Rene Unternaehrer"  
  
start on runlevel [2345]  
stop on runlevel [!2345]  
  
expect fork  
  
script  
sudo java -jar /home/root/mdcmscmd/mdcmd.jar start /home/root/mdcmscmd/drops &> /var/log/mdcmscmd.log  
end script
```



The mdcmscmd.conf.permission file looks the following:

```
description "MDCMSCmdService"
author "Rene Unternaehrer"

start on runlevel [2345]
stop on runlevel [!2345]

expect fork

script
sudo java -jar /home/root/mdcmscmd/mdcmd.jar start /home/root/mdcmscmd/drops 777 &>
/var/log/mdcmscmd.log
end script
```

In the *.permission file there is an additional parameter **777**. This is the permission to set on all the files created in the ok, nok, okResult and nokResult folder in order for the FTP to have access to these files. This is usually not needed.

If needed then use this file but rename it to mdcmscmd.conf.

The name of the file will be also the name of the service.

Now that we have copied the two necessary files we can start the service.

Upstart delivers the **initctl** command to observe and control jobs.

List all the jobs:

- Initctl list

If upstart is not installed yet you will get the following message:

```
root@Ubuntu-64-14:~# initctl list
The program 'initctl' is currently not installed. You can install it by typing:
apt install upstart
root@Ubuntu-64-14:~# apt
```

Then try to install upstart:

- Apt install upstart

It might need a install upstart-sysv

- Sudo apt-get install upstart-sysv

It might need a reboot:

- Sudo reboot

Then try again to list the jobs:

- Initctl list



Show status of a job:

- Initctl status <jobname>

Start a job:

- Initctl start <jobname>

Stop a job:

- Initctl stop <jobname>

Reload the configuration:

- Initctl reload-configuration

We first want to see the status of the job mdcmscmd:

```
root@Ubuntu:~# initctl status mdcmscmd
mdcmscmd stop/waiting
root@Ubuntu:~# █
```

If we would not have copied the mdcmscmd.conf file there we would get:

```
root@Ubuntu:~# initctl status mdcmscmd
initctl: Unknown job: mdcmscmd
root@Ubuntu:~# █
```

Now we can try to start the job:

```
root@Ubuntu:~# sudo initctl start mdcmscmd
mdcmscmd start/running, process 6185
root@Ubuntu:~# █
```



After we started the job the first time. We have a look at the directories:

Server: /home/root/mdcmcmd/drops

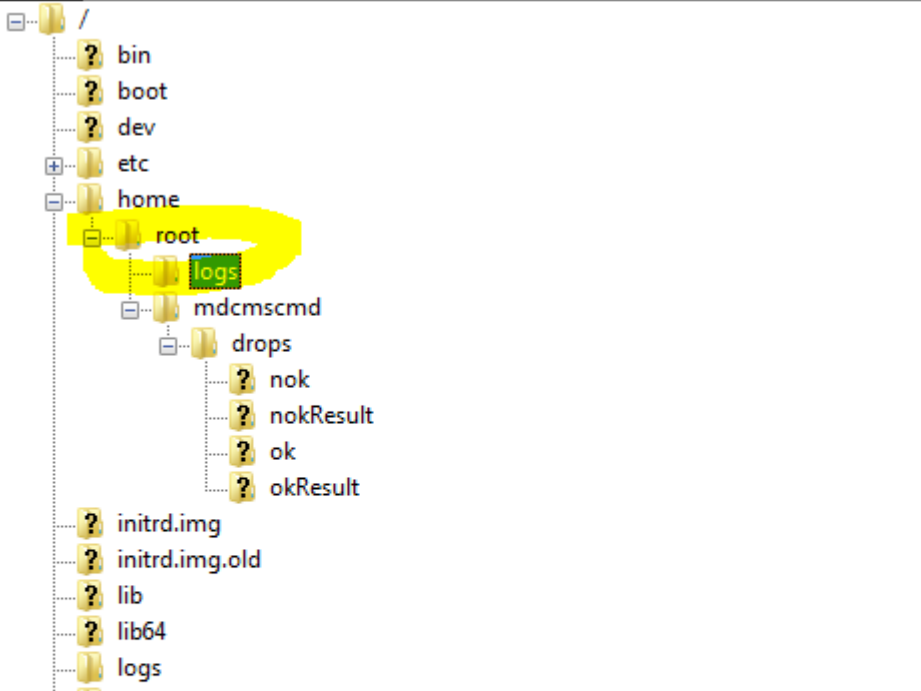
Dateiname	Dateigröße	Dateityp
..		
nok		Dateiordner
nokResult		Dateiordner
ok		Dateiordner
okResult		Dateiordner

At the beginning the service created the drops folder with the nok, nokResult, ok and okResult folders.

In the drops folder the scripts are copied that have to be executed. We simulate that later.

The service also created the logs folder:

Server: `/home/root/logs`



Dateiname	Dateigröße	Dateityp
..		
mdcmsService_Debug.log	1'454	LOG-Datei
mdcmsService_Error.log	0	LOG-Datei
mdcmsService_Fatal.log	0	LOG-Datei
mdcmsService_Info.log	1'169	LOG-Datei
mdcmsService_Warn.log	0	LOG-Datei

In the logs we can track what the service was doing so far.

Some usefull links:

<http://upstart.ubuntu.com/getting-started.html>

<https://wiki.debian.org/Upstart>

<http://askubuntu.com/questions/587631/ubuntu-14-04-server-launch-jar-in-screen-on-boot-restart>

4. Testing and Using the Service

..4.1 Testing locally

After getting the service up and running, you can test its behaviour.

In the folder testscripts-linux there are testscripts

- ⇒ ls, ls1-4.txt with the same content. They should all work and go to the ok folder.
- ⇒ Ls-wrong, ls-wrong1-4.txt. They should all not work and go to the nok folder.

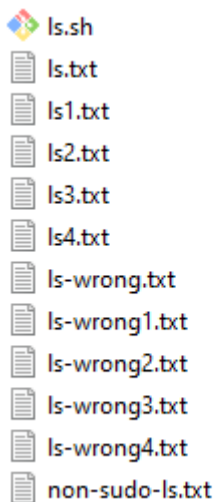
- 1) Copy the all the files into the **drops** folder for your service
- 2) within a few seconds, the files should be moved to the ok or nok folder within the drops folder

In both cases, the logs should show what and if the process performed.

..4.2 Test example

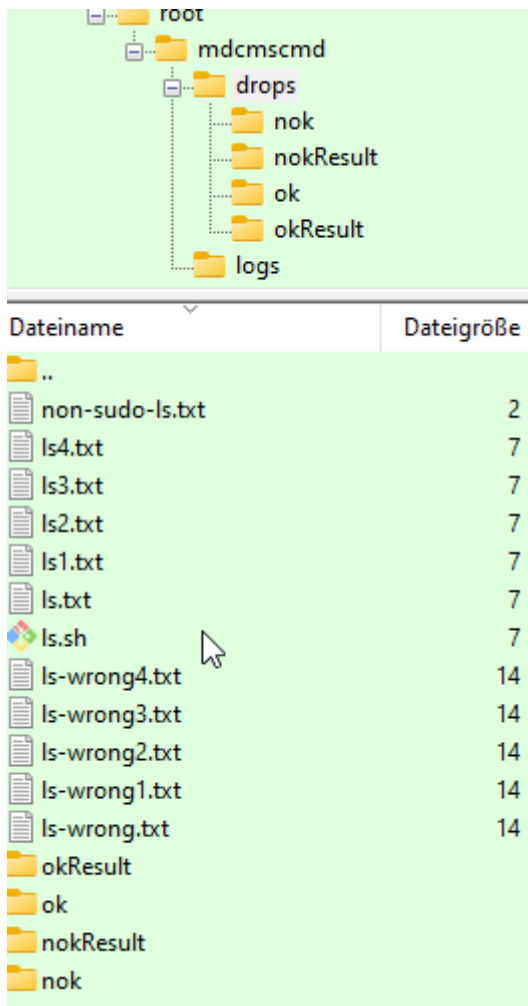
In the downloaded zipfile we provided some easy sample scripts that should either give a ok result or nok result.

The files are in the testscripts-linux folder:



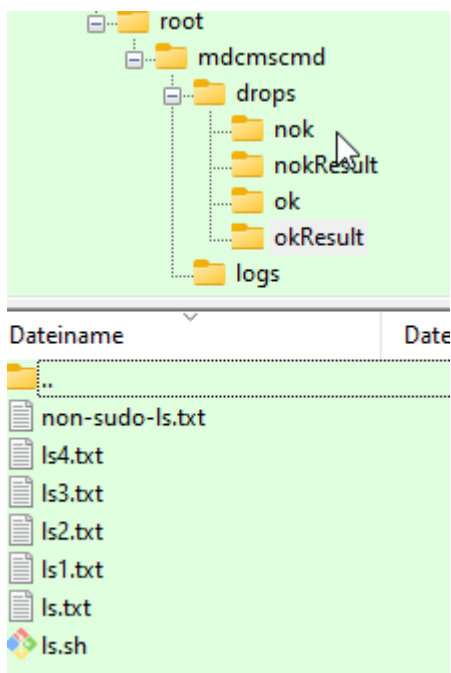
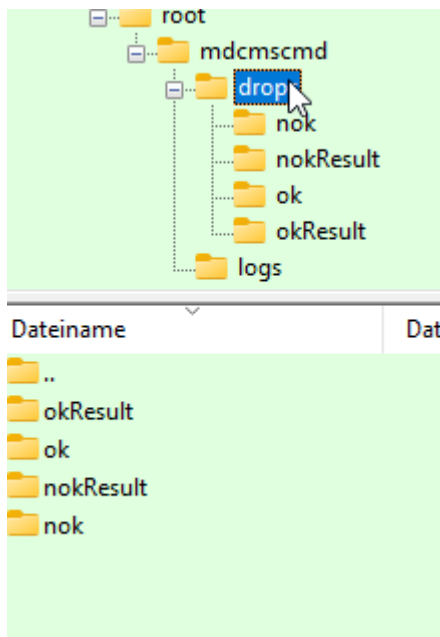
The ls*.txt ls.sh and non-sudo-ls.txt should perform ok and land in the ok, okResult folder
The other ls-wrong*.txt should perform nok and land in the nok, nokResult folder

Copy all these scripts when the mdcmscmd service is running into the drops folder:



The service will read now one file after the other and run the commands within the file.

Every file that is run disappears from the drops folder. When successful run then the file will be written into the ok folder and the result of the run into the okResult folder. When unsuccessful run then the file will be written into the nok folder and the result of the run into the nokResult folder.



Example: ls.txt

➤ Sudo ls

The ls.txt in the ok folder looks the same

➤ Sudo ls

The ls.txt in the okResult folder has the result

drops

logs

mdcmd.jar

5. Usage in RFP

In MDCMS, an RFP contains one or more objects that are to be installed by MDCMS.

The target location for an object is based on its attribute.

If the attribute is of type *REMOTE, a Remote Server Location is defined for it and 0 or more scripts are defined to be run either before or after installation of the object. The script can be set to run once per object or once per RFP.

At installation time, MDCMS uses FTP to send scripts, which are stored as templates within IFS on the IBM System i (AS/400), to the drops folder.

If the service is started, it will pick up the script and attempt to execute each command within it. If all commands execute without exception, the script is moved to the ok folder, which MDCMS checks to know that the script processed.

If a command fails, the script is moved to the nok folder, which MDCMS checks to know that the script did not process.