MDC

Table of Contents

Architecture description

Application consists of 3 layers:

1. User Portal
2. Database server
3. PowerShell scripts

Layers can be located on different physical workstations.

No any business logic with Active Directory on Portal level. Only PowerShell scripts can make changes in AD.

User Portal is an interface for users to request Active Directory changes (assign to services, remove from services).

Database is used to store information about users, assigned services and history, exchange data between Portal and PowerShell layer.

PowerShell scripts are used to execute commands created by Portal and import/synchronize data from PDB.

Minimal system requirements

This topic contains the information of hardware and software requirements to run the application.

Requirements for the Portal

1. Linux or Windows Server.
2. Minimal PHP version: 5.5.26. PHP is freeware. Download page: http://php.net/
3. Apache HTTP Server or IIS server.
4. Source code. Copy to the PHP server working folder.
5. Minimal hardware requirements:
    a. CPU: 1.4 GHz (64-bit) or faster multi-core.
    b. Memory (RAM): 2 GB.
    c. Hard disks and available storage space: 160 GB hard disk with a 60 GB system partition.
6. System language requirements: any language.

Requirements for the database layer

1. PostgreSQL is freeware.
2. Minimal version 9.4. Download page: http://www.postgresql.org/download/
3. Install pgAdmin. Download page: http://www.pgadmin.org/download/
4. Database backup file to restore schema and initial data.
5. Database name: “MDC”.
6. Minimal hardware requirements:
    a. CPU: 1.4 GHz (64-bit) or faster multi-core.
    b. Memory (RAM): 2 GB.
    c. Hard disks and available storage space: 160 GB hard disk with a 60 GB system partition.
7. System language requirements: any language.
8. Collation: UTF-8.

Requirements for the PowerShell layer

1. Minimal version: 2.0.
2. Windows Server 2008, 2008 R2, or 2012, 2012 R2.
3. Minimal hardware requirements:
    a. CPU: 1.4 GHz (64-bit) or faster multi-core.
    b. Memory (RAM): 2 GB.
    c. Hard disks and available storage space: 160 GB hard disk with a 60 GB system partition.
    d. ODBC driver for PostgreSQL. Download page: https://odbc.postgresql.org/

Working environment description

This topic addresses the information you need to configure environment for project's development.

Environment for the Portal

Application built with PHP framework CodeIgniter. CodeIgniter official web-site: https://www.codeigniter.com/

For UI used:

- jQuery http://jquery.com/
- Twitter Bootstrap http://getbootstrap.com/
- jQuery DataTables plugin https://datatables.net/
- jNotifyBar plugin

All scripts are stored locally in the application's folder.

Source code

Portal built with an open source framework CodeIgniter. Official web-site: https://codeigniter.com/

Source code location: /var/www/palm/docroot

Main execution file is **index.php**. Location: root of application. This file contains different configurations depending on current environment (development, testing, production).

Caution: do not overwrite this file as copy-paste on test and production environment.

Configuration files

Configuration files location: /application/config/

Portal configuration stored in file **config.php**

This CodeIgniter configuration file also contains custom MDC configuration items – "MDC Application" section.

Connection to database

Configuration to connect database is stored in file **database.php**

Password configuration line:
$db['default']['password'] = decrypt(ENCRYPTION_KEY, '[encryped_password]');

Database connection password is encrypted.

How to encrypt password

Portal have a page to encrypt password: /index.php/Encrypt/db_pwd

Enter password in the text box and press [Encrypt] button.

Environment for the PowerShell layer

PS scripts are used to change data in Active Directory.

Files location

PowerShell scripts can be located in any folder on the local computer. Files MDC-Execute.ps1 must be located in the same folder with other PS files.

Connection to database

PowerShell has no built-in support code for the PostgreSQL protocol, so it therefore cannot communicate with PostgreSQL without some kind of client driver.

- ODBC driver for PostgreSQL should be installed: https://odbc.postgresql.org/
- ODBC driver should be configured.

ODBC driver configuration

1. Open ODBC Data Source Administrator
2. Click "Add…" button.
3. Choose "PostgreSQL Unicode(x64)".
4. Click "Finish".
5. In the driver setup dialog fill information about database name, server address, username, password.
6. Click "Test" to validate setup.
7. If connection successful - press "Save".

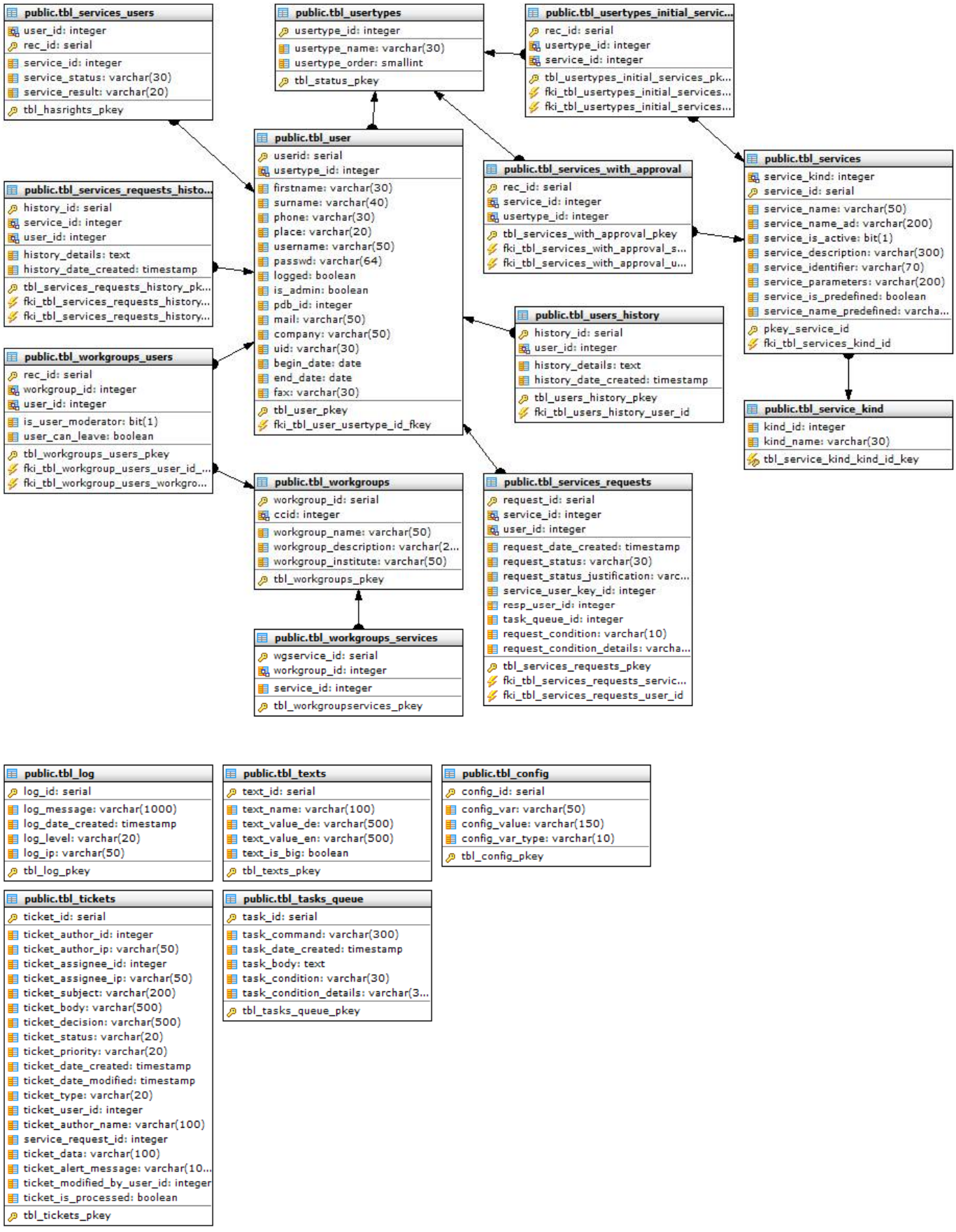
How to connect database from PowerShell script

Use DSN name as connection string in the object System.Data.Odbc.OdbcConnection. Default DSN name for PostgreSQL is: `PostgreSQL35W`

Database description

Table	Description
tbl_config	Application configuration
tbl_log	Contains application log.
tbl_service_kind	List of service types.
tbl_services	List of application services (WLAN, VPN, etc.)
tbl_services_requests	Information about services requested by user.
tbl_services_requests_history	Contains history of the service.
tbl_services_users	Contains information about services assigned to user.
tbl_services_with_approval	List of services with approval by Cost Center responsible person.
tbl_tasks_queue	List of Portal commands.
tbl_texts	Portal text resources.
tbl_user	List of Portal users.
tbl_users_history	Contains history of users activity.
tbl_usertypes	List of user types.
tbl_usertypes_initial_services	Contains information about default services for each user type.
tbl_workgroups	List of Cost Centers.
tbl_workgroups_services	List of services in Cost Centers.
tbl_workgroups_users	List of users in Cost Centers.

ERM of the database



Portal description

User interface built to request changes in Active Directory and see information about users (personal data, services, etc.).

Authentication on the Portal

Portal using adLDAP library to authorize users.

adLDAP official web-site: http://adldap.sourceforge.net/

API examples: http://adldap.sourceforge.net/wiki/doku.php?id=api_examples

Only active Active Directory users can get access to the Portal. If user is disabled in the Active Directory, username or password is incorrect – no access to the Portal.

Information about security

To access any page on the Portal user should be authenticated. Even user manually change the URL – Portal's security system redirects to the login page.

Permission-driven user interface

Portal have a user interface in which the appearance or non-appearance of menu items or functions is driven by the access permissions granted to user.

User roles description

Portal user can have different roles. Portal roles are stored in the database.

- Regular user
- Administrator
- Cost Center responsible person

Regular user can see profile information and request/remove services.

Administrator is a user with the attribute is_admin = TRUE in the database table "tbl_user".

Cost Center responsible person is a user which have a relation with a cost center and database attribute is_user_moderator = TRUE

Multilingual interface

Language resource files are stored in: /var/www/palm/docroot/application/language

How to synchronize language resources between developers (and databases)

1. Developer create a new language key, e.g.: lang('label_service_name')
2. On a page "/index.php/Admin/texts" press button [Generate resource files]. It creates "common_lang.php" file for each language.
3. Check-in pending "common_lang.php" files in TFS.
4. Other developer on a page "/index.php/Admin/texts" should press button [Synchronize]. System reads all "common_lang.php" files and creates resource key in database if key is missing.
   (!) Existing resources in database stay without changes.

How to deploy resources

1. On a page "/index.php/Admin/texts" press button [Generate resource files].
2. Copy "common_lang.php" to server for each language folder.
3. On a page "/index.php/Admin/texts" press button [Synchronize] to update database.
   (!) Existing resources in database stay without changes.

Services requests for users

Each authenticated Portal user can request or remove service. Some services should be accepted by Cost Center's responsible person. Responsible person can accept or reject requested service.

Commands for PowerShell

Portal cannot make changes in Active Directory. To request changes in AD, Portal creates a command for PowerShell. Command's data stored in JSON format. When user requests assignment to service or removal – for all these operations Portal create a command.

Command contains the full information how to operate with data (username, user database id, service information, etc.).

Requests for administrators

Some operations require confirmation by IT administrator. Requests types:

- UserExternalEmail
- DeactivateUserAccount
- RemoveUserServices

Administrator should accept or decline options in the request.

PowerShell layer description

PowerShell module

PS module for MDC project contains a list of common functions like connection to database, get Portal command data, update data in the database, etc.

Module should be located in the directory *%SystemRoot%\System32\WindowsPowerShell\v1.0\Modules\MDC*

Module filename – MDC.psm1

Infrastructure configuration

To run commands execution script and maintenance script use Task Scheduler. These tasks should run with the highest privileges.

Commands execution scripts

To execute commands created by Portal is used script MDC-Execute.ps1 This scripts read information from database table tbl_tasks_queue , parse data and execute each command. Execution business rules described below.

Command	**Description**
AssignUserToWLANService	PS script to assign user to the AD service.
AssignUserToVPNService	PS script to assign user to the AD service.
AssignUserFaxNumber	PS script to set user's fax number.
CreateHomeDirectory	PS script to create folder.
SetupInternalEmail	PS script to assign user to the AD service.
SetupExternalEmail	PS script to request setup of user's external email account.
RemoveUserFromWLANService	PS script to remove users from AD service.
RemoveUserFromVPNService	PS script to remove users from AD service.
RemoveUserFaxNumber	PS script to remove user's fax number.
DeactivateUserAccount	PS script to deactivate user account.

Each script read information about Portal command by task ID and parse command data.

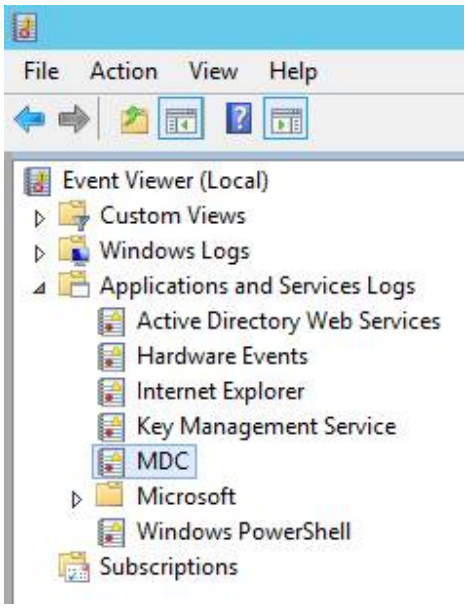
PowerShell scripts logging

PS scripts can write logs. Functions to write logs are implemented in the PowerShell module. Logs are writing into the database (table “tbl_log”) and Windows Events (Event Viewer).

MDC log in Event Viewer:

- Name: MDC
- Location: Application and Services Logs

Windows Event Viewer



Import/synchronize data

To import data from PDB and synchronize database, MDC application using script MDC-Sync.ps1. This script runs from scheduled task. Script must be configured to read export file, for example:

[XML]$Users = Get-Content C:\incoming\palm_export.xml

PDB export file location: C:\incoming\palm_export.xml

Application deployment

Database

Database engine should be installed and configured. Database name can be any. Default database name on the test environment is: palm_test

Portal deployment

Note: on the Linux server file names are case-sensitive.

Application base dir: /var/www/palm/docroot

Configuration files location: /application/config/

File config.php should be configured with base site URL – WITH a trailing slash:

URL can be changed by administrator to any value. Examples:

- Test environment: https://palm-test.mdc-berlin.net/
- Production environment: https://portal.mdc-berlin.net/

File database.php should be configured to connect PostgreSQL database:

- Variable **$query_builder** must have value **TRUE**.
- $db['default']['hostname'] = '<host address>'; // can be changed
- $db['default']['username'] = 'palm_test'; // can be changed
- $db['default']['password'] = decrypt(ENCRYPTION_KEY, '<encrypted database password>');
- $db['default']['database'] = '<database name>'; // can be changed
- $db['default']['dbdriver'] = 'postgre';
- $db['default']['dbprefix'] = '';
- $db['default']['pconnect'] = TRUE;
- $db['default']['db_debug'] = FALSE;
- $db['default']['cache_on'] = FALSE;
- $db['default']['cachedir'] = '';
- $db['default']['char_set'] = 'utf8';
- $db['default']['dbcollat'] = 'utf8_general_ci';
- $db['default']['swap_pre'] = '';
- $db['default']['autoinit'] = TRUE;
- $db['default']['stricton'] = FALSE;
- $db['default']['port'] = <port>; // can be changed

adLDAP configuration

adLDAP library is used to operate with Active Directory.

File location: /application/config/Adldap.php

```
$config['account_suffix'] = '<account suffix>';
$config['base_dn'] = '<base>';
$config['domain_controllers'] = array ("<AD controller>");
$config['ad_username'] = '<username with rights to change AD>';
$config['ad_password'] = '<AD manager password>';
$config['real_primarygroup'] = true;
$config['recursive_groups'] = true;
```

PowerShell scripts

PS scripts can be located in any folder. For example: C:\PSScripts

List of PS scripts:

- MDC-Sync.ps1
- MDC-Execute.ps1
- AssignUserFaxNumber.ps1
- AssignUserToService.ps1
- AssignUserToVPNService.ps1
- AssignUserToWLANService.ps1
- CreateUserHomeDirectory.ps1
- DeactivateUserAccount.ps1
- RemoveUserFaxNumber.ps1
- RemoveUserFromService.ps1
- RemoveUserFromVPNService.ps1
- RemoveUserFromWLANService.ps1
- SetupExternalEmail.ps1
- SetupInternalEmail.ps1

PowerShell module for MDC

Module file “MDC.psm1” and manifest file “MDC.psd1” should be located in folder:
%SystemRoot%\System32\WindowsPowerShell\v1.0\Modules\MDC

To use module with name “MDC” – *Import-Module MDC* - names MUST be:

- **MDC**.psm1
- **MDC**.psd1
- Folder name: **MDC**

Third-party utilities used in PS scripts

Some PS scripts uses external utilities to set permissions to the user’s home directory, send an email to provider, etc.

SetACL

SetACL is a freeware utility for manipulating security on Microsoft Windows. Used in the PS script to create user’s home directory.

Official web-site: https://helgeklein.com/setacl/

SetACL location: C:\Utils\SetACL.exe

Blat

Blat is a freeware command line utility to send an email to provider. Used in the external email service request.

Official web-site: http://www.blat.net/

Blat location: C:\Utils\blat\blat.exe

Tasks configuration in the Task Scheduler

PowerShell layer must have 2 scheduled tasks:

- MDC Execute
- MDC Sync

How to create and configure tasks

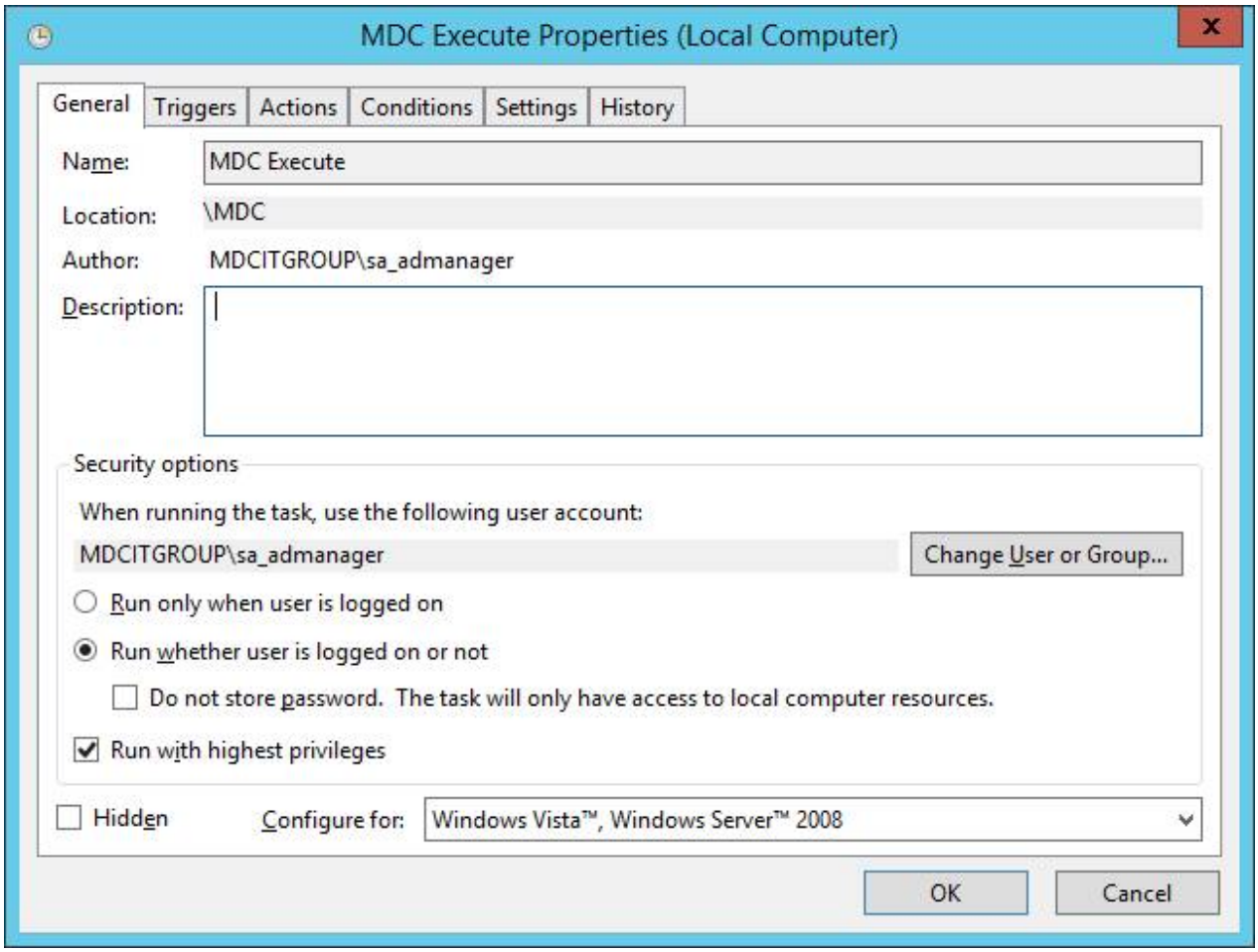
Open Task Scheduler.

Task “MDC Execute”

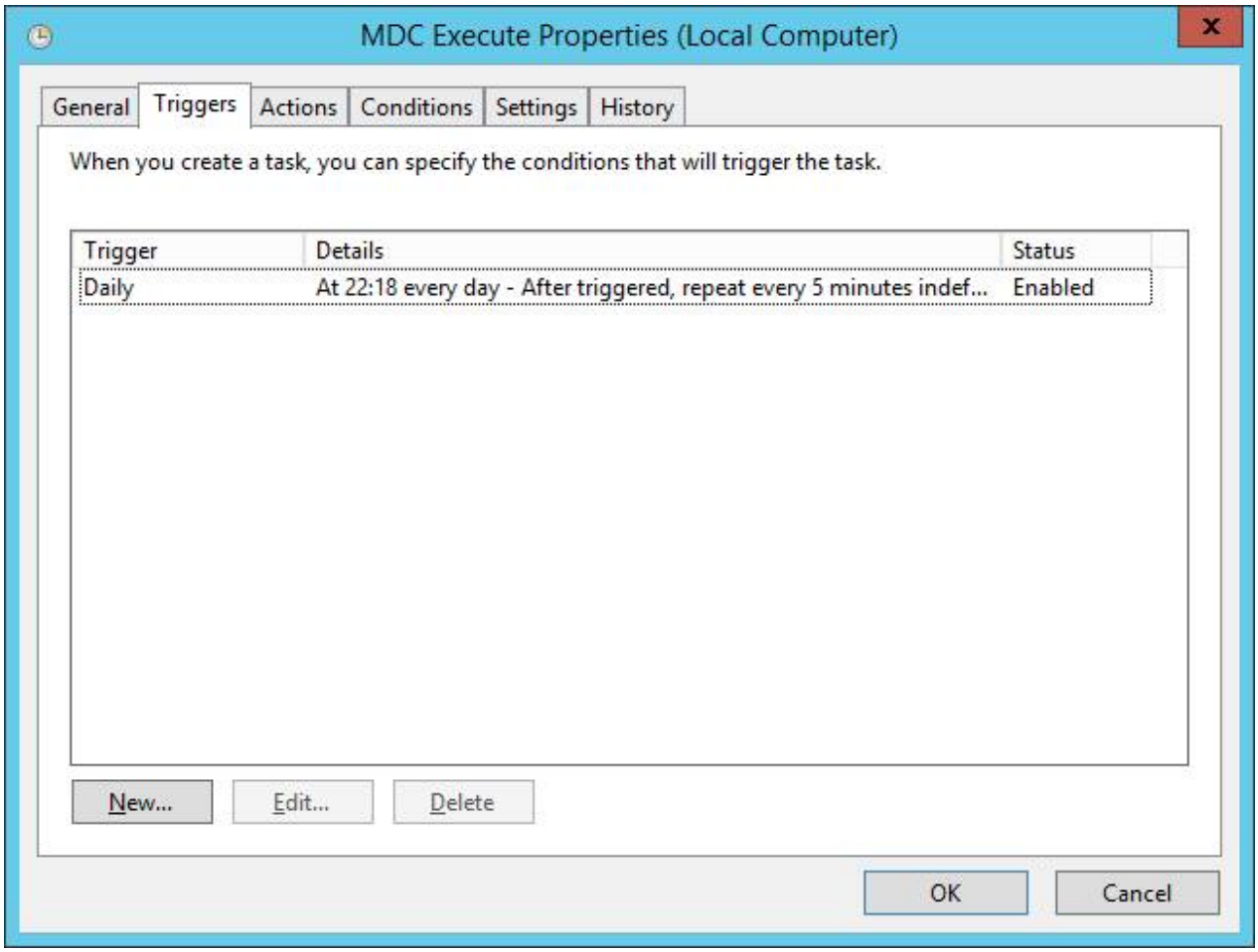
Click on “Create Task…”

Tab “General”
Name: MDC Execute
“Security options” section:

- When running the task, use the following user account: MDCITGROUP\sa_admanager
- Check radio button “Run whether user is logged or not”
- Check “Run with highest provileges”



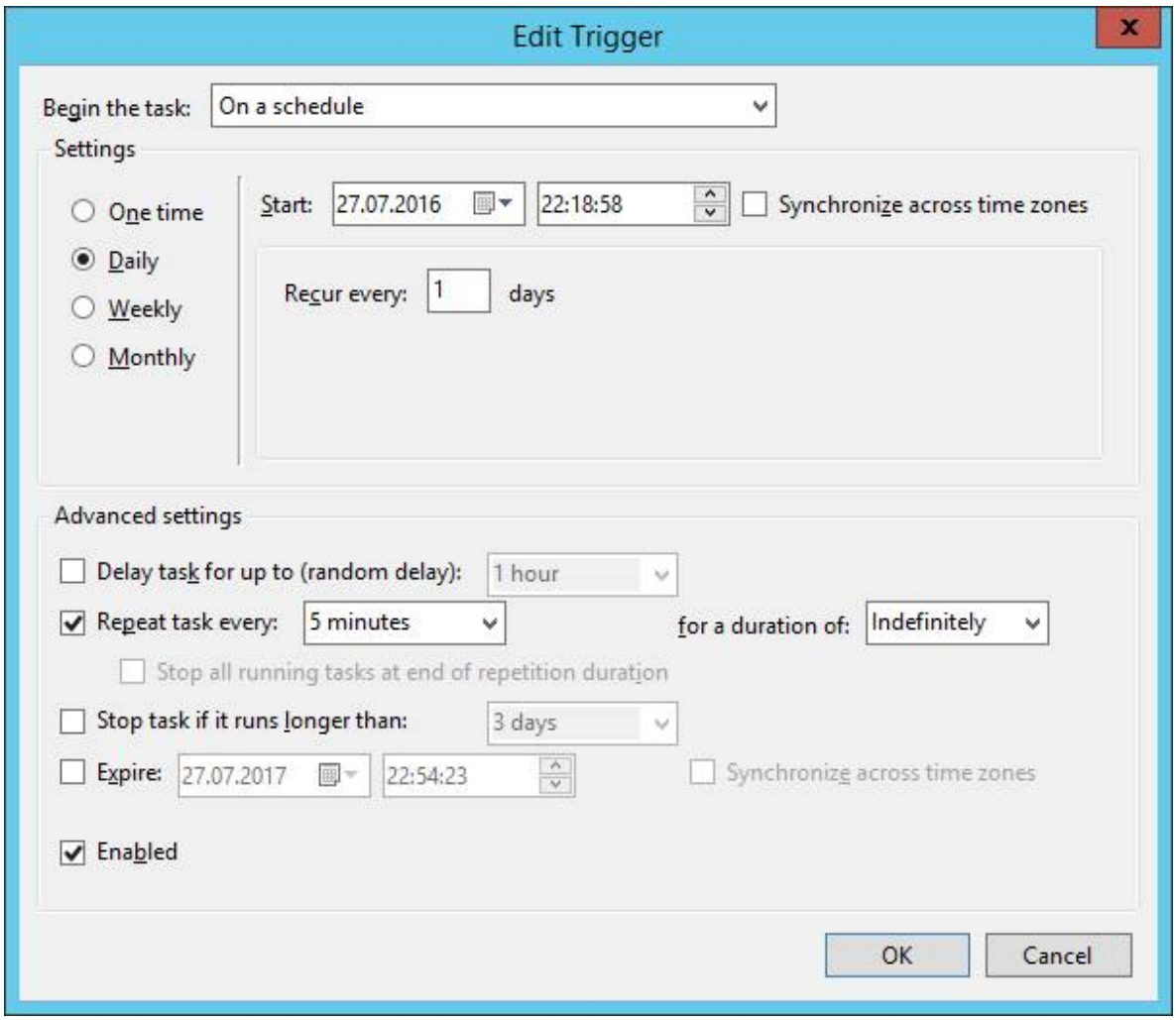
Tab “Triggers”
Click “New…” button.



“Triggers” > “Edit Trigger” dialog

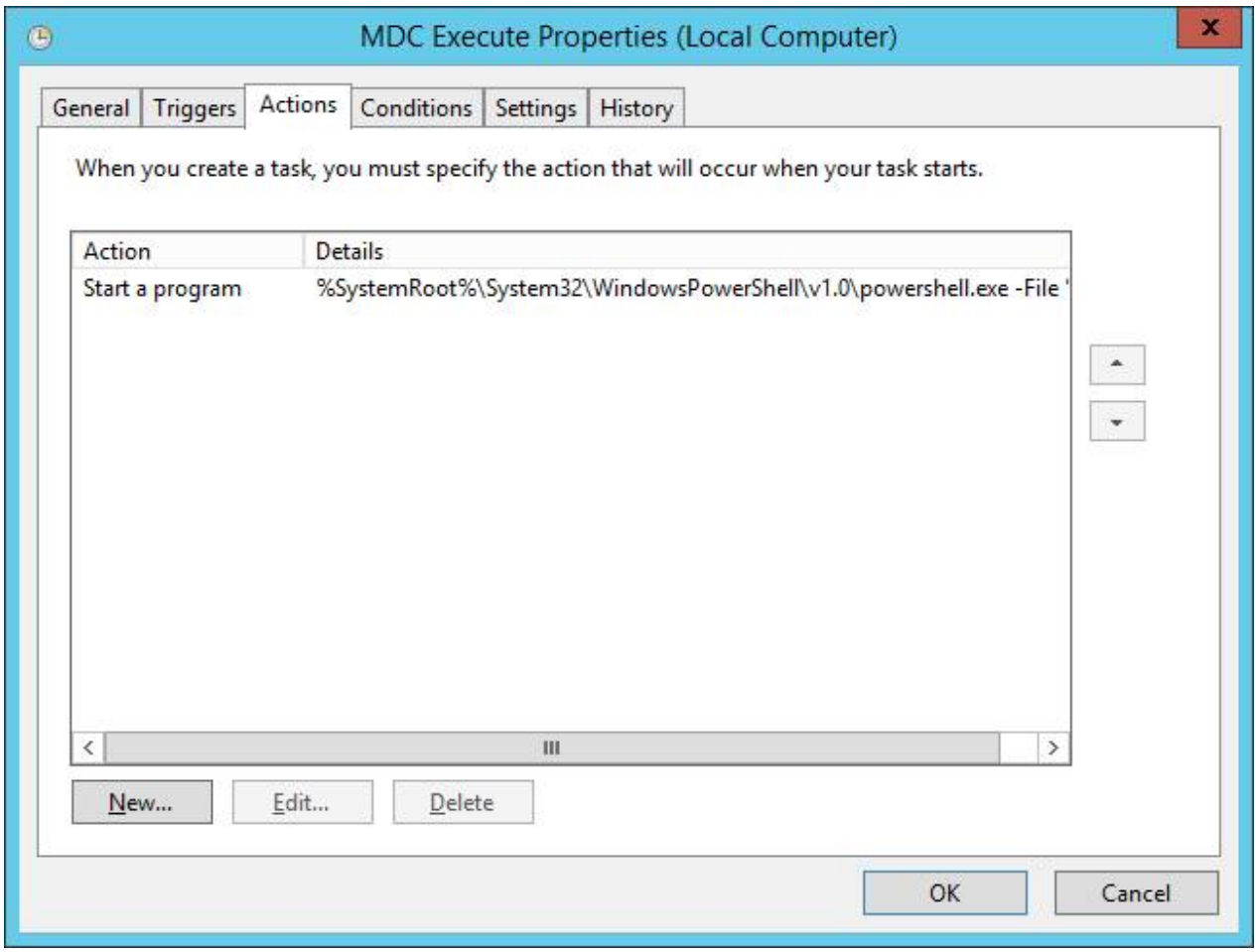
In the “Settings” section check radio button “Daily”.
In the “Advanced settings” section:

- Check “Repeat task every:”
    - Choose period (recommended 5 minutes)



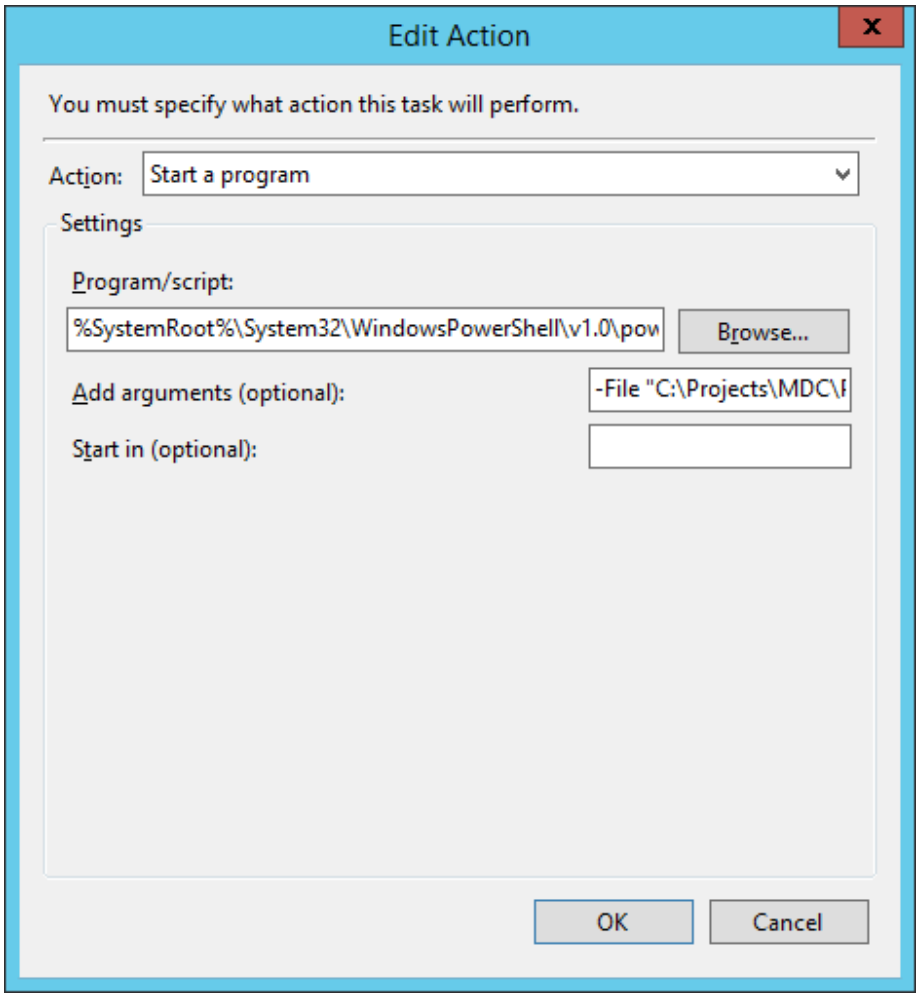
Tab “Actions”

Click “New…” button.

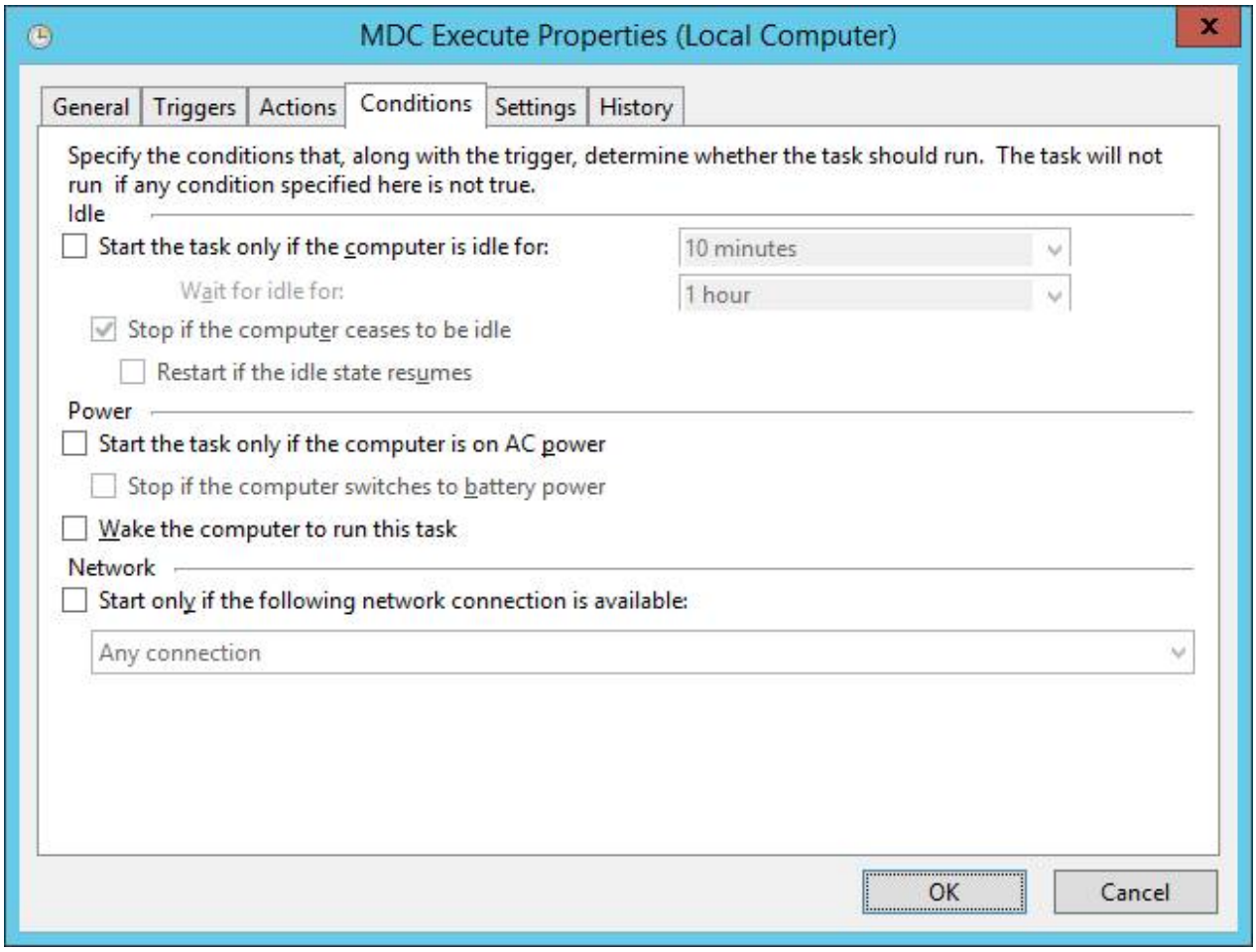


“Actions” > “Edit Action” dialog

Program/script: %SystemRoot%\System32\WindowsPowerShell\v1.0\powershell.exe
Add arguments (optional): -File "C:\Projects\MDC\PowerShell\MDC-Execute.ps1" -ExecutionPolicy Unrestricted

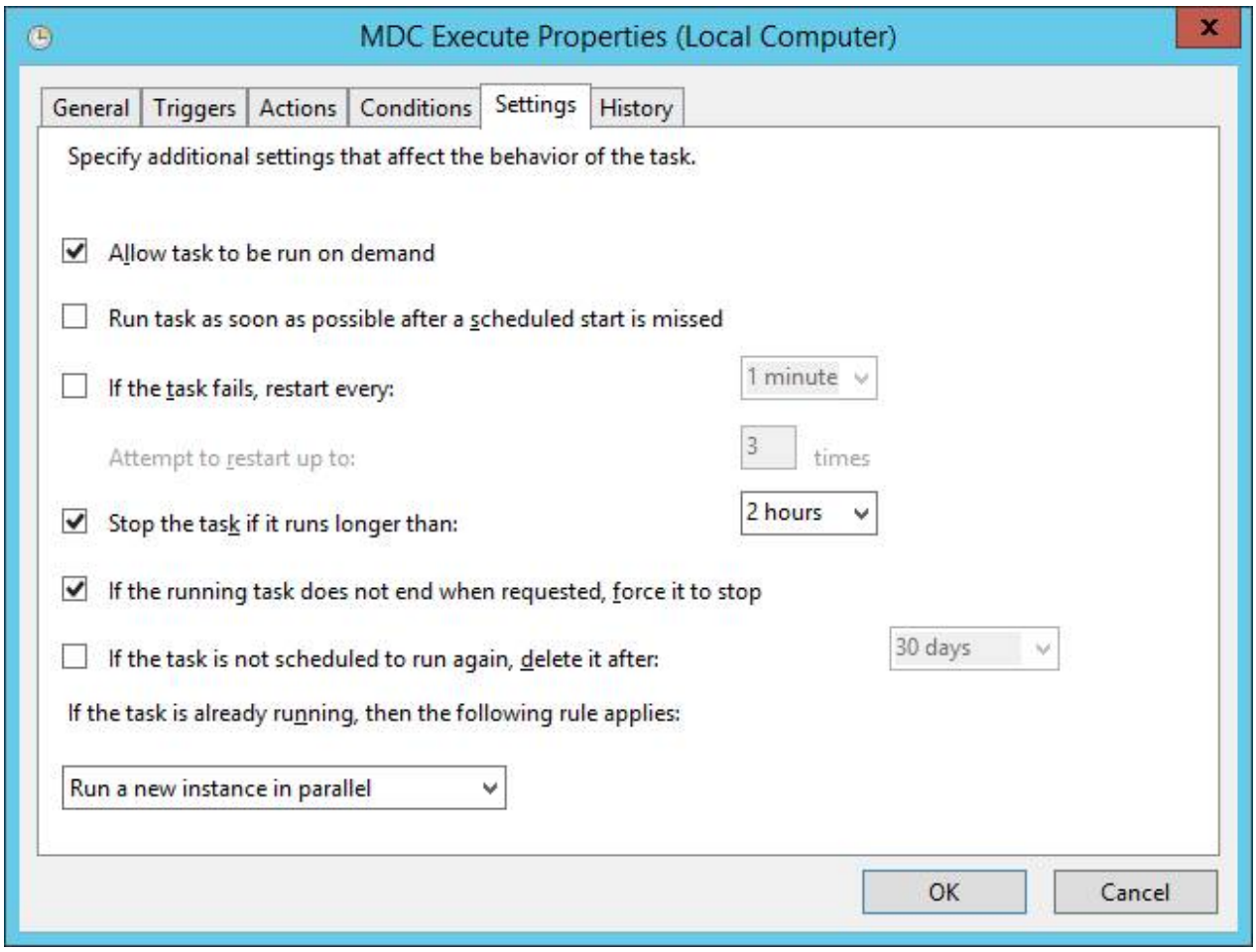


Tab “Conditions”



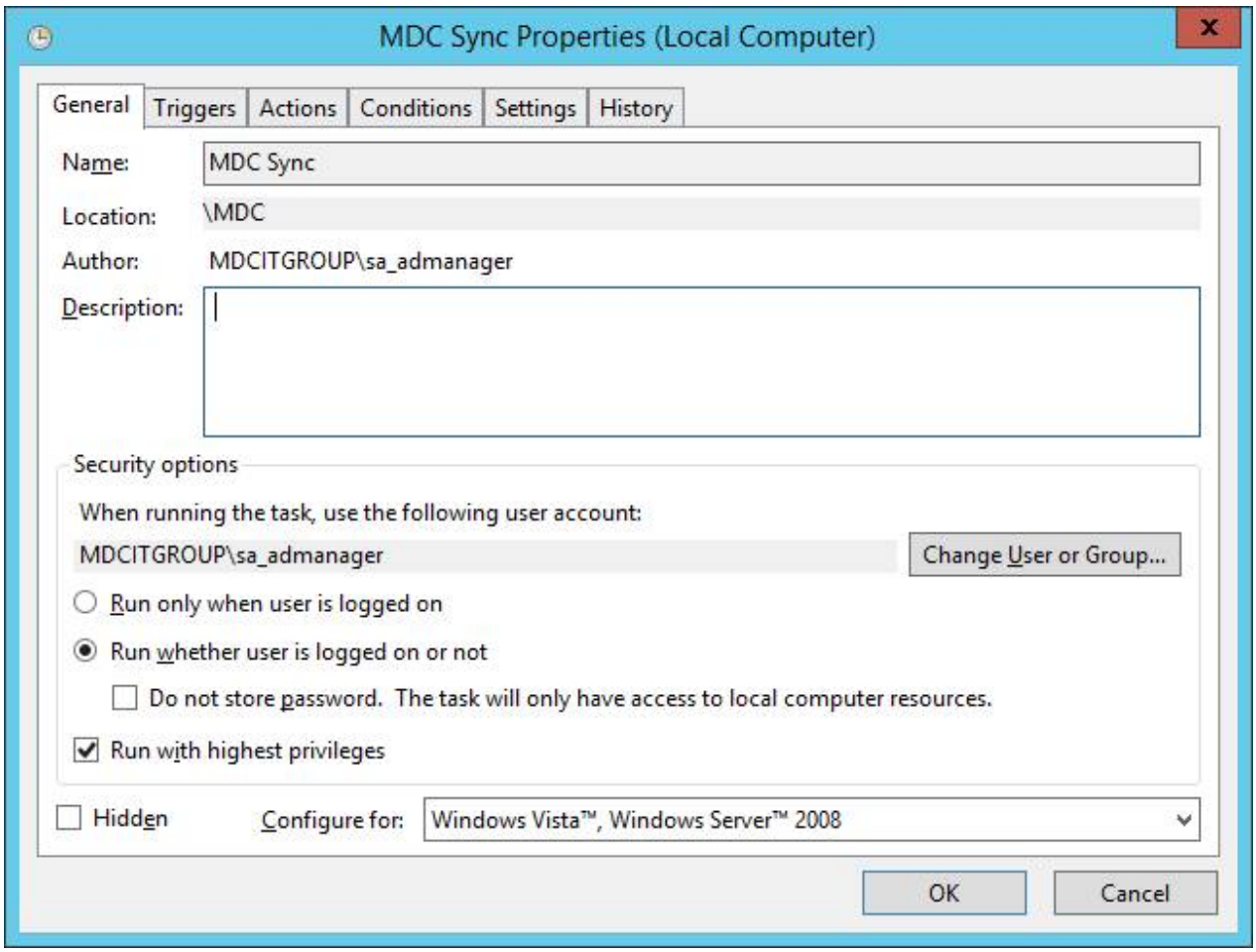
Tab “Settings”

Important: select “Run a new instance in parallel”.

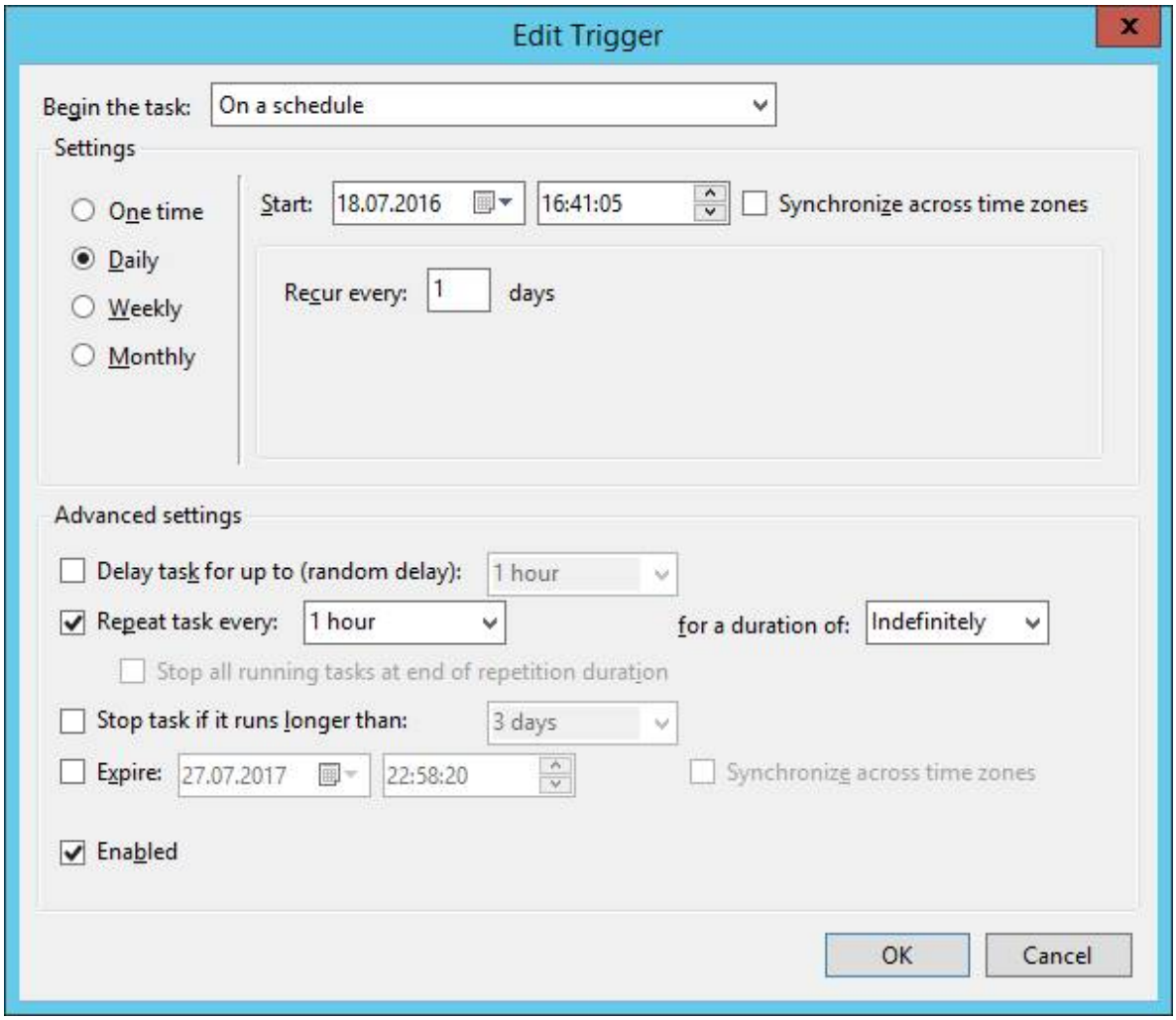


Task “MDC Sync”

Configuration the same as “MDC Execute”.

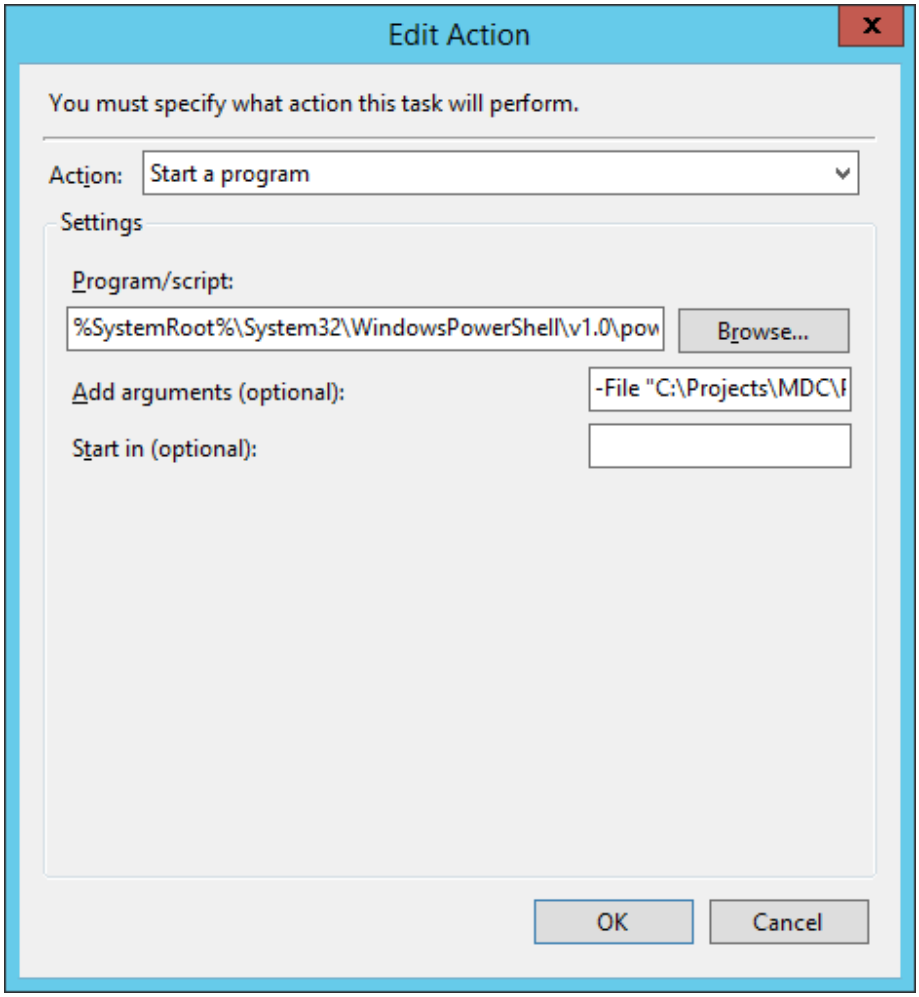


“Triggers” > “Edit Trigger” dialog



Tab “Actions” > “Edit Action” dialog

Program/script: %SystemRoot%\System32\WindowsPowerShell\v1.0\powershell.exe
Add arguments (optional): -File "C:\Projects\MDC\PowerShell\MDC-Sync.ps1" -ExecutionPolicy Unrestricted



Tab “Settings”

Important: select “Run a new instance in parallel”.