



CYBERARK RESPONSE MANAGER

Name of Company: CyberArk

Website: www.cyberark.com

Name of Product: CARM

Version: 1.0.0

Date: July 2017

OVERVIEW

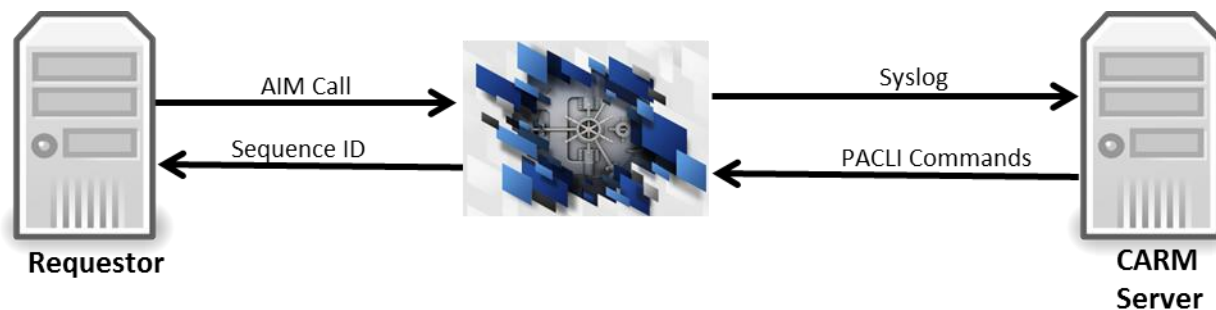
The CyberArk Response Manager (CARM) is a powerful tool that allows Cyber Security orchestrators to trigger actions in CyberArk that can enable/disable users, change security policies, force a credential change, etc. It also provides enriched information that can help the orchestrator make a decision about a security event.

USE CASES OVERVIEW

- 1) Raise the Security Policy of a Privileged Account. Under certain Circumstances you might want to temporarily, or permanently, increase the requirements to show/copy/connect to a specific account. This is achieved by moving the account up to a more restrictive platform. For example, one with Dual Control, or where you must provide a reason for access.
- 2) Lower the Security Policy of a Privileged Account.
- 3) Disable/Enable a CyberArk user. It is possible that, based on behavior patterns, you might want to temporarily, or permanently, prevent a user to access any CyberArk Privileged Account. This is achieved by sampling disabling the user at the Vault. An Enable function is provided if this turned out to be a temporary measure.
- 4) Force a Credential Change. You might detect an event that requires immediate password change of an account managed by CyberArk, without waiting for the next password rotation. This is achieved by marking the account for Immediate Password Change.
- 5) Retrieve the details for CyberArk credentials access (Show/Copy/Connect) to a target account during a specific time period. These details include the action done, the reason entered, user, and time.

PRODUCT DIAGRAM & DESCRIPTION OF PRODUCT INTEGRATION

The Requestor issues an AAM call for an object in the vault, there is a different object for each use case, with the actions to be executed encoded in the Reason for Retrieval. The Vault generates a Syslog event that will be sent to the CARM server where the actions are run. The CARM Server will extract the parameters for action (the name of the object itself is the action to be executed) from the Reason and execute the actions, using PACLI, against the vault.



The AAM call returns a Sequence ID. The CARM server creates an object in the **Temp** folder of the **ResponseManager** safe, named <Action Name>-<Sequence ID> (i.e. ChangeCredentials-1), for tracking the status (or return results) of the action. The Requestor can query this object to determine the status of the action.

INSTALLATION

The CyberArk Response Manager (CARM) does not require a specific directory for installation. Choose a location convenient to you. This applies to all the components below. The Requestor, Vault, and CARM servers must be time synchronized.

VAULT SERVER

- 1) Copy the CARM.xml file, from the distribution, to the CyberArk Vault Install **Syslog** directory on the Vault.
- 2) Edit the **DBPARAM.ini** file on the Vault to add CARM as an additional Syslog server (if you already have a Syslog server, add a comma followed by CARM.xml) sending messages **295,319**, and **320**. Here's an example:

```
[SYSLOG]
SyslogTranslatorFile=Syslog\SplunkCIM.xml,Syslog\CARM.xml
SyslogServerPort=11514
SyslogServerIP=10.0.1.22,10.0.1.30
SyslogServerProtocol=UDP
SyslogMessageCodeFilter=0-999|295,319,320
UseLegacySyslogFormat=No,No
```

For complete instructions to configure a Syslog server, or additional Syslog servers, refer to your PAS implementation guide.

- 3) Add the CARM server IP address to the **SyslogServer** parameter (if you already have a Syslog server, add a comma followed IP of the CARM server IP)
- 4) The **UseLegacySyslogFormat** parameter must be set to **No** for CARM.
- 5) Restart the Vault, using the PrivateArk Server.

CARM SERVER

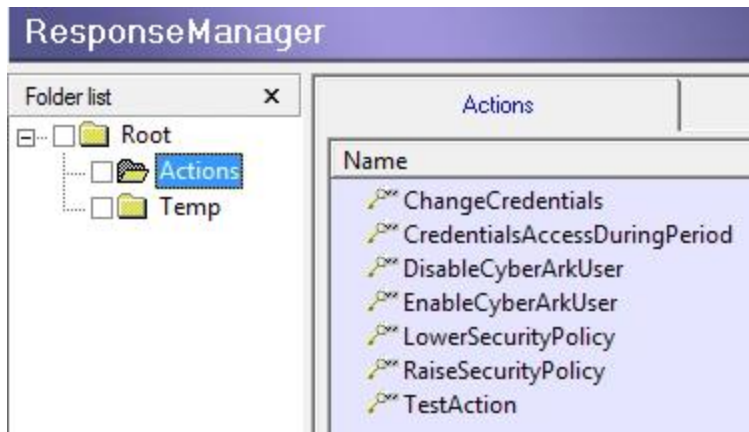
- 1) Extract the following files from the distribution and copy them to the install directory:
CARMService.exe
CARMInstaller.exe
CARM-Config.ini
CreateCredFile.exe
Vault.Proto
- 2) Create a folder named **logs** inside the install directory
- 3) Edit the **CARM-Config.ini** file:
 - a. Set **PROTOCOL** to the same value as the **SyslogServerProtocol** in the Vault **DBPARAM.ini**.
 - b. Set **PORT** to the same number in the **SyslogServerPort** in the Vault **DBPARAM.ini**.
- 4) Install **PACLI** (Contact CyberArk if you don't have a license) and permanently add the installation directory to the **%PATH%** environment variable.
- 5) Run **CARMInstaller.exe**. It will prompt for different parameters along with administrative Vault user credentials to create a user and objects inside the Vault during the installation process. It'll not store/display the credential, nor will it be needed again after the installation is complete:

```
C:\CyberArk-CARM>CARMInstaller.exe
CyberArk Response Manager Installer. Configuration File=CARM-Config.ini Vault File=Vault.ini
The installer will need a CyberArk Vault user with the ability to create users and safes.
Vault Name[Vault]:
Vault Address:10.0.1.10
Vault Port[18581]:
Vault User: Administrator
Password:
Creating user: CARM-user
Creating Safe...
Creating Credential File...
Command ended successfully

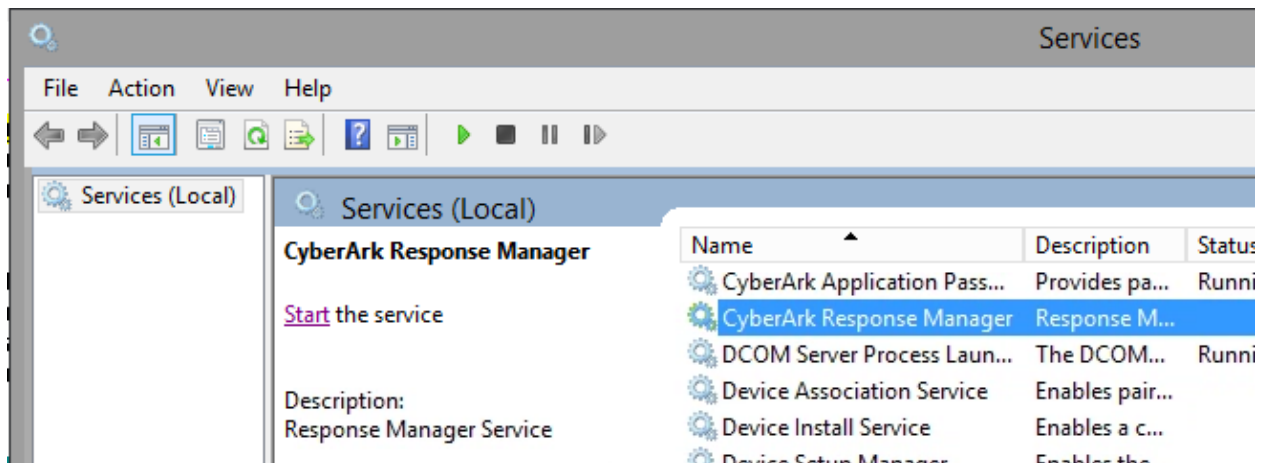
Creating Actions...
Installing the CARM Service...
Installation Complete.

C:\CyberArk-CARM>
```

The installer will create the **ResponseManager** safe and add objects to it:



- 6) The installer also created a user named **CARM-user** with the following Authorizations:
 - a. Audit Users.
 - b. Add/Update Users.
 - c. Reset User's Passwords.
- 7) From the PVWA to create an application named **CARM-App** and take the appropriate steps to secure it. The Application must have **Retrieve Accounts** on the **ResponseManager** safe.
- 8) Start the service:



- 9) On the Windows Firewall open the port you specified in the SyslogServerPort parameter in the previous step.

REQUESTOR

This is the server from which the orchestration software will trigger actions in CyberArk. The CyberArk AAM agent must be installed in this server

- 1) Install AAM (Contact CyberArk if you don't have a license).
- 2) Configure Credential Provider to use No Cache. You might want to use a separate configuration file for CARM (refer to the Credential Provider and ASCP Implementation Guide, so the following parameters won't affect multiple Credential Provider installations:
 - a. Set **CacheLevel=None**
 - b. Remove, or comment out, the remaining parameters in the **[Cache]** section.
 - c. Restart the AAM service.
- 3) Make sure the Credential Provider User has the following Permissions on the **ResponseManager** safe:
 - a. Retrieve Accounts.
 - b. List Accounts.
 - c. View Safe Members.
- 4) Choose an installation directory and copy the **TestActionsScripts** folder from the distribution.

POST INSTALLATION CONFIGURATION

THE CARM-CONFIG.INI

```
[GENERAL]
VAULT= Vault.ini
CREDFILE= CARM-user.ini
PORT = 11514
PROTOCOL = UDP
policySecuritySets = policySecuritySets = [
    ["Oracle", "Oracle-Level2", "Oracle-Level3"],
    ["LinuxPCIRoot", "LinuxPCIRoot-Level2"],
    ["UnixSSH", "UnixSSH-Level1", "UnixSSH-Level2", "UnixSSH-Level3"],
    ["WinServerLocal", "WinServerLocal-Level2", "WinServerLocal-Level3"]
]
safesToQuery = [""]
safesToExclude = ["PSMRecordings"]
RetentionPeriod=30
DEBUGPACLI=tru
```

VAULT	Name of the Vault.ini file
CREDFILE	Name of the CARM-user credential file
PORT	Vault Syslog port
PROTOCOL	Vault Syslog protocol
policySecuritySets	Can contain multiple arrays, each one detailing the security policy escalation chain. The members of each array are sequence of PlatformsID's each with a more restrictive policy than the one before. You can find the PlatformID's on the PVWA -> Administration -> Platform Management.
safesToQuery	It contains a list of safes to search into if safe is not provided for the query.
safesToExclude	It contains a list of safes to avoid querying
DEBUGPACLI	Enable PACLI debugging
RetentionPeriod	Maximum number of days to keep completed action objects in the Response Manager safe Temp folder. Objects older than that will be deleted. A value of -1 means no pruning will occur. A Value of zero is not allowed.

TEST THE INSTALLATION

The **TestActionsScripts** folder contains a set of Python scripts to invoke each of the actions. These scripts can be invoked from an application or the command line. There is also a **TestAction.py** script that you can use to verify the installation is complete:

```
[root@centos01 TestActionScripts]# ./TestAction.py
Success=True output=[True]
```

USE CASES DETAIL

Each of the **TestActionScripts** can be invoked with parameters on the command line. There's also a **CyberArkResponseManager.py** which is the main wrapper for the action invocation in the Vault. This file must be accessible to all the other scripts. It makes the AAM call to retrieve the Sequence ID, it assembles the Task ID by concatenating <Action Name>-<Sequence ID>, then queries the Object in the Temp folder of the **Response Manager** safe until it completes and then displays the TaskStatus (Completed or Failed), and any additional information the action returns.

ChangeCredentials.py: It marks an account for immediate password change. It takes **Account Address** and **User Name** as parameters.

DisableCyberArkUser.py: It disables a CyberArk user. It takes the **CyberArk User** login as a parameter.

EnableCyberArkUser.py: It enables a CyberArk user. It takes the **CyberArk User** login as a parameter.

RaiseSecurityPolicy.py: It moves an account up to the next higher security policy listed in one of the arrays in the **policySecuritySets** parameter described above. It takes **Account Address** and **User Name** as parameters.

LowerSecurityPolicy.py: It moves an account down to the next lower security policy listed in one of the arrays in the **policySecuritySets** parameter. It takes **Account Address** and **User Name** as parameters.

CredentialsAccessDuringPeriod.py: It takes **Account Address** and **User Name** as parameters. It returns an array showing the most recent retrievals of an account credentials :

```
[root@centos01 TestActionScripts]# ./CredentialsAccessDuringPeriod.py rhel2 logon
Success=True output=[{"REQUESTREASON": "(Action: Show Password)", "ACTIVITY": "Retrieve password", "CODE": "295", "USER": "Mike", "TIME": "Thu Mar 23 17:22:23 2017"}, {"REQUESTREASON": "(Action: Show Password)", "ACTIVITY": "Retrieve password", "CODE": "295", "USER": "Mike", "TIME": "Thu Mar 23 20:22:04 2017"}, {"REQUESTREASON": "(Action: Show Password)", "ACTIVITY": "Retrieve password", "CODE": "295", "USER": "Mike", "TIME": "Thu Mar 23 20:22:06 2017"}, {"REQUESTREASON": "(Action: Show Password)", "ACTIVITY": "Retrieve password", "CODE": "295", "USER": "Mike", "TIME": "Thu Mar 23 20:22:07 2017"}, {"REQUESTREASON": "(Action: Show Password)", "ACTIVITY": "Retrieve password", "CODE": "295", "USER": "Mike", "TIME": "Thu Mar 23 20:22:09 2017"}, {"REQUESTREASON": "(Action: Show Password)", "ACTIVITY": "Retrieve password", "CODE": "295", "USER": "Mike", "TIME": "Thu Mar 23 20:22:10 2017"}, {"REQUESTREASON": "(Action: Show Password)", "ACTIVITY": "Retrieve password", "CODE": "295", "USER": "Mike", "TIME": "Thu Mar 23 20:22:11 2017"}]]
[root@centos01 TestActionScripts]#
```

By default a period of the last 4 hours is assumed but can be override by providing the custom number of hours.

WRITING YOUR OWN CALLS

If you prefer not to use the provided Python scripts, you can write your calls in any language supported by the **AAM SDK**. Each of the actions has a matching object name in the **ResponseManager** safe (see step #6 of CARM Server install). You'll need to retrieve the **SequenceID** from that object with the parameters for the action in the **reason** method of the **PSDKPasswordRequest()** object encoded using the Base64 algorithm. Then you'll need to retrieve the **Action-SequenceID** object from the **Temp** folder in the **Response Manager** safe to determine Success or Failure. The **reason** must be formatted as a JSON object with only one header named **data** and no carriage returns:

```
{'data': {"Parameter" : Value, "Parameter" : Value}}
```

For the **EnableCyberArkUser** and **DisableCyberArkUser** actions the JSON will look like this:

```
{'data': {"cyberark.username" : Value}}
```

For all other actions it will look like this:

```
{'data': {"Address" : Value, "UserName" : Value}}
```

You must use the **CARM-App** application (that you created on step #5 of the CARM Server install) as the App ID for the AAM calls.

TROUBLESHOOTING

The **logs** folder contains a **CARM-Console** file that will provide an indication of the actions being performed . If you don't see any traffic from the Vault, either the Syslog properties are misconfigured or the Windows Firewall is blocking traffic on the Syslog port you chose.

Make sure the Requestor, Vault, and CARM servers are time synchronized.

