****

Onboarding Automation

*September 2019*

Ideally all onboarding processes are automated, however due to limited API support from Beyond Trust, PAM onboarding is a combination of manual process and scripting. Key API enhancements are submitted to BT.

1. **Execution Environment**

The onboarding scripts are executed on the monitoring servers, where firewall and IP whitelist are setup for monitoring API as well.

The account to execute the script is managed by PAM.

The API key are encrypted and stored in configuration file:

*PasswordSafeApiKey={AES}RuzqxLXHh4/jwNRbCU8VVQ==,a32Nye2XbgO1AWCKHpirws/hfyOWteti6wDaB8IqkzIIrPv1h0tArR7BlxWfInRQmAOXhFJ4AEaXEmwQF+bSv68GCjK4Hi6buIbv9qhE0/mgMggurdVoMsLeCAxV2V1Nuf2vMSsoBhKNrWdpbKTh79JFPXqT09iODx+uqjxd8aaDjIXhGt53rXC70X+ZUSbG*

1. **Commands**

To execute the onboarding script, run the psrun.sh command, it takes one or more argument:

The first argument is the task name, e.g. app\_named\_acct, app\_named\_acct\_cleanup, default\_acct, …

Addition arguments are for specific task, e.g. for app\_named\_acct task (onboard application named account), it takes an excel file name as the input.

E.g.

./psrun.sh app\_named\_acct acct.xlsx

In general, for each onboard script, there will be a corresponding cleanup script to rollback.

E.g.

./psrun.sh app\_named\_acct\_cleanup acct.xlsx

1. **Task List**
2. Default Account

Default account is managed by smart rule, the smart rule uses asset smart rule for platform and account name as filter; the onboarding script only tags the asset to correct platform value.

Three tasks are setup in psrun.sh: 'tag\_windows', 'tag\_linux' and 'tag\_aix'

Each task take a smart rule name as input, e.g.

./psrun.sh tag\_linux tmp\_address\_group\_for\_linux

The script retrieves all assets covered by the input smart rule, and set the platform attribute value to “Windows”, “Linux”, or “AIX”.

1. Application Named Account

The application and account data are provided in excel file with the following format:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Application | Account | PeopleSoft No. | Comment | Type of Account |  |  |  |
| AMOS | johe.smith | 7001234 | admin account | Named Account |  |  |  |
| … | … | … | … |  |  |  |  |

The “Type of Account” is a filter, other types of account also in the same excel file.

For each application in the excel:

* The script creates a virtual asset with name = {Application} \_DummyAsset, e.g. AMOS\_DummyAsset, IP = 1.1.10.1.
* Tags the asset with Application={Application}\_A, e.g. AMOS\_A
* Onboard the asset as a managed system with type = “Generic Platform”.

For each account in the excel:

* The script creates managed account on the corresponding managed system for the application with description: application={Application} owner={Name} comment={Comment}, e.g.

*application=AMOS*

*owner=7001234*

*Comment=admin account*

* The script also set the following attribute values on the account:

Application={Application}

AccountType=App\_Named\_Vault

Command to onboard:

*./psrun.sh app\_named\_acct\_vault app\_ acct.xlsx*

Command to rollback:

*./psrun.sh app\_named\_acct\_vault\_cleanup acct.xlsx*

1. Application Shared, Service, Vendor Account

The application and account data are provided in excel file with the following format:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Application | Account | Name | Comment | Type of Account |  |  |  |
| AMOS | sadmin |  | admin account | Shared Account  (or Service/Vendor Account) |  |  |  |
| … | … | … | … |  |  |  |  |

The “Type of Account” is a filter, other types of account also in the same excel file.

For each application in the excel:

* The script creates a virtual asset with name = {Application} \_DummyAsset, e.g. AMOS\_DummyAsset, IP = 1.1.10.1.
* Tags the asset with Application={Application}\_A, e.g. AMOS\_A
* Onboard the asset as a managed system with type = “Generic Platform”.

For each account in the excel:

* The script creates managed account on the corresponding managed system for the application with description: application={Application} owner={Name} comment={Comment}, e.g.

*application=AMOS*

*owner=*

*Comment=admin account*

* The script also set the following attribute values on the account:

Application={Application}

AccountType=App\_Service\_Vault (Or App\_Shared\_Vault/App\_Vendor\_Vault)

Command to onboard:

*./psrun.sh app\_shared\_vault acct.xlsx*

*./psrun.sh app\_service\_vault acct.xlsx*

*./psrun.sh app\_vendor\_vault acct.xlsx*

Command to rollback:

*./psrun.sh app\_shared\_vault\_cleanup acct.xlsx*

*./psrun.sh app\_service\_vault\_cleanup acct.xlsx*

*./psrun.sh app\_vendor\_vault\_cleanup acct.xlsx*

1. Platform Named Account

Not managed.

1. Platform Service Account/Shared Account

The platform, application and account data are provided in excel file with the following format:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Asset | Application | Account | Password | Owner | Comment | Type of Account |  |  |  |
| 10.21.12.121  10.21.12.122 | AMOS | sadmin |  |  | admin account | Shared Account  (or Service) |  |  |  |
| All Windows | … | … |  | … | … |  |  |  |  |

For each row in the excel:

1. If asset explicitly enumerated, then validate it is valid and managed already. If all Windows, all Linux, then use all asset in the corresponding smart rule: PAM\_Windows, PAM\_Linux, ...

2. Verify the asset is a managed system.

3. Verify the asset if not used by other application and tag the application to asset.

4. Create the managed account, if the password exist in excel, use the password, otherwise set a default one. The account’s description is set to: application={Application} owner={Owner} comment={Comment}

5. Tag the account with Application (= {Application}, if exist) and AccountType (=App\_Service\_Vault or App\_Shared\_Vault ) attribute.

The cleanup remove the account of the specified type listed in the excel.

The script takes two argument, account type and excel.

Command to onboard:

*./psrun.sh platform\_shared\_vault acct.xlsx*

*./psrun.sh app\_service\_vault acct.xlsx*

Command to rollback:

*./psrun.sh platform\_shared\_vault\_cleanup acct.xlsx*

*./psrun.sh platform\_service\_vault\_cleanup acct.xlsx*

1. Mash Account

Run this script to manage the DSS key used by Linux team to patching servers.

The script take two parameters:

1. An excel file name as parameter, the excel file is generated by Puppet.
2. The parent account server IP

|  |  |
| --- | --- |
| Hostname | ip\_address |
| rhldafsc042.na.rccl.com | 10.14.31.39 |
| … | … |

* The script validates all puppet managed servers exist in PS. (Note, this step is performed early, all missing asset issue should have been resolved before final run)
* Use the mshdaemon account on the specified parent server (passed in as 2nd parameter) as the parent account, for other asset, create the managed account and add it as a synced account to the parent account.

Command to onboard:

*./psrun.sh mash ~/prod/mashlist.xls 10.14.31.39*

Command to rollback (this only remove the managed account, will see whether necessary to remove asset):

*./psrun.sh mash\_cleanup ~/prod/mashlist.xls 10.14.31.39*

1. Other Utilities

**Set asset attribute**: this utilities read all asset in the smart rule, wipe out all existing values for the specified attribute name, and set the new value for the attribute.

./psrun.sh set\_attribute\_for\_assets {smart rule name} {attribute\_name} {attribute\_value}

**Append asset attribute**: this utilities read all asset in the smart rule, append the new value for the attribute, and skip it if attribute value exists already.

./psrun.sh append\_attribute\_for\_assets {smart rule name} {attribute\_name} {attribute\_value}