Accelerate 2024 - Automation Workshop

Ansible Automation with FlashArray

This Workshop material is a remix of Simon Dodsley's excellent Ansible FlashArray Workshop. The original repository can be found here

The Workshop

This content is a toolkit for demonstrating Ansible's capabilities on the Pure Storage FlashArray by providing hands-on or self-paced training.

The Workshop Lab is pre-configured and ready to go as-is in **Pure's Test Drive** environment. If you are running this workshop in your own environment, you will need to configure the FlashArray and Ansible environment as per the instructions in the original repository.

To start the Lab:

- On the Windows host, click on the Chrome icon on the desktop and click on the "flasharray1" bookmark to open the FlashArray GUI.
- Login to the FlashArray using the credentials provided in the Credentials tab.
- Minimize the browser window and open the terminal by clicking on the "Putty" icon on the desktop.
- Select the "linux1" host and click "Open" to open the terminal.
- Login with the credentials provided in the Credentials tab.
- Change directories to the **ansible-workshop/ansible-flasharray/1.0-get-facts** directory.
- Follow the guide. The internal Guide may be difficult to follow with the limited screen space, so it is also available on the desktop and in the GitHub repository as a PDF file.
- Click on this link to open the Guide from the GitHub Repository: Guide

Section 1 - Ansible FlashArray Basic (single FA)

- Exercise 1.0 - Using the purefa_info module
- Exercise 1.1 - Adding a host to a FlashArray
- Exercise 1.2 - Creating volumes on a FlashArray
- Exercise 1.3 - Connecting volumes to a host on a FlashArray
- Exercise 1.4 - Creating a protection group on a FlashArray

Section 2 - Ansible FlashArray Advanced (requires 2 FAs)

- Exercise 2.0 - Connecting two FlashArrays for replication
- Exercise 2.1 - Configuring FlashArray Networking
- Exercise 2.2 - Configure ActiveCluster pods

Exercise 1.0 - Using the purefa_info module

Objective

Demonstrate the use of the purefa_info module to get facts (usefule information) from a Pure Storage FlashArray and display them to a terminal window using the debug module.

Guide

Use the Linux terminal in Test Drive, logging in with the supplied credentials.

Step 1:

Run the playbook - Execute the following:

```
$ ansible-playbook purefa-info.yml
```

The output will look similar to this.

```
$ ansible-navigator run purefa-info.yml --mode stdout


PLAY [GRAB FLASHARRAY FACTS] *************************************************************


TASK [COLLECT FLASHARRAY FACTS] **********************************************************

ok: [localhost]


TASK [DISPLAY COMPLETE FLASHARRAY MINIMUM INFORMATION] **********************************

ok: [localhost] => {

  "array_facts": {

    "changed": false,

    "failed": false,

    "purefa_info": {

      "default": {

        "admins": 9,

        "api_versions": [

          "1.0",

          "1.1",
    "1.2",
    "1.3",
    "1.4",
    "1.5",
    "1.6",
    "1.7",
    "1.8",
    "1.9",
    "1.10",
    "1.11",
    "1.12",
    "1.13",
    "1.14",
    "1.15",
    "1.16",
    "1.17",
    "1.18",
    "2.0",
    "2.1"
  ],
  "array_model": "FA-405",
  "array_name": "acme-array-1",
  "connected_arrays": 0,
  "connection_key": "e448c603-ecfd-8b4e-fc02-0d742e81a779",
  "hostgroups": 4,
  "hosts": 10,
  "maintenance_window": [],
  "pods": 0,
  "protection_groups": 10,
  "purity_version": "5.3.17",
  "remote_assist": "disabled",
  "snapshots": 25,
            "volume_groups": 1,
            "volumes": 38
          }
        }
      }
    }

PLAY RECAP
****************************************************************************************************
localhost              : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Step 2:

Finally let's append two more tasks to get more specific info from facts gathered, to the above playbook.

```
- name: DISPLAY ONLY THE FLASHARRAY MODEL
  ansible.builtin.debug:
    var: array_facts['purefa_info']['default']['array_model']


- name: DISPLAY ONLY THE PURITY VERSION
  ansible.builtin.debug:
    var: array_facts['purefa_info']['default']['purity_version']
```

- var: array_facts['purefa_info']['default']['array_model'] displays the model name for the FlashArray
- array_facts['purefa_info']['default']['purity_version'] displays the Purity version for the FlashArray

Because the purefa_info module returns useful information in structured data, it is really easy to grab specific information without using regex or filters. Fact modules are very powerful tools to grab specific device information that can be used in subsequent tasks, or even used to create dynamic documentation (reports, csv files, markdown).

Step 3:

Run the playbook - Save the file and execute the following:

```
$ ansible-playbook purefa-info.yml
```

Playbook Output

The output will look as follows.

```
$ ansible-playbook purefa-info.yml


PLAY [GRAB FLASHARRAY FACTS]
**********************************************************************************
```

```
TASK [COLLECT FLASHARRAY FACTS]
*********************************************************************************

ok: [localhost]


TASK [DISPLAY COMPLETE FLASHARRAY MINIMUM INFORMATION]
***************************************************************

ok: [localhost] => {

    "array_facts": {

        "changed": false,

        "failed": false,

        "purefa_info": {

            "default": {

                "admins": 9,

                "api_versions": [

                    "1.0",

                    "1.1",

                    "1.2",

                    "1.3",

                    "1.4",

                    "1.5",

                    "1.6",

                    "1.7",

                    "1.8",

                    "1.9",

                    "1.10",

                    "1.11",

                    "1.12",

                    "1.13",

                    "1.14",

                    "1.15",

                    "1.16",

                    "1.17",
                    "1.18",
                    "2.0",
                    "2.1"
                ],
                "array_model": "FA-405",
                "array_name": "sn1-405-c07-27",
                "connected_arrays": 0,
                "connection_key": "e448c603-ecfd-8b4e-fc02-0d742e81a779",
                "hostgroups": 4,
                "hosts": 10,
                "maintenance_window": [],
                "pods": 0,
                "protection_groups": 10,
                "purity_version": "5.3.17",
                "remote_assist": "disabled",
                "snapshots": 25,
                "volume_groups": 1,
                "volumes": 38
            }
        }
    }
}

TASK [DISPLAY ONLY THE FLASHARRAY MODEL] ***********************************************************************

ok: [localhost] => {
    "array_facts['purefa_info']['default']['array_model']": "FA-405"
}

TASK [DISPLAY ONLY THE PURITY VERSION] **************************************************************************

ok: [localhost] => {
    "array_facts['purefa_info']['default']['purity_version']": "5.3.17"
```

```
}


PLAY RECAP
*****************************************************************************************************

localhost              : ok=4    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Solution

The finished Ansible Playbook is provided here: purefa-info.yml.

Going Further

For this bonus exercise add the tags: debug parameter (at the task level) to the existing debug task.

```
- name: DISPLAY COMPLETE FLASHARRAY MINIMUM INFORMATION

  ansible.builtin.debug:

    var: array_facts

  tags: debug
```

Now re-run the playbook with the --skip-tags debug command line option.

```
ansible-playbook purefa-info.yml --skip-tags debug
```

Ansible will only run three tasks, skipping the DISPLAY COMPLETE FLASHARRAY MINIMUM INFORMATION task.

Exercise 1.1 - Adding hosts to a FlashArray

Objective

Demonstrate the use of the purefa_host module to add a host to a Pure Storage FlashArray.

Step 1:

Run the playbook - Execute the following:

```
$ ansible-playbook purefa-host.yml
```

Playbook Output

The output will look something like this.

```
$ ansible-playbook purefa-host.yml


PLAY [HOST SETUP]
****************************************************************************************************


TASK [Gathering Facts]
*********************************************************************************************

ok: [localhost]
```

```
TASK [CREATE HOST]
************************************************************************************************

changed: [localhost]


PLAY RECAP
****************************************************************************************************
***

localhost          : ok=2   changed=1   unreachable=0   failed=0   skipped=0   rescued=0   ignored=0
```

Verifying the Solution

On the Windows host, open a browser and navigate to the Pure Storage FlashArray management interface by clicking on the "flasharray1" shortcut. Login with the credentials provided in the Credentials Tab of the Guide. Go to Storage --> Hosts and verify that the host "ansiblehost" has been added.

Going Further

Feel free to edit the purefa-host.yml file and replace the variable ansible_hostname: ansible_hostname with a different hostname to add a different host to the FlashArray.

Exercise 1.2 - Adding volumes to a FlashArray

Objective

Demonstrate the use of the purefa_volume module to add a host to a Pure Storage FlashArray.

Guide

Step 1:

Using the text editor, create a new file called purefa-volume.yml.

Step 2:

Enter the following play definition into purefa-volume.yml:

```
---

- name: VOLUME SETUP

  hosts: localhost

  connection: local

  gather_facts: true

  vars:

    url: flasharray1.testdrive.local

    api: e448c603-ecfd-8b4e-fc02-0d742e81a779
```

- The --- at the top of the file indicates that this is a YAML file.
- The hosts: localhost, indicates the play is run on the current host.

- connection: local tells the Playbook to run locally (rather than SSHing to itself)
- gather_facts: true enables facts gathering.
- The vars: parameter is a group of parameters to be used in the playbook.
- url: flasharray1.testdrive.local is the management IP address of your FlashArray - change this reflect your local environment.
- api: e448c603-ecfd-8b4e-fc02-0d742e81a779 is the API token for a user on the FlashArra - change this reflect your local environment.

Step 3:

Next, add the first task to the playbook. This task will use the purefa_volume module to create three volume object on the Pure Storage FlashArray.

```
tasks:
  - name: CREATE VOLUMES
    purestorage.flasharray.purefa_volume:
      name: volume_{{ item }}
      size: 32G
      fa_url: "{{ url }}"
      api_token: "{{ api }}"
    loop:
      - a
      - b
      - c
```

- name: CREATE VOLUMES is a user defined description that will display in the terminal output.
- purefa_volume: tells the task which module to use.
- The name parameter tells the module the name of the volume to create. The {{ item }} varable instructs Ansible with the loop value.
- The size parameter tells the module the size to create the volume. The size identifier can be in b, k, M, G, T or P.
- The fa_url: "{{url}}" parameter tells the module to connect to the FlashArray Management IP address, which is stored as a variable url defined in the vars section of the playbook.
- The api_token: "{{api}}" parameter tells the module to connect to the FlashArray using this API token, which is stored as a variable api defined in the vars section of the playbook.
- loop: tells the task to loop over the provided list. The list in this case is the suffix to be appended to the volume name.

Save the file and exit out of the editor.

Step 4:

Run the playbook - Execute the following:

```
$ ansible-playbook purefa-volume.yml
```

Playbook Output

The output will look something like this.

```
$ ansible-playbook purefa-volume.yml


PLAY [VOLUME SETUP]
*********************************************************************************************************



TASK [Gathering Facts]
******************************************************************************************************

ok: [localhost]


TASK [CREATE VOLUMES]
*******************************************************************************************************

changed: [localhost] => (item=a)

changed: [localhost] => (item=b)

changed: [localhost] => (item=c)


PLAY RECAP
***************************************************************************************************************
***

localhost             : ok=2   changed=1   unreachable=0   failed=0   skipped=0   rescued=0   ignored=0
```

Verifying the Solution

Login to the Pure Storage FlashArray with your web browser.

The list of volumes configured in the array can be found by navigating the menu on the left to Storage, then selecting the top menu item Volumes.

Going Further

Feel free to edit the .yml file and replace any of the variables to create different volumes on the FlashArray.

Exercise 1.3 - Connecting volumes to a host on a FlashArray

Objective

Demonstrate the use of the purefa_host module to connect an existing volume to a host on a Pure Storage FlashArray.

Guide

Step 1:

Using the text editor, create a new file called purefa-connect.yml.

Step 2:

Enter the following play definition into purefa-connect.yml:

```
---
- name: CONNECT SETUP
  hosts: localhost
  connection: local
  gather_facts: true
  vars:
    url: flasharray1.testdrive.local
    api: e448c603-ecfd-8b4e-fc02-0d742e81a779
    vol: volume_a
```

- The --- at the top of the file indicates that this is a YAML file.
- The hosts: localhost, indicates the play is run on the current host.
- connection: local tells the Playbook to run locally (rather than SSHing to itself)
- gather_facts: true enables facts gathering.
- The vars: parameter is a group of parameters to be used in the playbook.
- url: flasharray1.testdrive.local is the management IP address of your FlashArray - change this reflect your local environment.
- api: e448c603-ecfd-8b4e-fc02-0d742e81a779 is the API token for a user on the FlashArra - change this reflect your local environment.

Step 3:

Next, add the first task to the playbook. This task will use the purefa_host module to attach one of the volumes created in Exercise 1.2 to the host created in Exercise 1.1 on the Pure Storage FlashArray.

```
tasks:
  - name: ATTACH VOLUMES
    purestorage.flasharray.purefa_host:
      name: "{{ ansible_hostname }}"
      volume: "{{ vol }}"
      fa_url: "{{ url }}"
      api_token: "{{ api }}"
```

- name: ATTACH VOLUMES is a user defined description that will display in the terminal output.

- purefa_host: tells the task which module to use.
- The name parameter tells the module to use the ansible_hostname variable obtained from the host fact gathering.
- The volume parameter tells the module ito use the vol variable as the name of the volume to connect to the host object.
- The fa_url: "{{url}}" parameter tells the module to connect to the FlashArray Management IP address, which is stored as a variable url defined in the vars section of the playbook.
- The api_token: "{{api}}" parameter tells the module to connect to the FlashArray using this API token, which is stored as a variable api defined in the vars section of the playbook.

Save the file and exit out of the editor.

Step 4:

Run the playbook - Execute the following:

```
$ ansible-playbook purefa-connect.yml
```

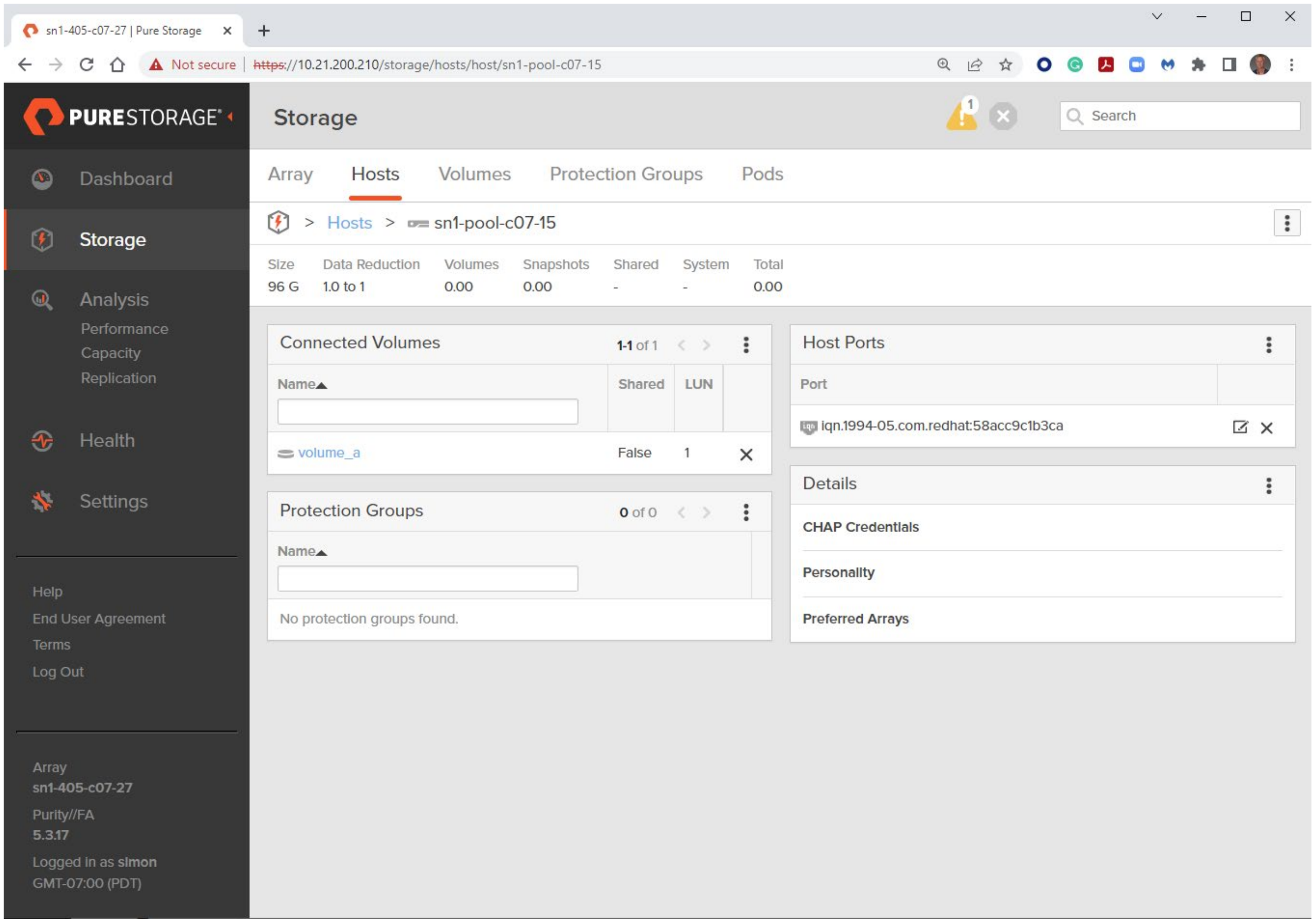
Playbook Output

```
$ ansible-playbook purefa-volume.yml


PLAY [CONNECT SETUP]
****************************************************************************************************


TASK [Gathering Facts]
****************************************************************************************************

ok: [localhost]


TASK [CONNECT VOLUMES]
****************************************************************************************************

changed: [localhost]


PLAY RECAP
****************************************************************************************************
***

localhost            : ok=2   changed=1   unreachable=0   failed=0   skipped=0   rescued=0   ignored=0
```

Verifying the Solution

Login to the Pure Storage FlashArray with your web browser.

Navigating using the menu on the left to Storage, then select Hosts top menu, finally select the host object you have created in the Hosts sub-window. This will show the volume connected to the host and the LUN ID assigned to the volume.



Size	Data Reduction	Volumes	Snapshots	Shared	System	Total
96 G	1.0 to 1	0.00	0.00	-	-	0.00

Connected Volumes 1-1 of 1

Name▲	Shared	LUN
volume_a	False	1

Protection Groups 0 of 0

Name▲

No protection groups found.

Host Ports

Port
iqn.1994-05.com.redhat:58acc9c1b3ca

Details

CHAP Credentials

Personality

Preferred Arrays

Array
sn1-405-c07-27
Purity//FA
5.3.17
Logged in as simon
GMT-07:00 (PDT)

Going Further

For a bonus exercise you can rescan the iSCSI dataplane on your host to discover the newly connected volume, format and mount the volume, making it ready for application or user use.

Add the following tasks to the playbook:

```
  - name: GET VOLUME SERIAL NUMBER

    purestorage.flasharray.purefa_info:

      gather_subset: volumes

      fa_url: "{{ url }}"

      api_token: "{{ api }}"

    register: volumes_data



  - set_fact:

      volume_serial: "{{ volumes_data.purefa_info.volumes[vol].serial }}"
  - name: RESCAN MULTIPATH
    ansible.builtin.command: /usr/sbin/multipath -r
    ansible.builtin.command: /usr/bin/scsi-rescan

  - name: GET DEVICE ID FOR VOLUME
    ansible.builtin.shell:
      cmd: /usr/sbin/multipath -ll |grep -i {{ volume_serial }}| awk '{print $2}'
    register: mpath_dev

  - name: FORMAT VOLUME
    community.general.filesystem:
      fstype: ext4
      dev: '/dev/{{ mpath_dev.stdout }}'

  - name: MOUNT VOLUME
    ansible.posix.mount:
      path: "/workshop-mount"
      fstype: ext4
      src: '/dev/{{ mpath_dev.stdout }}'
      state: mounted
```

Run the playbook - Execute the following:

```
$ ansible-playbook purefa-connect.yml
```

and then execute the command mount /workshop-mount.

Exercise 1.4 - Creating a Protection Group on a FlashArray

Objective

Demonstrate the use of the purefa_pg module to create a protection group on a Pure Storage FlashArray and assign existing volumes to it.

Guide

- The --- at the top of the file indicates that this is a YAML file.
- The hosts: localhost, indicates the play is run on the current host.
- connection: local tells the Playbook to run locally (rather than SSHing to itself)

- gather_facts: true enables facts gathering.
- The vars: parameter is a group of parameters to be used in the playbook.
- url: flasharray1.testdrive.local is the management IP address of your FlashArray - change this reflect your local environment.
- api: e448c603-ecfd-8b4e-fc02-0d742e81a779 is the API token for a user on the FlashArra - change this reflect your local environment.

Step 1:

Run the playbook - Execute the following:

```
$ ansible-playbook purefa-pgroup.yml
```

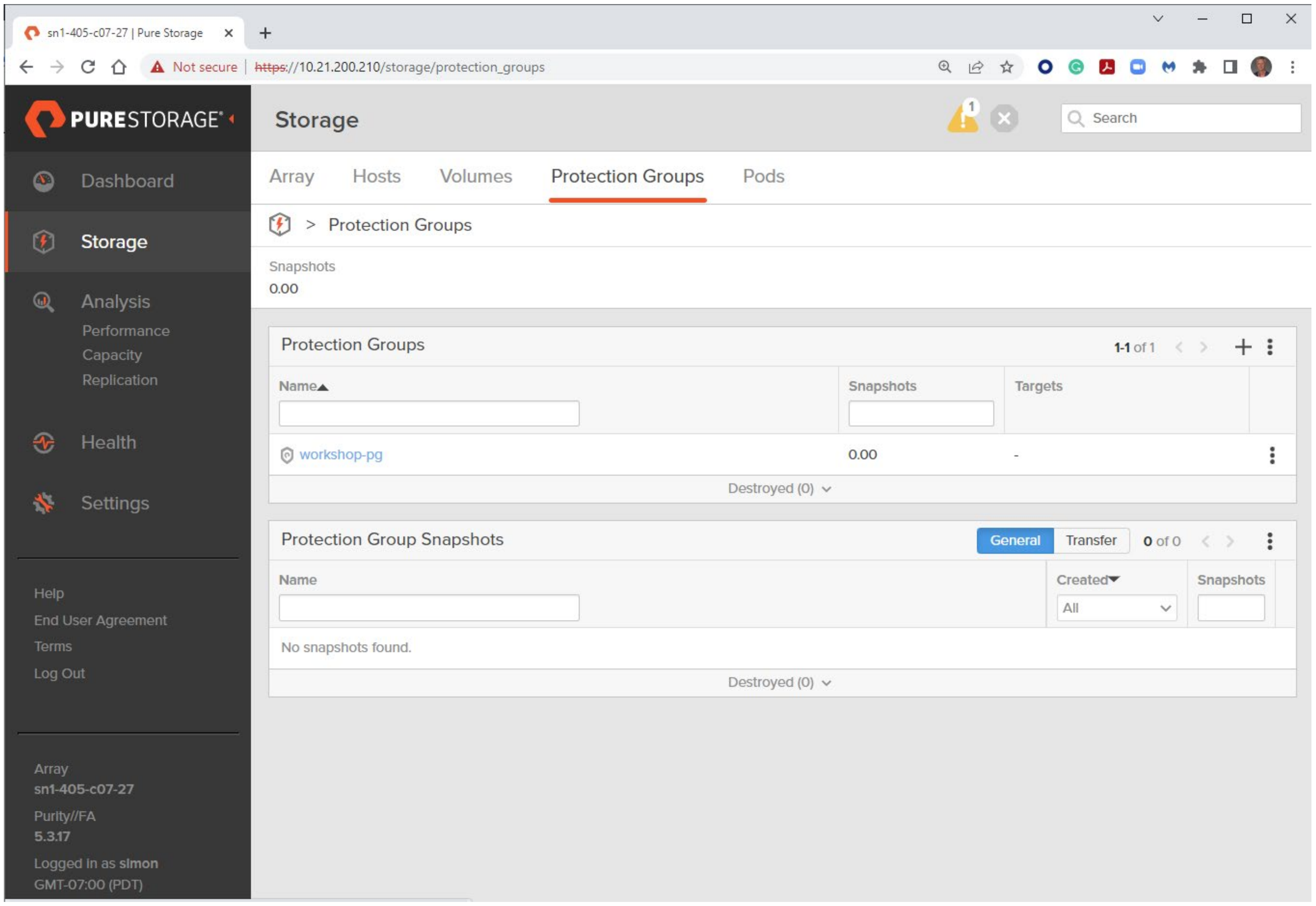
Playbook Output

```
$ ansible-playbook purefa-pgroup.yml


PLAY [PROTECTION GROUP SETUP]
******************************************************************************************


TASK [Gathering Facts]
*************************************************************************************************

ok: [localhost]


TASK [CREATE PROTECTION GROUP]
*****************************************************************************************

changed: [localhost]


PLAY RECAP
**********************************************************************************************************
***

localhost          : ok=2   changed=1   unreachable=0   failed=0   skipped=0   rescued=0   ignored=0
```

Verifying the Solution

Login to the Pure Storage FlashArray with your web browser.

The list of protection groups configured in the array can be found by navigating the menu on the left to Storage, then selecting the top menu item Protection Groups.



Snapshots
0.00

Protection Groups 1-1 of 1

Name▲	Snapshots	Targets
workshop-pg	0.00	-

Destroyed (0)

Protection Group Snapshots (General | Transfer) 0 of 0

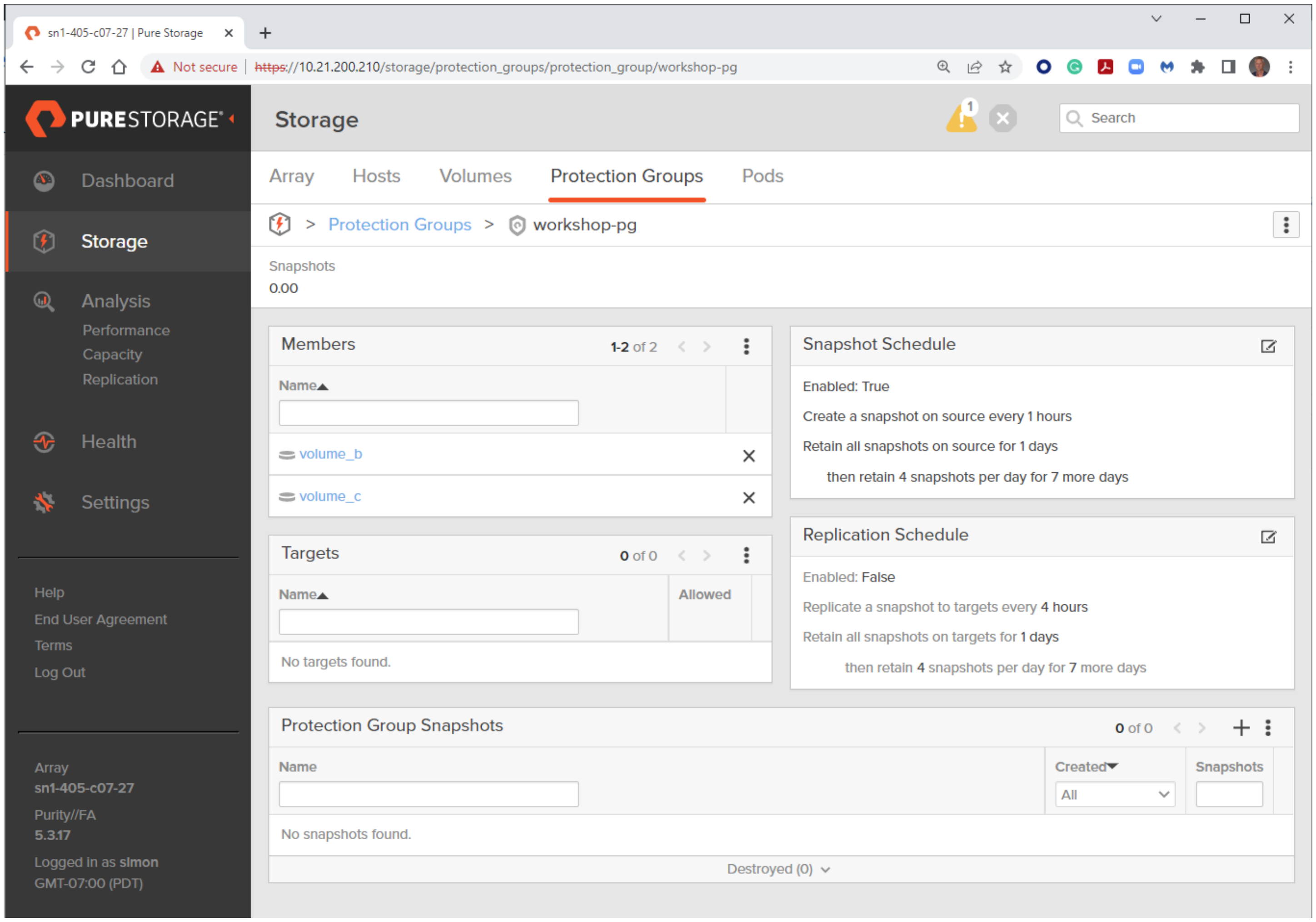
Name	Created▼	Snapshots
	All	

No snapshots found.

Destroyed (0)

Array
sn1-405-c07-27
Purity//FA
5.3.17
Logged in as simon
GMT-07:00 (PDT)

Selecting the created protection group will give more detail on the contents of the group,



Storage > Protection Groups > workshop-pg

Snapshots
0.00

Members 1-2 of 2

Name▲
volume_b
volume_c

Snapshot Schedule

Enabled: True
Create a snapshot on source every 1 hours
Retain all snapshots on source for 1 days
then retain 4 snapshots per day for 7 more days

Targets 0 of 0

Name▲	Allowed

No targets found.

Replication Schedule

Enabled: False
Replicate a snapshot to targets every 4 hours
Retain all snapshots on targets for 1 days
then retain 4 snapshots per day for 7 more days

Protection Group Snapshots 0 of 0

Name	Created▼	Snapshots
	All	

No snapshots found.

Destroyed (0)

Array sn1-405-c07-27
Purity//FA 5.3.17
Logged in as simon
GMT-07:00 (PDT)

Exercise 2.0 - Connecting two FlashArrays for replication

Objective

Demonstrate the use of the purefa_connect module to connect two Pure Storage FlashArrays for replication, either asynchronous or synchronous.

Guide

- The --- at the top of the file indicates that this is a YAML file.
- The hosts: localhost, indicates the play is run on the current host.
- connection: local tells the Playbook to run locally (rather than SSHing to itself)
- gather_facts: true enables facts gathering.
- The vars: parameter is a group of parameters to be used in the playbook.
- url: flasharray1.testdrive.local is the management IP address of your source FlashArray - change this reflect your local environment.
- api: e448c603-ecfd-8b4e-fc02-0d742e81a779 is the API token for a user on the source FlashArray - change this reflect your local environment.

- target_url: flasharray2.testdrive.local is the management IP address of your target FlashArray - change this reflect your local environment.
- target_api: 24a96e35-e0e2-806e-c0cc-eaf45e7fa887 is the API token for a user on the target FlashArray - change this reflect your local environment.
- name: CREATE ASYNC|SYNC CONNECTION is a user defined description that will display in the terminal output.
- purefa_connect: tells the task which module to use.
- The connection parameter tells the module the type of replication connection to create. Choices are async or sync.
- The fa_url: "{{url}}" parameter tells the module to connect to the FlashArray Management IP address, which is stored as a variable url defined in the vars section of the playbook. This makes this array the source array in the replication pair.
- The api_token: "{{api}}" parameter tells the module to connect to the FlashArray using this API token, which is stored as a variable api defined in the vars section of the playbook.
- The target_url: "{{target_url}}" parameter tells the module the management IP address of the target FlashArray, which is stored as a variable target_url defined in the vars section of the playbook.
- The target_api: "{{target_api}}" parameter tells the module the API token to use for the target FlashArray, which is stored as a variable target_api defined in the vars section of the playbook.

Step 1:

Run the playbook - Execute the following:

```
$ ansible-playbook purefa-connect.yml
```

Playbook Output

```
$ ansible-playbook purefa-connect.yml


PLAY [REPLICATION CONNECTION]
*********************************************************************************************



TASK [Gathering Facts]
****************************************************************************************************

ok: [localhost]


TASK [CREATE ASYNC CONNECTION]
********************************************************************************************

changed: [localhost]


TASK [CREATE SYNC CONNECTION]
*********************************************************************************************
```

```
changed: [localhost]


PLAY RECAP
*****************************************************************************************************
***

localhost              : ok=3   changed=2   unreachable=0   failed=0   skipped=0   rescued=0   ignored=0
```

Verifying the Solution

Login to the source (flasharray1) Pure Storage FlashArray with your web browser.

Navigate to the Protection -> Array window to see the array connections that have been created from the source side of the connections.

Login to the target (flasharray2) Pure Storage FlashArray with your web browser.

Navigate to the Protection -> Array window to see the array connections that have been created from the target side of the connections.

Exercise 2.1 - Configuring FlashArray Networking

Objective

Demonstrate the use of the purefa_network module to configure netowrk interfaces on a Pure Storage FlashArray.

Guide

- The --- at the top of the file indicates that this is a YAML file.
- The hosts: localhost, indicates the play is run on the current host.
- connection: local tells the Playbook to run locally (rather than SSHing to itself)
- gather_facts: true enables facts gathering.
- The vars: parameter is a group of parameters to be used in the playbook.
- url: flasharray1.testdrive.local is the management IP address of your source FlashArray - change this reflect your local environment.
- api: e448c603-ecfd-8b4e-fc02-0d742e81a779 is the API token for a user on the source FlashArray - change this reflect your local environment.
- name: SETUP ETHERNET INTERFACE is a user defined description that will display in the terminal output.
- purefa_network: tells the task which module to use.
- The name parameter tells the module which interface on which controller to configure the ethernet information on.
- The gateway parameter is the IP address of gateway for the ethernet subnet.
- The address parameter is the IP address to assign to the interface in CIDR notation.
- The mtu parameter is the MTU value to use for the interface.

- The fa_url: "{{url}}" parameter tells the module to connect to the FlashArray Management IP address, which is stored as a variable url defined in the vars section of the playbook. This makes this array the source array in the replication pair.
- The api_token: "{{api}}" parameter tells the module to connect to the FlashArray using this API token, which is stored as a variable api defined in the vars section of the playbook.

Step 1:

Run the playbook - Execute the following:

```
$ ansible-playbook purefa-network.yml
```

Playbook Output

```
$ ansible-playbook purefa-network.yml


PLAY [NETWORKING]
**¡*****************************************************************************************************************


TASK [Gathering Facts]
***************************************************************************************************************

ok: [localhost]


TASK [SETUP ETHERNET INTERFACE]
*****************************************************************************************************

changed: [localhost]


PLAY RECAP
**************************************************************************************************************************
***

localhost            : ok=2   changed=1   unreachable=0   failed=0   skipped=0   rescued=0   ignored=0
```

Verifying the Solution

Login to the source (flasharray1) Pure Storage FlashArray with your web browser.

Navigate to the Settings -> Network window and look in the Ethernet sub-window for the interface you configured..

Exercise 2.2 - Configure ActiveCluster pods

Objective

Demonstrate the use of the purefa_pod module to create an ActiveCluster replication pod and then stretch that pod across two connected FlashArrays.

Guide

- The --- at the top of the file indicates that this is a YAML file.
- The hosts: localhost, indicates the play is run on the current host.
- connection: local tells the Playbook to run locally (rather than SSHing to itself)
- gather_facts: true enables facts gathering.
- The vars: parameter is a group of parameters to be used in the playbook.
- url: flasharray1.testdrive.local is the management IP address of your source FlashArray - change this reflect your local environment.
- api: e448c603-ecfd-8b4e-fc02-0d742e81a779 is the API token for a user on the source FlashArray - change this reflect your local environment.
- name: CREATE POD | STRETCH POD is a user defined description that will display in the terminal output.
- purefa_pod: tells the task which module to use.
- The name parameter tells the module the name of the pod to either create or work with if the pod already exists..
- The stretch parameter is the name of the connected target array that the pod is to be stretched to. This name must exactly match the target name shown in the source arrays Array Connections list.
- The fa_url: "{{url}}" parameter tells the module to connect to the FlashArray Management IP address, which is stored as a variable url defined in the vars section of the playbook. This makes this array the source array in the replication pair.
- The api_token: "{{api}}" parameter tells the module to connect to the FlashArray using this API token, which is stored as a variable api defined in the vars section of the playbook.

Step 1:

Run the playbook - Execute the following:

```
$ ansible-playbook purefa-pod.yml
```

Playbook Output

```
$ ansible-playbook purefa-pod.yml


PLAY [ACTIVECLUSTER POD]
***********************************************************************************************


TASK [Gathering Facts]
***********************************************************************************************

ok: [localhost]


TASK [CREATE POD]
***********************************************************************************************
```

```
changed: [localhost]


TASK [STRETCH POD]
**************************************************************************************************

changed: [localhost]


PLAY RECAP
*******************************************************************************************************
***

localhost              : ok=3    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Verifying the Solution

Login to the source (flasharray1) Pure Storage FlashArray with your web browser.

Navigate to the Storage -> Pods window to see the array connections that have been created from the source side of the connections.

Notice that the Array filed contains both the source and target array names indicating the pod has successfully stretched.

Going Further

In the pod stretch task you were required to know the exact name of the target array to which the pod was being stretched.

To assist in making this process more automated you can perform a purefa_info task on the target array and use the response from this to find the array name automatically and use this in the stretch task.

This bonus exercise allows you to perform the pod stretch in a fully automated fashion.

Add the following information into the vars section of the playbook:

```
  target_url: flasharray2.testdrive.local

  target_api: 24a96e35-e0e2-806e-c0cc-eaf45e7fa887
```

Replace the second task in your YAML file, the pod stretch task, with the following:

```
  - name: TARGET INFO

    purestorage.flasharray.purefa_info:

      fa_url: "{{ target_url }}"

      api_token: "{{ target_api }}"

    register: target_array


  - name: STRETCH POD

    purestorage.flasharray.purefa_pod:
```

name: pod-1

stretch: "{{ target_array['purefa_info']['default']['array_name'] }}"

fa_url: "{{ url }}"

api_token: "{{ api }}"

Exercise 2.3 - Configure asynchronous replication

Objective

Demonstrate the use of the purefa_pg module to configure the Protection Group created in Exercise 1.4 to replicate to the array connected in Exercise 2.0.

Guide

- The --- at the top of the file indicates that this is a YAML file.
- The hosts: localhost, indicates the play is run on the current host.
- connection: local tells the Playbook to run locally (rather than SSHing to itself)
- gather_facts: true enables facts gathering.
- The vars: parameter is a group of parameters to be used in the playbook.
- url: flasharray1.testdrive.local is the management IP address of your source FlashArray - change this reflect your local environment.
- api: e448c603-ecfd-8b4e-fc02-0d742e81a779 is the API token for a user on the source FlashArray - change this reflect your local environment.
- name: UPDATE PG is a user defined description that will display in the terminal output.
- purefa_pg: tells the task which module to use.
- The name parameter tells the module the name of the pod to either create or work with if the pod already exists..
- The target parameter is the name of the connected target array that the protection group is to replicate to. This name must exactly match the target name shown in the source arrays Array Connections list.
- The fa_url: "{{url}}" parameter tells the module to connect to the FlashArray Management IP address, which is stored as a variable url defined in the vars section of the playbook. This makes this array the source array in the replication pair.
- The api_token: "{{api}}" parameter tells the module to connect to the FlashArray using this API token, which is stored as a variable api defined in the vars section of the playbook.
-

Step 1:

Run the playbook - Execute the following:

```
$ ansible-playbook purefa-async.yml
```

Playbook Output

```
$ ansible-playbook purefa-async.yml


PLAY [ASYNC SETUP]
*******************************************************************************************************


TASK [Gathering Facts]
***************************************************************************************************
ok: [localhost]


TASK [UPDATE PG]
*********************************************************************************************************
changed: [localhost]


PLAY RECAP
************************************************************************************************************
***
localhost              : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Verifying the Solution

Login to the source Pure Storage FlashArray with your web browser.

Navigate to the Protection -> Protection Groups window and select the Source Protection Group workshop-pg to see the target array has been configured.

Going Further

In the update pg task you were required to know the exact name of the target array to which the pod was being stretched.

To assist in making this process more automated you can perform a purefa_info task on the target array and use the response from this to find the array name automatically and use this in the stretch task.

This bonus exercise allows you to perform the pod stretch in a fully automated fashion.

Add the following information into the vars section of the playbook:

```
  target_url: flasharray2.testdrive.local

  target_api: 24a96e35-e0e2-806e-c0cc-eaf45e7fa887
```

Replace the task in your YAML file, with the following:

```
  - name: TARGET INFO

    purestorage.flasharray.purefa_info:

      fa_url: "{{ target_url }}"

      api_token: "{{ target_api }}"
    register: target_array


  - name: UPDATE PG
    purestorage.flasharray.purefa_pg:
      name: workshop-pg
      target: "{{ target_array['purefa_info']['default']['array_name'] }}"
      fa_url: "{{ url }}"
      api_token: "{{ api }}"
```

Exercise 2.4 - Configuring FlashArray replication schedules

Objective

Demonstrate the use of the purefa_pgsched module to manage the local snapshot and replication schedules for a protection group.

Guide

- The --- at the top of the file indicates that this is a YAML file.
- The hosts: localhost, indicates the play is run on the current host.
- connection: local tells the Playbook to run locally (rather than SSHing to itself)
- gather_facts: true enables facts gathering.
- The vars: parameter is a group of parameters to be used in the playbook.
- url: flasharray1.testdrive.local is the management IP address of your FlashArray - change this reflect your local environment.
- api: e448c603-ecfd-8b4e-fc02-0d742e81a779 is the API token for a user on the FlashArray - change this reflect your local environment.
- name: LOCAL SNAP SCHEDULE | REPLICATION SCHEDULE is a user defined description that will display in the terminal output.
- purefa_pgsched: tells the task which module to use.
- The name parameter tells the module the name of the protection group whose schedule will be affected by the task.
- The schedule parameter specifies whether we are working on the local snapshot or the replication schedule.
- The enabled parameter defines whether schedule is enabled or not.
- The snap_frequency parameter defines the local snapshot frequency in seconds.
- The replicate_frequency parameter defines the replication frequency in seconds.
- The snap_at parameter defines the preferred time of day that local snapshots are generated when the snap_frequency is a multiple of 86400 (ie. 1 day).

- The replicate_at parameter defines the preferred time of day that replication snapshots are generated when the snap_frequency is a multiple of 86400 (ie. 1 day).
- The days parameter defines the number of days to keep the per_day local snapshots beyond the all_for period before they are eradicated.
- The per_day parameter defines the number of per_day local snapshots to keep beyond the all_for period..
- The target_per_day parameter defines the number of per_day replica snapshots to keep beyond the target_all_for period.
- The all_for parameter specifies the number of seconds to keep local snapshots before they are eradicated.
- The target_all_for parameter specifies the number of seconds to keep replica snapshots before they are eradicated.
- The fa_url: "{{url}}" parameter tells the module to connect to the FlashArray Management IP address, which is stored as a variable url defined in the vars section of the playbook. This makes this array the source array in the replication pair.
- The api_token: "{{api}}" parameter tells the module to connect to the FlashArray using this API token, which is stored as a variable api defined in the vars section of the playbook.

Step 1:

Run the playbook - Execute the following:

```
$ ansible-playbook purefa-sched.yml
```

Playbook Output

```
$ ansible-playbook purefa-sched.yml


PLAY [PROTECTION GROUP SCHEDULING]
**************************************************************************************


TASK [Gathering Facts]
**************************************************************************************************

ok: [localhost]


TASK [LOCAL SNAP SCHEDULE]
**********************************************************************************************

changed: [localhost]


TASK [REPLICATION SCHEDULE]
*********************************************************************************************

changed: [localhost]
```

```
PLAY RECAP
****************************************************************************************************************
***

localhost              : ok=3   changed=2   unreachable=0   failed=0   skipped=0   rescued=0   ignored=0
```

Verifying the Solution

Login to the source Pure Storage FlashArray with your web browser.

Navigate to the Protection -> Protection Groups window and select the workshop-pg group to see the two replication schedule configurations.

Going Further

Replication schedules can have blackout periods assigned to them to stop replica snapshots being created in a specific time window. Change the REPLICATION SCHEDULE task previously used to the following and rerun the playbook:

```
  - name: REPLICATION SCHEDULE WITH BLACKOUT
    purestorage.flasharray.purefa_pgsched:
      name: workshop-pg
      schedule: replication
      replicate_frequency: 3600
      target_per_day: 10
      target_all_for: 7
      blackout_start: 2AM
      blackout_end: 7AM
      fa_url: "{{ url }}"
      api_token: "{{ api }}"
```

Exercise 2.5 - Configuring common infrastructire settings on a FlashArray

Objective

Demonstrate the use of multiple FlashArray modules to configure common infrastructure settings on a FlashArray.

In this example we will set the the NTP servers using the purefa_ntp module, the DNS servers using the purefa_dns module and the SYSLOG server using the purefa_syslog module.

Guide

- The --- at the top of the file indicates that this is a YAML file.
- The hosts: localhost, indicates the play is run on the current host.
- connection: local tells the Playbook to run locally (rather than SSHing to itself)

- gather_facts: true enables facts gathering.
- The vars: parameter is a group of parameters to be used in the playbook.
- url: flasharray1.testdrive.local is the management IP address of your source FlashArray - change this reflect your local environment.
- api: e448c603-ecfd-8b4e-fc02-0d742e81a779 is the API token for a user on the source FlashArray - change this reflect your local environment.
- name: CONFIGURE NTP | DNS | SYSLOG is a user defined description that will display in the terminal output.
- purefa_*: tells the task which module to use.
- The other parameters tells the modules what values to use for the appropriate configuration values. More details can be found in the online Ansible Collection documentation.
- The fa_url: "{{url}}" parameter tells the module to connect to the FlashArray Management IP address, which is stored as a variable url defined in the vars section of the playbook. This makes this array the source array in the replication pair.
- The api_token: "{{api}}" parameter tells the module to connect to the FlashArray using this API token, which is stored as a variable api defined in the vars section of the playbook.

Step 1:

Run the playbook - Execute the following:

```
$ ansible-playbook purefa-infra.yml
```

Playbook Output

```
$ ansible-playbook purefa-infra.yml


PLAY [INFRASTRUCTURE SETTINGS]
******************************************************************************************



TASK [Gathering Facts]
**************************************************************************************************

ok: [localhost]



TASK [CONFIGURE NTP]
****************************************************************************************************

changed: [localhost]



TASK [CONFIGURE DNS]
****************************************************************************************************

changed: [localhost]
```

```
TASK [CONFIGURE SYSLOG]
*************************************************************************************************

changed: [localhost]


PLAY RECAP
*************************************************************************************************************
***

localhost              : ok=4    changed=3    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Verifying the Solution

Login to the source (flasharray1) Pure Storage FlashArray with your web browser.

Navigate to the Settings -> System window to see the array configuration. There are sub-windows specifically for Array Time, where the NTP servers are listed, and Syslog Servers.

To see the DNS domain and nameserver settings navigate to Settings -> Network page and look at the DNS Settings sub-window.

Going Further

Configuring the infrastructure settings is only the tip of the iceberg for FlashArray. Using these and other infrastructure related modules for the FlashArray you can build very complex playbooks to configure and maintain your critical configurations.