Cisco DevNet Associate – Skills Bases Assessment

Objectives

Part 1: Prep Work to be Completed Before the Exam

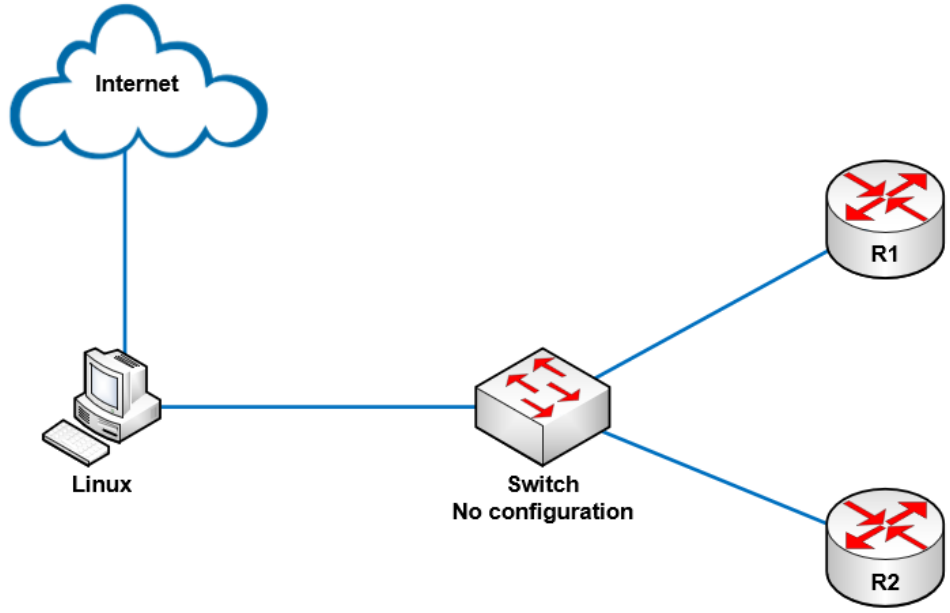
Part 2: RESTful API

Part 3: Docker

Part 4: Ansible

Part 5: Upload Your Exam to GitHub

Topology



Equipment

- A Linux (Ubuntu preferred) physical or virtual machine (Do **NOT** use the DEVASC-VM)
- Installed software (see below)
- 2 real or virtual routers (with a preconfigured ethernet interface and SSH access)
- 1 switch if using real equipment (unmanaged – no configuration)
- Network connection to the Internet and to real or virtual routers

Part 1: Prep (should be completed prior to starting the exam)

1. From your preferred hypervisor, create a new Linux VM (Ubuntu recommended).
2. Name the user account: **cisco** and the password: **cisco**.
3. Make all changes to your VM as needed (language, keyboard, timezone, permissions, etc.).
4. Install these packages in the VM:
   - Python3
   - Pip3
   - Go (if using CISSHGO)
   - Git
   - Postman
   - Docker
   - Ansible and ansible.netcommon plugin
   - VS Code
   - netmiko
5. Clone the **devnetsba** repository to the **/home/cisco** directory.

   https://github.com/Scott4564/devnetsba.git

   Note: The entire skills exam must be placed in this directory.
6. You will need at least 2 routers for this exam. Use Git to download this repository into the main skills exam directory:
   Note: Only download this if you have no other routers available for the exam.

   https://github.com/Scott4564/cisshgo_for_devnet.git
7. Create these subdirectories:
   - **devnetsba**
     - **restapi**
     - **docker**
     - **ansible**
       - **backups**
     - **cisshgo** (optional)
8. If you are using the CSR1000v routers, you will have to add support for sha1 hash to Linux. Open the SSH config file and add this data:

   sudo nano ~/.ssh/config

```
Host *
  kexAlgorithms +diffie-hellman-group-exchange-sha1,diffie-
hellman-group14-sha1
```

.

Begin Exam

Part 2: Run First

From the devnetsba directory, make the file **`run-first.sh`** executable and run it.
Note: This step MUST be completed before the exam will be graded.

Part 3: Restful API (20 pts)

NASA's InSight Mars lander takes continuous weather measurements (temperature, wind, pressure) on the surface of Mars at Elysium Planitia, a flat, smooth plain near Mars' equator.

This API provides per-Sol summary data for each of the last seven available Sols (Martian Days). As more data from a particular Sol are downlinked from the spacecraft (sometimes several days later), these values are recalculated, and consequently may change as more data are received on Earth.

1. Go to: **`https://api.nasa.gov/index.html`** and signup for a free account.
2. You will be given an API key to use in this project.
   Note: Copy and paste this information into a Notepad or similar document.
3. You will use the **InSight: Mars Weather Service API** for this project. Use the information found there to complete the API.
4. Use Postman to create a new **GET** request named **InSight**.
5. Save it in a collection named **InSight**.
6. After the API runs successfully, then **send and download** the response.json to the **restapi** directory.
7. Export the collection and save it in the **restapi** folder (Under the Collections tab on the left, use the menu "**`...`**" for this).

Part 4: Docker (30 pts)

Setup and run a new Docker Ubuntu container from a Dockerfile.

8. From within the **docker** directory, create a new **`Dockerfile`** with the following parameters:
   A. Use the **`FROM`** argument to load the latest **`Ubuntu`** image
   B. Use the **`MAINTAINER`** argument to add your **`name`** and **`email_address`**
   C. Use the **`WORKDIR`** argument to the path of **`/home/cisco/devnetsba/docker`**
   D. Use the **`RUN`** argument these two commands:
      - **`apt update`**
      - **`apt install arp-scan -y`**
   E. Use the **`CMD`** argument to use **`arp-scan`** to scan the **`172.17.0.0/16`** network (cisshgo network), your practice network, or your local network
   F. Save and exit
9. Build an image named **devnet-scan** to include everything in the Docker directory (may take several minutes to run).
10. Run the devnet-scan container in interactive mode and in a terminal (may take a while to run). Note: If this runs more than a minute use **`Ctrl + c`** to stop it.
11. Copy, paste, and save the build, run, and results commands/outputs to a file named **scan-results.txt**.

    Note: If you had to build multiple times, remove all images and containers except to correct ones. There should only be one container and one image shown (ubuntu image is ok the leave).

Part 5: Ansible (40 pts)

Use real equipment (preferred), CSR100v-VM, or CISSHGO to gather information about at least 2 routers.

From the **ansible** directory, create the appropriate inventory, ansible configuration, playbook files (three files only):

1. The hosts file needs to include:
   A. A group named **routers**
   B. The device with variables ***name***, **ansible_host=*ip.address*** **ansible_port=*port***
   Note: Port numbers are only needed when using cisshgo.
   C. Include these variables (7 in total):
      - ansible_connection
      - ansible_network_os
      - ansible_user
      - ansible_password
      - ansible_become
      - ansible_become_method
      - ansible_become_password
2. The ansible.cfg needs to include these variables (9 in total):
   A. Set the hosts file path to your local inventory
   B. Don't worry about RSA Fingerprints
   C. Do not create .retry files on job failure
   D. Allow Ansible to make 20 connections at one time
   E. Make Python3 the Interpreter for Ansible
   F. Silence Action Warnings
   G. Do not show deprecation warnings
   H. Use YAML as the callback plugin
   I. Use the stdout_callback when running ad-hoc commands
3. An Ansible playbook named **backup.yml** needs to include:
   A. The play is named **Show Me the Money**
   B. Get information for the **routers** group
   C. Gather **no** facts
   D. Use a **network_cli** connection
   E. The first task is named **Show Configuration Data**
      - Show the running configuration data
      - Show the IP interface brief data
      - Show the version data
   F. The second task is named **Save the Output**
      - Save both data files into the **ansible/backups** directory and name them ***device_name*-config.txt** based on the inventory device name.
      Note: Be sure all the show command are added to a single file. You should only have two output files.
4. Run the playbook. Note: You may need to install Netmiko in Python3 if play fails.

Part 6: Upload Your Exam to GitHub (10 pts)

Upload you entire **devnetsba** directory to a branch of my **devnetsba** repository.

1. Remove the cisshgo directory if you downloaded it in Part 1.
2. Remove any other unneeded files.
3. Be sure you are in the **devnetsba** directory.
4. Use Git to initialize, add, commit, branch, and push your skills exam:
   A. Use your **FirstName_LastName** as the name for a new branch
   B. Make this the active branch
   C. Add everything in the directory to this branch
      Note: Do NOT merge branches with main
   D. Commit changes using your First (space) Last name as the message for the commit
   E. Upload the new branch to GitHub:

Note: Use this account information you are ready to upload.

Username: `devnetsba`

Password: `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`