Description

This reusable asset consist of a Python script that creates a bastion session with the Oracle OCI bastion service. The main purposes of this asset are:

* Demonstrate the usage of OCI Python SDK
* Create a simple OS independent command-line interface for creating bastion sessions
* Create a simple way to make reusable configuration
* Create a ssh command that works with Linux, Mac OS and Windows by providing the flexibility to configure either ssh command or putty commands.

The script creates bastion session over ssh, and creates an example command to set up the tunnel for the target application. Other protocols like RDP can then be tunneled over the ssh session through the bastion service.

Why use bastion service

Oracle Cloud Infrastructure (OCI) Bastion service, is a fully managed service providing secure and ephemeral Secure Shell (ssh) access to the private resources in OCI. OCI Bastion service, like the bastion fortress of medieval times, improves security posture by providing an additional layer of defense against external threats.

Accessing virtual services directly from the internet is a clear no-go. Best practices is to never expose compute resources directly, neither for ssh or RDP traffic. RDP is known to be one of the most common Initial Access vectors for ransomware types of attacks.

Common practice is to place a compute nude with a minimum OS in a DMZ as jump host, and always use this as the entrypoint. The main weaknesses with this model is:

* A extra computer-node that needs to be managed, monitored and patched
* Extra set of required resources with risk of misconfiguration
* The jump server will require an additional layer of user governance.

The OCI Bastion service removes the public and private virtual cloud networking (VCN) hassle for access to a jump host. No public IP is needed, resulting in no surface attack area or zero-day vulnerabilities with a dedicated jump host. Customers also eliminate shared credentials, broad access limits, and other bad habits of using jump hosts. OCI Bastion service integrates with OCI Identity and Access Management (IAM) and allows the organization to control who can access a bastion or a session and what they can do with those resources.

The OCI Bastion service exists in two flavors:

* Managed session

With managed sessions an agent is running on the compute node, and the bastion session connects to the agent and tunnels ssh through the agent. The managed Session makes it possible to connect to a compute node from other networks without configure routing between the network where the compute node resides, and the network the bastion connection is initiated from.

* Port Forwarding

In this mode the bastion service does not tunnel though the agent, but the bastion service must have access to the subnet where the compute node resides, and the subnet security list

For additional description of the bastion service please review:

Requirements

The python script require Pyhton 3.x. As always use the lates secure patched version.

Latest version of the OCI CLI

Install guide:

The file requirements.txt lists the Python modules required.

Install the required modules with pip install -r requirements.txt

The script has the following command-line parameters:

Script Usage

In addition, the script require a JSON configuration file. The following elements exists in the configuration file

"sessions": JSON array with one JSON config entry pr. Session

"ociconfigurations": JSON Array with one JSON element pr. OCI SDK configuration profile.

Each “session” element has the following elements:

            "sessionName": Session name, from –session commandline parameter

            "sessionType":"PORT\_FORWARDING" or MANAGED

            "OCIConfig": name of profile, it looked up in the ociconfigurations array

            "bastionOCID": OCID to the configured Bastion service

            "bastionPublicKeyFile":file with public key to the Bastion SSh session

            "bastionPrivateKeyFile":Private key of bastion session key pair,

            "targetPrivateKeyFile":Used for reference in the target ssh command,

            "targetPort": Portnumber for the target SSh tunnel though the bastions Service. Use standard port for RDP fi target is RDP,

            "localPort":"2222",

            "sessionDisplayName": Display name of the session in the OCI Console

            "targetOCIDID":OCID of the target service,

            "targetPrivateIP":IP Adress of the target

            "osUserName": Used for the generated ssh command ,

            "ociRegion": Region where the target resoruce runs,

            "timetolive":Time the Bastion tunnel lives. Maximum value is 3600 sec.,

            "maxWaitCount":If the script creates the tunnel after creation of the session, maximum number of retries

            "waitRefresh": time in sec.between each retry to establish the tunnel

"ociconfigurations": Array of OCI configurations

Each array entry got the following JSON elements

          "configName": Name of entry. Referenced from a session element above

          "configFileName": Path to the OCi CLI configuration file

          "profileName": Name of profile in the OCI CLI configuration file

(Example file)[files/sample.json]

Script commandline options.

 --configfile  name of json configfile with named session and OCI CLI config info

 --session     named session, section in config file

 --exec        executes the ssh command and establishes the ssh connection

 --loglevel    logging level, info or debug. default info

 --log         logging output file or stdout, defaul stdout

Example command:

Python bastionsession.py –configfile bastion.json –session PORT-demo

Sample output