Check Point R80 API integration with Ansible and Amazon Web Services

Ryan Darst
US Solution Center
August 15, 2017
Version 1.2

Overview

This document will discuss the capabilities of the Check Point R80.10 management server to automatically provision security in Amazon Web Services (AWS) with Check Point vSEC gateways. This example deployment is driven completely by automation scripts by Anisble.

Ansible is an open source automation platform. Check Point has provided a module for Ansible to integrate with the Check Point R80 management platform. This module is available at https://community.checkpoint.com. Ansible can be used to automate complex tasks with a tool that is easy to use.

In this demonstration environment we will use Ansible to create all of the following

- Check Point Network Objects, Hosts, Security Rules, and Policy
- AWS Network Elements - VPC, Subnets, Security Groups
- AWS EC2 Elements – Instances, AutoScale Groups, Load Balancers
- AWS Route53 Elements – Register the External Load Balancer to DNS

As part of the automation, it is also possible to remove the environment as well. Simply running a single script can remove everything that was created with the deployment script.

By modifying simple variables in Ansible it would be possible to have deployments occur in every region around the world by simple changing the region location variable in the scripts.

The Check Point gateways used in this demo are using dedicated IOPS for faster startup. AWS does charge a bit more for this feature. If you would like to use the standard EBS storage simply delete the volumes section of the Check Point Edge Instance and Create CHKP Launch Config for Demo as part of the aws-vpc-create.yml.

Requirements

The needed elements are listed below to build out a simple environment

- The example files are located with this document
- An R80.10 management server with auto-provisioning setup and the vSEC controller configured. An example autoprovision file is available in the example files. See SK112575 for more information on setup of the vSEC controller
- A Linux instance. Ubuntu 16.04LTS was used in this demo.
- A Route53 Domain Name if you would like to have the DNS name for the External Load Balancer automaticity provisioned.

Setup the Environment

R80.10 Setup

Make sure that you have an R80.10 instance that is setup and working with AWS in the proper region. An example autoprovision.json is available in the example scripts file. Make sure to use either IAM roles for the R80.10 Manager in AWS or credentials for an off premise manager.

The file location and name is /opt/CPsuite-R80/fw1/conf/autoprovision.json

Adjust the file to fit your environment and region. This example was built in the US-East-2 region.

The demonstration R80.10 server was also running in AWS. This provides easy access by using IAM roles in AWS to access the details of the AWS account. This is documented in SK112575 as well. It is also possible to use an on premise R80.10 server as well.

A user account is needed on the R80 manager. The demo uses an account of api_user and password of vpn123. Please adjust as needed to your environment.

The linux machine where ansible is running will also need to be a trusted client to the R80.10 manager.

The R80.10 API blade will need to be setup to allow access, configure it from the Manage and Settings configuration in R80.10.

Linux – Ansible Setup

Setup a standard Linux install either on premise or in AWS. Either one will work as long as they have access to the AWS API and also the R80.10 management server.

Setting up Ansible is very easy. With a standard install of Ubuntu 16.04LTS it takes a few packages to make it active. A number of additional packages are required to be used with Ansible and AWS (boto/boto3).

Uncompress the example files on the Ubuntu server. The Ansible scripts are in the ansible directory.

Once you are logged into the Ubuntu Instance, run these commands to install Ansible.

```
sudo su -

#Update System
apt-get update
apt-get -y upgrade

#Install Ansible
apt-get -y install software-properties-common
apt-add-repository -y ppa:ansible/ansible
apt-get update
apt-get -y install ansible

#Install Python and Python-Pip
apt install -y python-pip
pip install --upgrade pip
pip install simplejson
pip install netaddr

#Install boto
pip install boto
pip install boto3

#Install unzip
apt-get -y install unzip
```

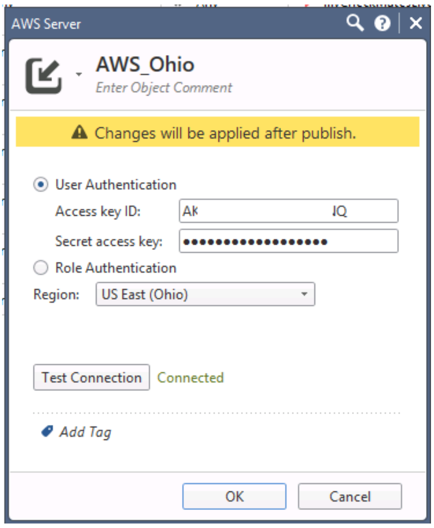
Once Ansible is installed a few statements are needed in the vars_ohio.yml file (this file is included in the examples file) in the ansible directory to setup some specific variables for the R80.10 management server.

Run the command “api fingerprint” on the R80.10 Management server. Take the SHA1 Fingerprint and replace the data on the mgmt_fingerprint variable in the vars_ohio.yml file. This is used to verify the server when connecting and is required with the Ansible module.

The example below is for an R80.10 management server at 10.17.0.245 using the api_user/vpn123 credentials to login and the fingerprint from the server.

Note the user has to be defined on the R80 manager and the ansible server setup as a trusted GUI client!

The R80_AWS_DataCenter variable is the Data Center object that is configured in R80. This is used to create the tagged objects as part of the demo with the vSEC controller



AWS_Ohio
Enter Object Comment

Changes will be applied after publish.

● User Authentication
Access key ID: AK IQ
Secret access key: ●●●●●●●●●●●●●●●●●
○ Role Authentication
Region: US East (Ohio)

Connected

.

```
#R80_Vars
mgmt_server: 10.17.0.245
mgmt_user: api_user
mgmt_password: vpn123
mgmt_fingerprint:
C2:F7:A0:B5:07:11:23:F1:CA:DE:A2:A0:A7:DE:C9:B4:03:12:F6:91
#DataCenter Name in R80/R80.10 for the AWS DataCenter Object
R80_AWS_DataCenter: "AWS_Ohio"
```

Setup the AWS credentials in the example source_aws file. Update this with a current access keys for your account. These credentials can be created from AWS IAM under Users/Access Keys.

The last step is to install the Check Point Ansible Module. Follow the directions as part of the readme that is included in the module from Exchange Point. The current file as of this writing is check_point_mgmt_v1.0.1.zip. Here are simple directions on the install. As of Ansible 2.3 the files should be placed into a custom library directory as shown below.

```
#Copy the CHKK Anisble module to the ubuntu machine and
install - Get the latest from ExchangePoint
#Latest version at writing is check_point_mgmt_v1.0.1.zip
mkdir /usr/share/ansible
cd /usr/share/ansible
#Place check_point_mgmt_v1.0.1.zip file here
cp  check_point_mgmt_v1.0.1.zip /usr/share/ansible
cd /usr/share/ansible
unzip check_point_mgmt_v1.0.1.zip
# Update the path per the directions!!!
# Remember to change the sys.path.append line in the
check_point_mgmt.py file accordingly)

Update the /etc/ansible/ansible.cfg
Uncomment the library line and point it to the
/usr/share/ansible directory
```

```
[defaults]

# some basic default values...

#inventory       = /etc/ansible/hosts
library         = /usr/share/ansible/
#module_utils    = /usr/share/my_module_utils/
#remote_tmp      = ~/.ansible/tmp
```

Run the Demo

Once the R80.10 server and the Linux host are running it is now ready to launch the demo.

Double check that all the variables in the vars_ohio.yml match what you want to launch including the ip scheme for the new VPC that will be created, SSH keyname, and Route 53 DNS Zone where you will place the dns name.

If you are satisfied with the values it is time to launch the environment.

Run the following command to load the AWS credentials to the environment variables

source source_aws

Run the following command to start the Ansible deployment.

```
ansible-playbook CheckMates_DEMO_Deploy.yml
```

If any errors occur address them and restart the deployment. You can also run each element of the deployment file one at a time by just using the respective yml file.

Watch the AWS environment and the R80 management server as new gateways are provisioned. It will take roughly 10 minutes for the environment to be completely online and ready. Use the url of the as defined in the vars.yml file to access the external load balancer. In the example it is vsec-demo and your Route53 DNS Zone.

Once the demo is complete, destroy the deployment by running the command

```
ansible-playbook CheckMates_DEMO_Delete.yml
```