CARBON BLACK ARM YOUR ENDPOINTS	Carbon Black Sensor Balancer 12/16/2015 dev-support@CarbonBlack.com

Overview
Configuration
Appendix A

Overview

The sensor load balancer is an API script used in conjunction with other configurations and techniques to equally distribute sensors among multiple Cb Enterprise server instances or clusters. It was written to address a common problem within a large enterprise deployment of Carbon Black (Cb) with multiple Cb Enterprise Server instances.

During the initial install a sensor is assigned to a single Cb Enterprise server instance by the provided INI file within the installation package. When there are multiple Cb Enterprise server instances within an enterprise, distribution can be difficult and require advanced software deployment techniques. Depending on the environment this can be challenging for the following reasons:

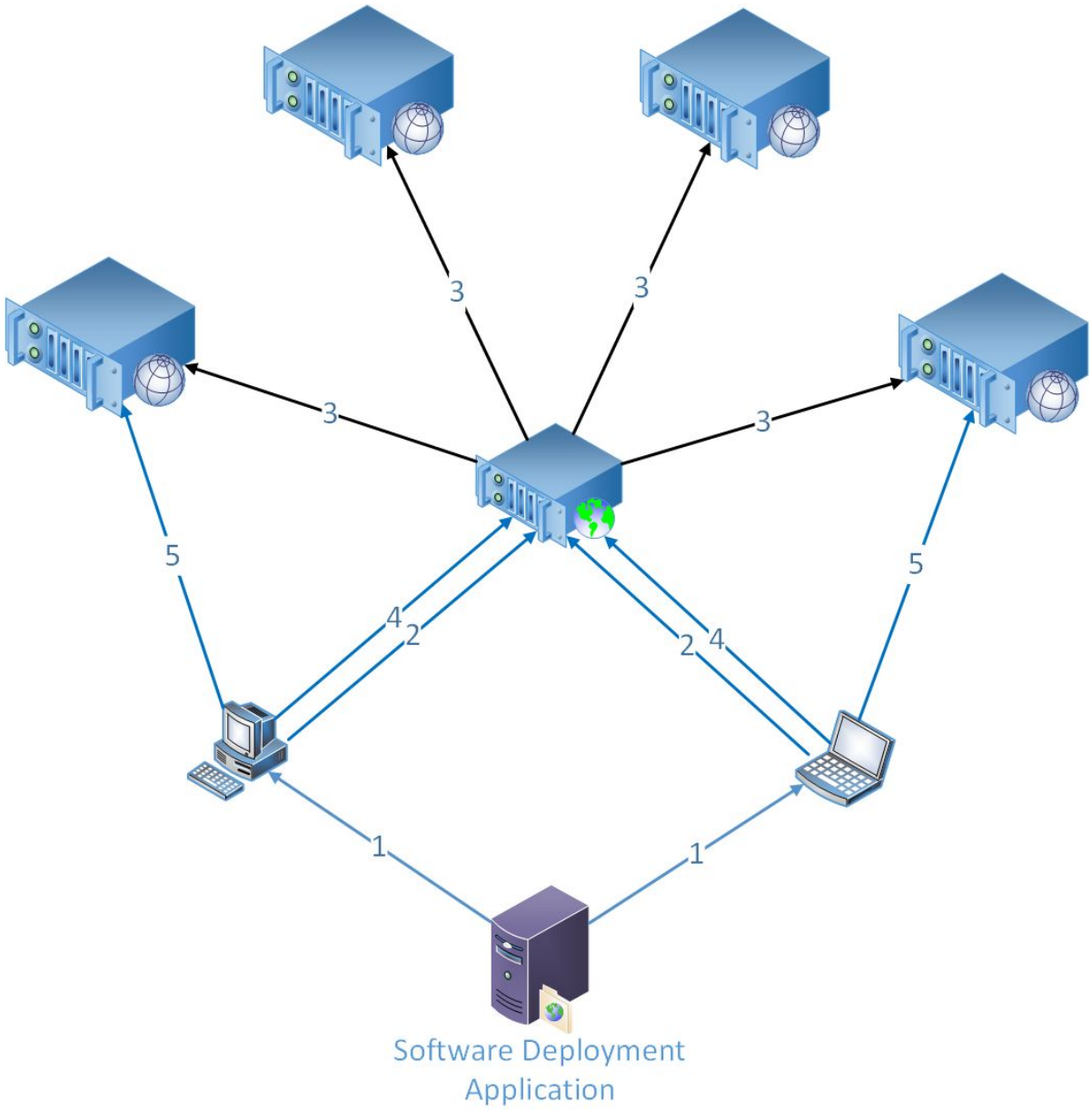
- The environment is not logically structured to deploy in such a manner (i.e Active Directory)
- Expertise or capability to "automate" this balance is limited and possibly unsustainable.
- The product is not natively "aware" of other instances and cannot automatically balance sensor deployments.

The sensor Balancer script will assist in this by allowing a single INI file and installation packaged to be used. Once installed it will communicate to a "Sensor Balancer" Cb Enterprise server. This can either be a dedicated server or an existing Cb Enterprise instance. This server will then run an API script on a regular basis to move the sensors to the appropriate sensor group that will redirect the sensor to the

new/appropriate instance where it will begin to push its data and reside for the life cycle of that sensor.

This chain of events is explained in the diagram below.

1. Software deployment pushes package and installs Cb Sensor using the base sensor group INI file
2. Cb Sensor initially registers and checks in to Cb “Sensor Balancer”.
3. At a regular interval the Cb “Sensor Balancer” will utilize the Cb API script to pull sensor deployment statistics from each participating cluster/instance. Sensor then gets moved via the API Script to the appropriate sensor groups for even distribution.
4. Sensor checks back in and receives new sensor group settings with appropriate cluster/instance assignment and URL.
5. Sensor checks in to assigned cluster/instance and begins to push data.



Configuration

The following instructions will walk you through configuring this script and the "Sensor Balancer".

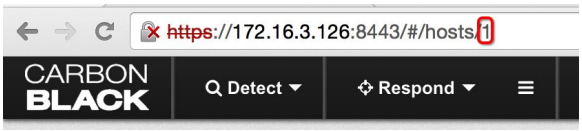
Pre-Requisites:

- Cb Enterprise Servers 5.x or greater
- All Cb Enterprise Servers participating in the balancing must be configured to allow sensor migrations.

1. Determine the Sensor Balancer

1. As mentioned above, this can be a current dedicated server or an existing Cb Enterprise server instance.
2. In larger organizations and deployments it may be appropriate to have a dedicated server to eliminate any possible extra load on an active instance.
3. This server will be where all new installations will originally report into.
4. This server will also house all of the needed configurations in order to properly implement this solution.

2. Configure User Accounts/API Tokens
    1. In order for the script to know the current sensor counts of a participating Cb Enterprise server instance it will need an API token to communicate.
    2. It is recommended to create a separate dedicated user account for account management and auditing purposes
    3. Record the API tokens
3. Site Throttling
    1. On the server that will be the "Sensor Balancer" you will want to configure a Site with 0Kbs throttling for 24/7.
    2. This will be assigned to the appropriate sensor groups to help ensure that no data is submitted by the sensors until it is appropriately assigned to the properly Cb Enterprise instance.
    3. If you are using a dedicated server it is recommend that you comment out the /data location of the nginx configuration to put in additional precautions in place to prevent data from being submitted by the sensors. Doing so will remove the capability of the server from being able accept sensor data if the sensors were to in avertedly submit data.
4. Sensor Groups
    1. On the server that will be the "Sensor Balancer" you will need to create at minimum 1 + the number of participating instances.
        1. One for the base sensor group.
            1. This sensor group will be where all sensors will originally land and be monitored by the API script.
            2. You will need to download the installation package from this group.
            3. Record the sensor group id located in the URL identified below



            4. Configure the 0Kbs Site for this group
        2. Migration Sensor Groups (One per participating Cb Server Instance)
            1. Configure the 0Kbs Site.
            2. Configure the Server URL of the desired Cb Enterprise server instance that the sensors will be moved/migrated too
            3. Record the Sensor Group ID

5. Script Configuration file
    1. Place the configuration file in /etc/cb/
    2. Modify the configuration file to include the sensor balancer information and participating Cb Server information (See Configuration File section below)
1. Script Logging file
    1. Place the configuration file in /etc/cb/
2. Script python file
    1. Place the python file in /usr/share/cb/
3. Cron job
    1. Create a new crontab with only the following value
        1. `*/5 * * * * cb /usr/bin/python /usr/share/cb/CbSensorBalancer.py >> /var/log/cb/job-runner/cbbalancer.out 2>&1`
    2. This will allow the script to run every 5 minutes and output logs to cbbalancer.out

Appendix A

Configuration File - `CbSensorBalancer.conf`

The Balancer section holds the configuration file information of the identified Sensor Balancer. If two sensor balancers and configurations are needed a separate file and script execution must occur.

```
[Balancer]
Name=US Servers
URL=https://127.0.0.1
Token=0123456789abcdef0123456789abcdef01234567
SSLVerify=False
BaseSensorGroup=1
```

The Balancer section has all of the following configuration options and requirements.

- Section Header - This value is required and cannot be changed
  `[Balancer]`
- Balancer configuration name - This value is text value for logging balancer activities based of the identified configuration file
  `Name=`
- Balancer API URL - URL used to connect to the Balancer's API
  `URL=`
- Balancer API token - Token used to connect to the Balancer's API
  `Token=`
- Balancer SSL Verification enforcement - Boolean True/False value used to enforce SSL certificate verification
  `SSLVerify=`
- Balancer Sensor Group - Integer representation of the sensor group that new sensors will land.
  `BaseSensorGroup=`

Participating Cb Server Instances Section - a complete section for each participating Cb Enterprise Server that sensors will be balanced too.

```
[Cluster1]
MigrateSensorGroup=2
URL=https://127.0.0.1:8443
```

```
Token=0123456789abcdef0123456789abcdef01234567
SSLVerify=False
```

These section has the following configuration options and requirements.

- Section Header - This value can be changed to identify the Cb Server Instance
  `[Cluster1]`
- Cb Server Instance Migration Sensor Group - Integer representation of the sensor group that new sensors will migrate to.
  `MigrateSensorGroup=`
- Cb Server instance API URL - URL used to connect to the Balancer's API
  `URL=`
- Balancer API token - Token used to connect to the Balancer's API
  `Token=`
- Cb Server instance SSL Verification enforcement - Boolean True/False value used to enforce SSL certificate verification
  `SSLVerify=`

Logging

By default the script will output INFO level logging to standard out using the configuration provided in CbSensorBalancer-logger.conf file. To enable more verbose logging modify the logger_root configuration log level to DEBUG as shown below.

From:

```
[logger_root]
level=INFO
handlers=stream_handler
```

To:

```
[logger_root]
level=DEBUG
handlers=stream_handler
```

In the configuration above, a cronjob was created and is logging to a file. This can be customized to log to any directory and file. However, ensure the directory is monitored by logrotate to ensure it does not continue to write to one large file. The logging available will provide insight into:

- What the Participating Cb Cluster/Instance sensor status is
- How many sensors available for balancing
- What was the decision made

- (verbosity depends on log level)

Script Run Modes

The CbSensorBalancer.py script has two run modes Simulation and Strict and can be ran togethe or seperately

Simulation Mode

`'-s'` or `'--simulate'`

This mode is used when performing testing or troubleshooting. It performs all script functionality except the actually moving the sensors to a new sensor group. This can allow you to verify all API calls and data looks appropriate before implementing or running script and assigning sensors to new sensor group and respectively new Cb Server Instance.

Strict Mode

`'-x'` or `'--strict'`

This mode is used to ensure that all participating clusters are reachable before assigning a sensor(s) to its appropriate group. If any API call attempted does not appropriate return data to verify the sensor count stats the script will log the error and gracefully exit, not moving any sensors. This is to prevent a Cb Server instance from unexpectedly receiving all new sensor due to other instances possible issues.

Backup and Restore

Backup

In order to backup ONLY the necessary configuration of a cb sensor balancer you must perform the following tasks. If a full backup and restore is required, follow the Carbon Black Backup and Restore documentation.

1. Backup certificates
   ```
   #tar -P --selinux -cvf cbcerts.tar /etc/cb/certs/ .
   ```
2. Backup Sensor Group configurations
   a. Stop Cb Enterprise services
      ```
      #service cb-enterprise stop
      ```
   b. Start PostgreSQL
      ```
      #service cb-pgsql start
      ```
   c. Backup required table data (requires root level permissions)

```
#pg_dump --data-only -t sensor_groups -t
sensor_client_certs -Fp -f sensorgroups.sql cb -p 5002
```

d. Stop PostgreSQL

```
#services cb-pgsql stop
```

e. Start Cb Enterprise services

```
#service cb-enterprise start
```

Restore

In order to restore ONLY the necessary configuration of a cb sensor balancer you must perform the following tasks. If a full backup and restore is required, follow the Carbon Black Backup and Restore documentation.

1. Copy `sensorgroups.sql` on to the current Cb Enterprise server
2. Restore certificates

```
#tar -P -xvf cbcerts.tar
```

3. Restore Sensor Group configurations
   a. Stop Cb Enterprise services

```
#service cb-enterprise stop
```

   b. Start PostgreSQL

```
#service cb-pgsql start
```

   c. Delete content of current required table data (requires root level permissions)

```
#psql cb -p 5002 -c 'TRUNCATE sensor_groups CASCADE'
```

   d. Restore data from backup

```
#psql cb -p 5002 -f sensorgroups.sql
```

   e. Stop PostgreSQL

```
#services cb-pgsql stop
```

   f. Start Cb Enterprise services

```
#service cb-enterprise start
```