EnhancedEndpointTracker Documentation

Release 1.0

agccie

Feb 12, 2019

Contents:

CHAPTER 1

Introduction

The Enhanced Endpoint Tracker is a Cisco ACI application that maintains a database of endpoint events on a per-node basis allowing for unique fabric-wide analysis. The application can be configured to analyze, notify, and automatically remediate various endpoint events. This gives ACI fabric operators better visibility and control over the endpoints in the fabric.

Features include:

- Easy to use GUI for viewing endpoint state and events within the fabric
- Per-node event history for each endpoint in the fabric. This allows administers to quickly verify that each node in the fabric has learned an endpoint correctly
- Analysis and Notifications for the following events:
  - Endpoint move
  - Off-subnet learns
  - Stale endpoint
- Notifications can be sent via syslog and email
- Automatically clear off-subnet endpoints
- Automatically clear stale endpoints
- Manually clear an endpoint through the GUI on user-selected nodes

This application uses Flask, MongoDB, and KnockoutJS

CHAPTER 2

Install

ACI-EnhancedEndpointTracker can be installed directly on the APIC as an ACI app or deployed as a standalone app.

Currently, the APIC imposes a **2G** memory limit and a **10G** disk quota on stateful applications. As a result, it may not be possible to run this as an ACI app on an APIC with a large number of endpoints.

As a best practice, it is recommended to deploy this app in standalone mode if the total number of per-node endpoints exceeds `65K`. You can determine the per-node endpoint count via the following moquery on the APIC:

```
apic# moquery -c epmDb -x query-target=subtree -x target-subtree-class=epmIpEp,
↪epmMacEp,epmRsMacEpToIpEpAtt -x rsp-subtree-include=count
```

If you have deployed the application on the APIC and it is exceeding the memory limits, you may see the symptoms below. **Note, there will be no impact** to the APIC or fabric under these conditions.

- Consistent monitor restarts
- Monitor restart due to "Worker 0 hello timeout"
- Monitor stuck at "Building endpoint database"

2.1 ACI Application

The most recent public release can be downloaded from ACI AppCenter. After downloading the app, follow the directions for uploading and installing the app on the APIC:

- 2.x Install Video Example
- 2.x Install Instructions
- 3.x Install Instructions

See *Building ACI Application* to build the ACI application from source.

2.2 Standalone Application

The standalone application is one that runs on a dedicated host/VM and makes remote connections to the APIC opposed to running as a container on the APIC. For large scale fabrics or development purposes, standalone is the recommended mode to run this application.

A pre-built OVA is available. After first boot of the OVA, execute the `firstRun.sh` script as described in step 3 of *Easy Setup*. The default credentials for the OVA are:

```
username: eptracker
password: cisco
```

Note: The OVA link may expire Jan 2019. Send an email to agossett@cisco.com if the link is no longer valid.

2.2.1 Easy Setup

The quickest way to get up and running is to spin up a host/VM/container and execute the install.sh script. This will install and configure python, apache, mongo, exim4, along with appropriate python requirements, cron, ntp, and logrotate. Additionally it will create a `firstRun` script that can be used to configure networking, ntp, and timezone for users unfamiliar with the OS. Lastly, it will execute the initial db setup.

1. Install Ubuntu Server 16.04 on a host or VM with the recommended minimal sizing:
   - 4 vCPU
   - 8G memory
   - 50G harddisk
2. From the terminal, download and execute the install script.

```
eptracker@ept-dev:~$ curl -sSl https://raw.githubusercontent.com/agccie/ACI-
↪EnhancedEndpointTracker/master/bash/install.sh > install.sh
eptracker@ept-dev:~$ chmod 777 install.sh
eptracker@ept-dev:~$ sudo ./install.sh --install
[sudo] password for eptracker:
Installing ............

Install Completed. Please see /home/eptracker/setup.log for more details. Reload the
machine before using this application.

After reload, first time user should run the firstRun.sh script
in eptracker's home directory:
   sudo /home/eptracker/firstRun.sh
```

3. After install, a `firstRun` script should be present in the install user's home directory. Execute the firstRun script to configure the VM along with setting up the initial app database.

```
eptracker@ept-dev:~$ sudo /home/eptracker/firstRun.sh

 Setting up system
 <snip>

 Setting up application
 Enter admin password:
```

(continues on next page)

(continued from previous page)

```
Re-enter password   :

        Setup has completed!
        You can now login to the web interface with username "admin" and the
        password you just configured at:
            https://192.168.5.231/


        It is recommended to reload the VM before proceeding.
        Reload now? [yes/no ] yes
Reloading ...
```

4. Setup is complete, the application can now be managed through the web interface.

Note: The source code is available at /var/www/eptracker. The apache module has been configured to service this directory. Any change to the python source code may require both python worker and apache to be restarted.

```
eptracker@ept-dev:/var/www/eptracker$ ./bash/workers.sh -ka
stopping all fabrics
eptracker@ept-dev:/var/www/eptracker$ sudo service apache2 restart
```

2.2.2 Upgrading

If you have downloaded the OVA you may want to upgrade the source code to the most recent release to get all recent fixes/features. To do so, simply perform a git pull on the source directory and restart apache. For example:

```
eptracker@eptracker:~$ cd /var/www/eptracker/
eptracker@eptracker:/var/www/eptracker$ git remote -v
origin      https://github.com/agccie/ACI-EnhancedEndpointTracker.git (fetch)
origin      https://github.com/agccie/ACI-EnhancedEndpointTracker.git (push)

eptracker@eptracker:/var/www/eptracker$ git reset --hard
<output omitted>
eptracker@eptracker:/var/www/eptracker$ git pull origin master
<output omitted>

eptracker@eptracker:/var/www/eptracker$ sudo service apache2 restart
```

2.2.3 Manual Setup

This application has primarily been developed and tested on Ubuntu host so that is recommended OS, however, any OS that supports the below requirements should work:

- Linux Distribution
- Flask with Python2.7
- MongoDB
- A webserver that can host flask applications
- exim4
  - exim4 is used only for sending email alerts via **mail** command. Alternative programs may also be used.

** Review the /bash/install.sh script for examples on installing python and all other dependencies **

2.3 Building ACI Application

To build the application you'll need a development environment with git, python2.7, zip, and docker installed.

Warning: Build process does not currently work on MAC OS due to incompatibility with sed program. It has successfully been performed on Ubuntu 16.04 and will likely work on other linux OS.

```
# install via apt-get, yum, dnf, etc...
root@ept-dev:~# apt-get install -y git python-pip zip

# install docker
root@ept-dev:~# curl -sSl https://get.docker.com/ | sh

# download the source code
root@ept-dev:~# git clone https://github.com/agccie/ACI-EnhancedEndpointTracker
root@ept-dev:~# cd ACI-EnhancedEndpointTracker

# install package requirements
root@ept-dev:~/ACI-EnhancedEndpointTracker# pip install aci_app_store/app_package/
↪cisco_aci_app_packager-1.0.tgz

# package application
root@ept-dev:~/ACI-EnhancedEndpointTracker# ./bash/build_app.sh
root@ept-dev:~/ACI-EnhancedEndpointTracker# ls -al ~/ | grep aci
-rw-r--r-- 1 root root    321062782 Nov 27 23:47 Cisco-EnhancedEndpointTracker-1.0.aci
```

Note: Docker is not required if the image file bundled within the app is available on the development environment. For example, you can install docker on a different server, bundle the required docker image file, and then sftp/scp to the development server.

```
# fetch the upstream docker image and copy to development server
root@srv1:~# docker pull agccie/ept:latest
root@srv1:~# docker save agccie/ept:latest | gzip -c > ~/my_docker_image.tgz
root@srv1:~# scp ~/my_docker_iamge.tgz root@ept-dev:~/

# package application with local docker image
root@ept-dev:~/ACI-EnhancedEndpointTracker# ./bash/build_app.sh --img ~/my_docker_
↪image.tgz
UTC 2017-11-27 23:47:17.083     INFO         build.py:(84): creating required ACI app␣
↪store directories
UTC 2017-11-27 23:47:17.481     INFO         build.py:(225): packaging application
UTC 2017-11-27 23:47:29.504     INFO         build.py:(236): packaged: ~/Cisco-
↪EnhancedEndpointTracker-1.0.aci
```

CHAPTER 3

Usage

3.1 Fabric Monitor Settings

The application can be configured to monitor multiple ACI fabrics. You can setup a fabric to monitor by clicking on the **New Fabric** button on the home page and then fill out the form. For existing monitors, you can click the fabric  button.

The options for each monitor are below:

- `Unique Fabric Name` The fabric name is used locally by endpoint tracker to distinguish between multiple fabrics. It must be a string between 1 and 64 characters and cannot be changed once configured. A short name such as **fab1** and **fab2** is recommended.
- `APIC Hostname` The hostname or IP address of a single APIC in the cluster. The application will use this IP to discover all other APICs in the cluster. If the initial APIC becomes unreachable, the other discovered APICs will automatically be used. Only out-of-band IPv4 address is currently supported for dynamic discovery of other APICs.
- `APIC Credentials` APIC username and password. User must have admin read access.
- `Switch SSH Credentials` Currently there is no API to clear an endpoint from a leaf. This application will SSH to the leaf and manually clear the endpoint. A username and password with ssh access to the leaf is only required if you need to clear endpoints within the app. The application can be set to ssh to the leaf TEP via the APIC or SSH directly to the switch out-of-band address.
- **The `Notification Options` The application can send syslog or email notifications for different events. At this time o**
  - Endpoint Move
  - Endpoint is becomes Stale on one or more leaves
  - An off-subnet endpoint is learned
- **`Remediate` The application can remediate potentially impacting endpoints by clearing them from the affected nodes. Re**

  - Automatically clear stale endpoints
  - Automatically clear off-subnet endpoints

- `Advanced Settings` Click the button for advanced settings. Generally these settings do not need to be changed unless needed for high scale environments.
  - perform endpoint move analysis
  - perform endpoint stale analysis
  - perform endpoint off-subnet analysis
  - `max_ep_events` maximum number of historical records per endpoint. When this number is exceeded, older records are discarded
  - `max_workers` maximum number of worker processes
  - `max_jobs` maximum queue size of pending events to processes. When this number is exceeded, the fabric monitor is restarted
  - `max_startup_jobs` maximum queue size of pending events to processes on initial endpoing build. When this number is exceeded, the fabric monitor is restarted
  - `max_fabric_events` maximum number of fabric monitor events. When this number is exceeded, older records are discarded

---

Note: When running in `app-mode` the fabric is automatically discovered when the app is installed on the APIC. The fabric name defaults to the controller fbDmNm, the APIC hostname is the docker gateway (**172.17.0.1** on most setups), and the APIC credentials user the app username with APIC created certificate. Only one fabric can be monitored in `app-mode`

Warning: There is a 2G memory set on the docker container while running in `app-mode`. Increasing the default max_jobs or max_startup_jobx can cause the application to crash. If fabric scale requires higher thresholds, consider moving application from `app-mode` to standalone mode.

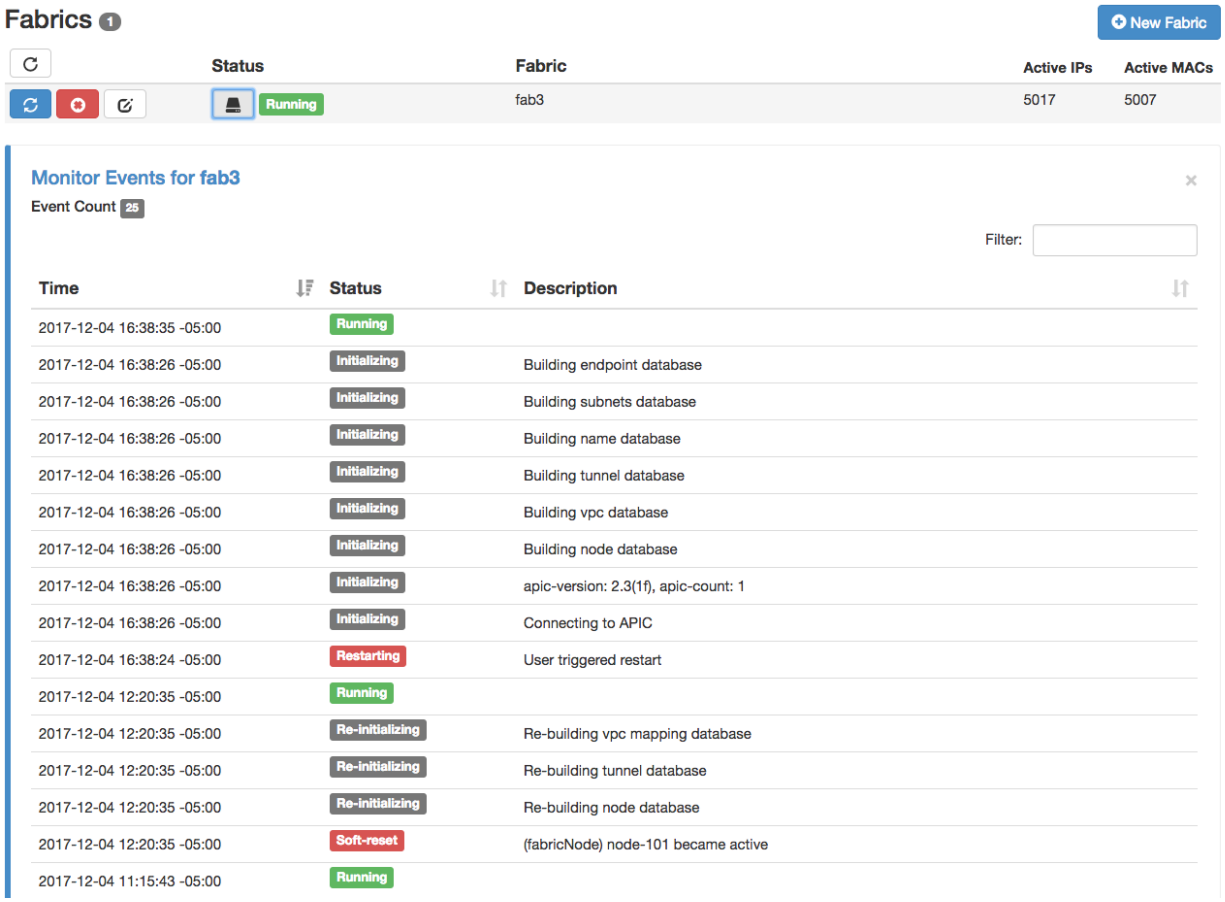
3.2 Controlling the Monitors

Once the fabric has been configured, you can view and control the status from the home page. Use the following buttons to control the fabric:

-  Start/Restart the monitors for the fabric
-  Stop the monitors for the fabric
-  Edit the fabric monitor settings
-  Refresh the status of all fabric monitors along with Top and Recent events
-  View history events of the fabric monitor.

3.3 Fabric Monitor History

The fabric monitor can be manually started or restarted. In addition, the monitor may restart if a new node comes online, a threshold such as max_jobs is exceeded, or a worker process has crashed. The history of restart events can be seen by clicking the  button. For example:

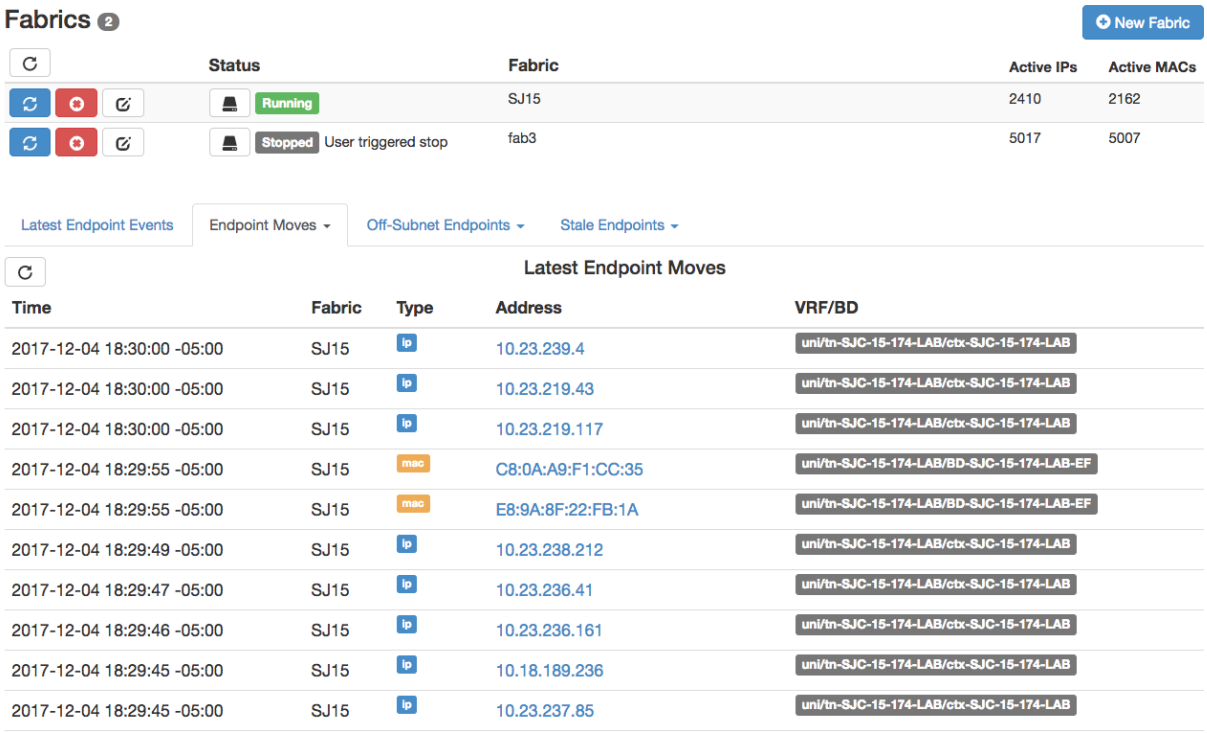


3.4 Fabric Overview

The fabric overview can be seen on the home page as soon as one or monitors are configured. The overview contains the last **50** records for the following events:

- `Latest Endpoint Events` - Each time an endpoint is created, deleted, or modified on a node the corresponding record will be created in the `ep_history` table. The most recent events are displayed here.
- `Latest Moves` - On each endpoint event, if `analyze_move` is enabled, a move analysis is performed. If the node, ifId, encap, pcTag, rw_bd, or rw_mac has changed between the last two local events, and the move is not a duplicate of the previous move, then a new entry is added to the `ep_moves` table. The most recent moves from the `ep_moves` table are displayed here.
- `Top Moves` - Each entry added to the `ep_moves` table has a corresponding count. The entries in the `ep_moves` table with the highest count are displayed here.

- `Currently Off-Subnet Endpoints` - On each IP endpoint event, if `analyze_offsubnet` is enabled, then analysis is performed to determine if endpoint is off-subnet. This is done by mapping the pcTag to bd_vnid via the `ep_epgs` table and then checking the IP against list of subnets for the corresponding bd_vnid in the `ep_subnets` table. If the IP is determined to be off-subnet, then entry is marked with `is_offsubnet` flag in the `ep_history` table. A job is added to the watch queue to ensure endpoint is still off-subnet after the transitory_offsubnet_time (30 seconds). If the `is_offsubnet` flag has not been cleared, then an `ep_offsubnet` table. The entries in the `ep_history` table with `is_offsubnet` flag set to True are display via `Currently Off-Subnet Endpoints`
- `Historical Off-Subnet Events` - This displays the latest IP endpoints added to the `ep_offsubnet` table.
- `Currently Stale Endpoints` - On each endpoint event, if `analyze_stale` is enabled, then analysis is performed to determine if the endpoint is stale on any node. This is performed by determining which node has learned the endpoint as a local entry (aware of vpc VTEP logic) and checking each node with an remote entry (XR) and ensuring it points back to the correct node. If the XR entry points to proxy or points to a node which has an XR bounce entry, this is also considered a correct learn. If the analysis determines the endpoint is stale, the `is_stale` flag is set in the `ep_history` table. A job is added to the watch queue to ensure the endpoint is still stale after the transitory_stale_time (30 seconds) or transitory_xr_stale_time (300 seconds) for entries that should be deleted from fabric. If the `is_stale` flag after the holdtime, then an entry is added to the `ep_stale` table. The entries in the `ep_history` table with `is_stale` flag set to True are displayed via `Currently Stale Endpoints`.
- `Historical Stale Endpoint Events` - This displays the latest endpoints added to the `ep_stale` table.



CHAPTER 4

Development

4.1 Setting Up the Environment

TODO

4.2 Packaging the App

TODO

4.3 Architecture

TODO

4.4 Tests

TODO

CHAPTER 5

Indices and tables

- genindex
- modindex
- search