**Install ELK On Ubuntu 18.04 Bionic Beaver Linux (for Cisco ASA)**

Original by: Nick Congleton

Ubuntu 18.04

19 September 2018

<https://linuxconfig.org/install-elk-on-ubuntu-18-04-bionic-beaver-linux>

**Intro**

**This document was modified to accept inputs from a Cisco ASA 5504 via syslog and display them in Kibana. Changes for the ASA syslog events will be noted via several Appendices**

**What is ELK**

If you're in a situation where you manage large amounts of data logs, the ELK stack is exactly what you're looking for. The ELK stack combines Elasticsearch, Logstash, and Kibana into a simple, yet powerful, open source stack that lets you manage large amounts of logged data from a convenient graphical web interface. All three tools are developed by Elastic, so they work in tandem perfectly, and they're very easy to get set up on your Ubuntu system.

**Set Static IP**

#### Ubuntu Server

To configure a static IP address on your Ubuntu 18.04 server you need to modify a relevant netplan network configuration file within /etc/netplan/ directory.

# This file describes the network interfaces available on your system

# For more information, see netplan(5).

network:

version: 2

renderer: networkd

ethernets:

enp0s3:

dhcp4: no

addresses: [192.168.1.178/24]

gateway4: 192.168.1.1

nameservers:

addresses: [8.8.8.8,8.8.4.4]

Once ready apply changes with:

$ sudo netplan apply

In case you run into some issues execute:

$ sudo netplan --debug apply

**Install the Dependencies**

Begin by installing the dependencies. These are all fairly common, though there are a couple of notes that you need to take into account. Of course, these are Nginx-based, so disable Apache or switch the port, if you're using it.   
  
Logstash doesn't support Java 10, which is available on Bionic from openjdk-11-jre. If you have it installed on your system, remove it. Use the older version until Logstash gets support.

$ **sudo apt install openjdk-8-jre apt-transport-https wget nginx**

**Add the Elastic Repository**

Elastic provides a complete repository for Debian based systems that includes all three pieces of software. You just need to add it to your system. Begin by importing their GPG key.

**$ wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -**

Next, add the repository. Create a file at **/etc/apt/sources.list.d/elastic.list**, and paste the following line into it.

**deb https://artifacts.elastic.co/packages/6.x/apt stable main**

Save that file, and exit. Update Apt.

**$ sudo apt update**

**Install Elasticsearch and Kibana**

You're now ready to install Elasticsearch and Kibana. They're available through Apt, so get them like you normally would.

**$ sudo apt install elasticsearch kibana**

You need to edit the Kibana configuration file at **/etc/kibana/kibana.yml** to tell it that the host server is localhost. The line is already there. Uncomment it.

**server.host: "localhost"**

Restart Kibana and start up Elasticsearch, and both will be ready to go.

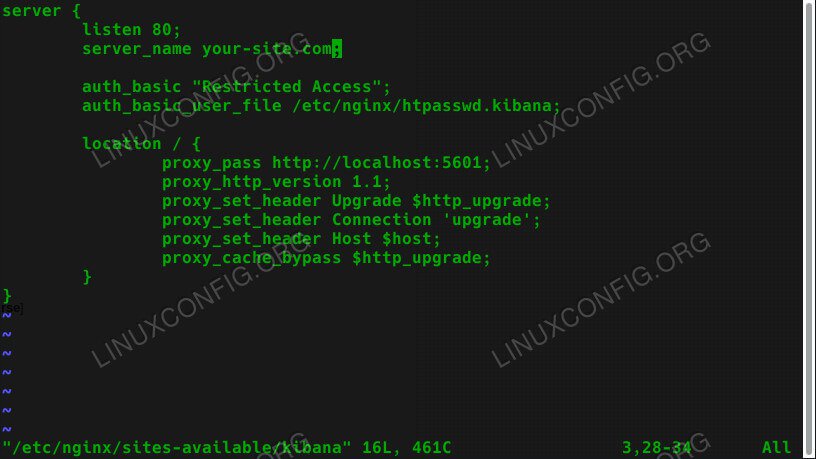
**$ sudo systemctl restart kibana**

**$ sudo systemctl start elasticsearch**

**Set Up Nginx**

Kibana is served through Nginx, so you need to set up a basic Nginx configuration to get it to serve your instance of Kibana. Start by creating a password for Kibana. This way, your server isn't accessible openly on the Internet. Use OpenSSL to generate the password, and place it in **/etc/nginx/htpasswd.kibana.** You can change the username to anything you want. In this instance, it's admin.

**$ echo "admin:`openssl passwd -apr1 <password goes here>`" | sudo tee -a /etc/nginx/htpasswd.kibana**

[](https://linuxconfig.org/images/ubuntu-18.04-elk-nginx-config.jpg)

Kibana Nginx Configuration On Ubuntu 18.04

After you have your password, create an Nginx configuration similar to the one below at **/etc/nginx/sites-available**. Make sure to use your actual server url or IP. The defaults should be good for everything else.

**server {**

**listen 80;**

**server\_name your-site.com;**

**auth\_basic "Restricted Access";**

**auth\_basic\_user\_file /etc/nginx/htpasswd.kibana;**

**location / {**

**proxy\_pass http://localhost:5601;**

**proxy\_http\_version 1.1;**

**proxy\_set\_header Upgrade $http\_upgrade;**

**proxy\_set\_header Connection 'upgrade';**

**proxy\_set\_header Host $host;**

**proxy\_cache\_bypass $http\_upgrade;**

**}**

**}**

After you have your configuration, remove the existing default config, and create a new symlink in **sites-enabled** for Kibana.

$ **sudo rm /etc/nginx/sites-enabled/default**

**$ sudo ln -s /etc/nginx/sites-available/kibana /etc/nginx/sites-enabled/kibana**

Restart Nginx for the changes to take effect.

**$ sudo systemctl restart nginx**

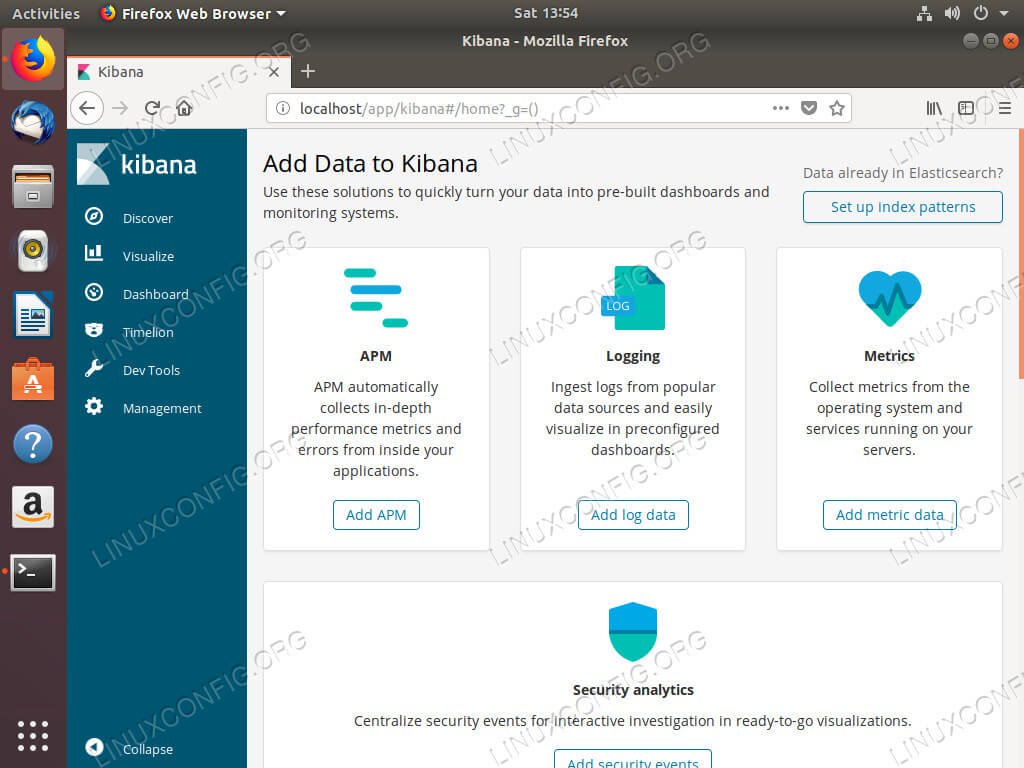
**Install Logstash**

The last thing that you need to do is install Logstash. Just install it with your package manager.

**$ sudo apt install logstash**

**Sign In To Kibana**

Open up your browser, and go to the address that you assigned to your Kibana instance in the Nginx configuration. You should be prompted to enter the username and password that you set up for Kibana. Enter them.

[](https://linuxconfig.org/images/ubuntu-18.04-elk-kibana.jpg)

Kibana Running On Ubuntu 18.04

Once you performed the configuration in the Appendices you will see data showing up in Kibana. You can begin using Kibana and setting up your preferences.

**Appendix A**

**Logstash Config (02.kirn.conf as of 11/6/18)**

File location: **/etc/logstash/conf.d**

File contents:

input {

udp {

port => 5514

type => "cisco-fw"

}

}

filter {

if [type] == "cisco-fw" {

grok {

match => ["message", "%{CISCO\_TAGGED\_SYSLOG} %{GREEDYDATA:cisco\_message}"]

} #end of grok-1

syslog\_pri { }

grok {

match => [

"cisco\_message", "%{CISCOFW106001}",

"cisco\_message", "%{CISCOFW106006\_106007\_106010}",

"cisco\_message", "%{CISCOFW106014}",

"cisco\_message", "%{CISCOFW106015}",

"cisco\_message", "%{CISCOFW106021}",

"cisco\_message", "%{CISCOFW106023}",

"cisco\_message", "%{CISCOFW106100}",

"cisco\_message", "%{CISCOFW110002}",

"cisco\_message", "%{CISCOFW302010}",

"cisco\_message", "%{CISCOFW302013\_302014\_302015\_302016}",

"cisco\_message", "%{CISCOFW302020\_302021}",

"cisco\_message", "%{CISCOFW305011}",

"cisco\_message", "%{CISCOFW313001\_313004\_313008}",

"cisco\_message", "%{CISCOFW313005}",

"cisco\_message", "%{CISCOFW402117}",

"cisco\_message", "%{CISCOFW402119}",

"cisco\_message", "%{CISCOFW419001}",

"cisco\_message", "%{CISCOFW419002}",

"cisco\_message", "%{CISCOFW500004}",

"cisco\_message", "%{CISCOFW602303\_602304}",

"cisco\_message", "%{CISCOFW710001\_710002\_710003\_710005\_710006}",

"cisco\_message", "%{CISCOFW713172}",

"cisco\_message", "%{CISCOFW733100}"

]

} #end of grok-2

geoip {

add\_tag => ["GeoIP"]

# database => "/usr/local/share/GeoLiteCity.dat"

source => "src\_ip"

target => "geoip"

}

if [geoip][city\_name] == "" { mutate { remove\_field => "[geopip][city\_name]" } }

if [geoip][continent\_code] == "" { mutate { remove\_field => "[geoip][continent\_code]" } }

if [geoip][country\_code2] == "" { mutate { remove\_field => "[geoip][country\_code2]" } }

if [geoip][country\_code3] == "" { mutate { remove\_field => "[geoip][country\_code3]" } }

if [geoip][country\_name] == "" { mutate { remove\_field => "[geoip][country\_name]" } }

if [geoip][latitude] == "" { mutate { remove\_field => "[geoip][latitude]" } }

if [geoip][longitude] == "" { mutate { remove\_field => "[geoip][longitude]" } }

if [geoip][postal\_code] == "" { mutate { remove\_field => "[geoip][postal\_code]" } }

if [geoip][region\_name] == "" { mutate { remove\_field => "[geoip][region\_name]" } }

if [geoip][time\_zone] == "" { mutate { remove\_field => "[geoip][time\_zone]" } }

date {

match => ["timestamp",

"MMM dd HH:mm:ss",

"MMM d HH:mm:ss",

"MMM dd yyyy HH:mm:ss",

"MMM d yyyy HH:mm:ss"

]

} #end of date

} #end of if

} #end of filter

#end of file

output {

elasticsearch {

hosts => ["localhost"]

index => "log-%{type}-%{+yyyyMM}"

}

} #end of output

#end of file

**Appendix B**

**Setup GeoLiteCity DB**

Download the GeoLiteCity database:

* wget <http://geolite.maxmind.com/download/geoip/database/GeoLiteCity_CSV/GeoLiteCity-latest.zip>

Extract the .dat file:

* unzip GeoLiteCity-latest.zip

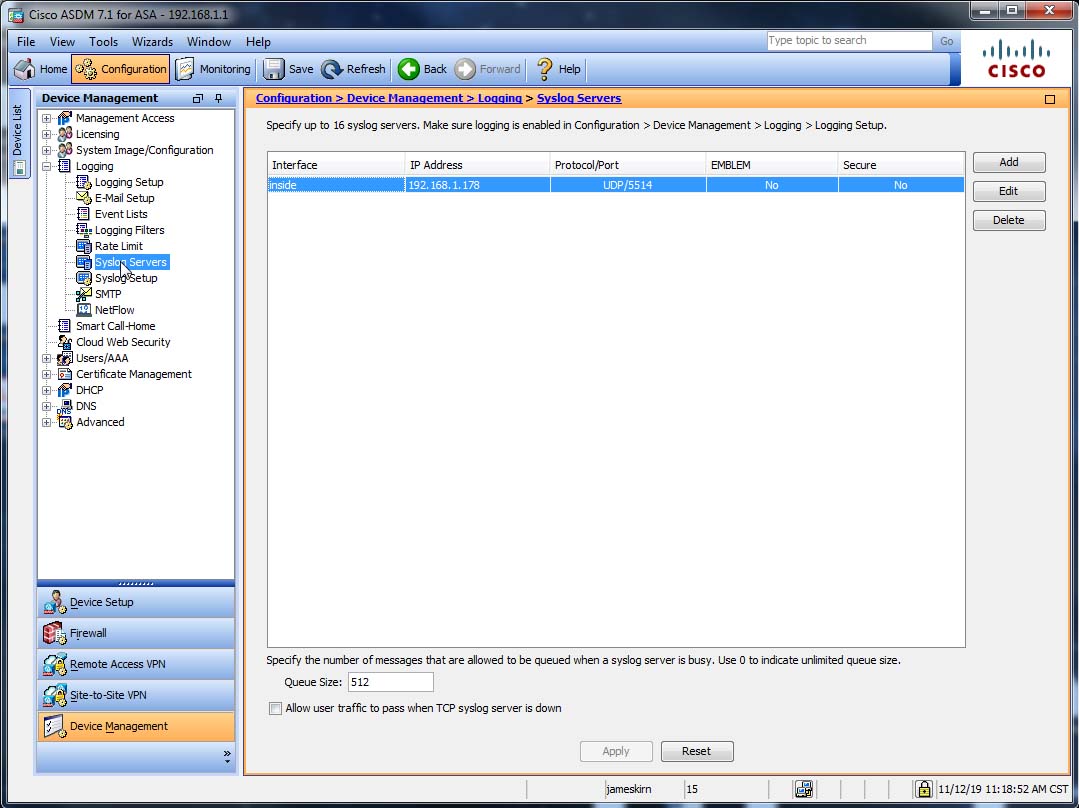
Move it to the expected location:

* cp GeoLiteCity-latest.dat /usr/local/share/GeoLiteCity.dat

**Appendix C**

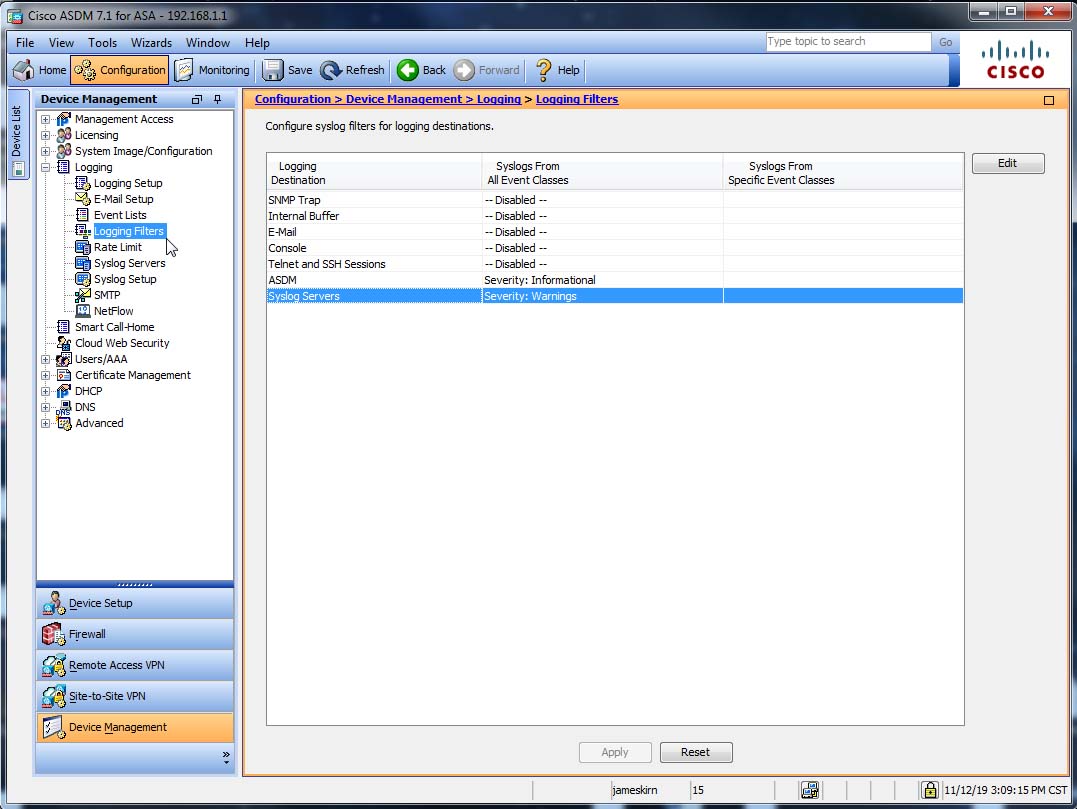
**Setup Cisco ASA 5504 to send syslog msgs**

Log into your ASA via ASDM and setup the Syslog Servers:



The IP address should point to your ELK server.

Also check the Logging Filters to make sure it is sending the events you desire:

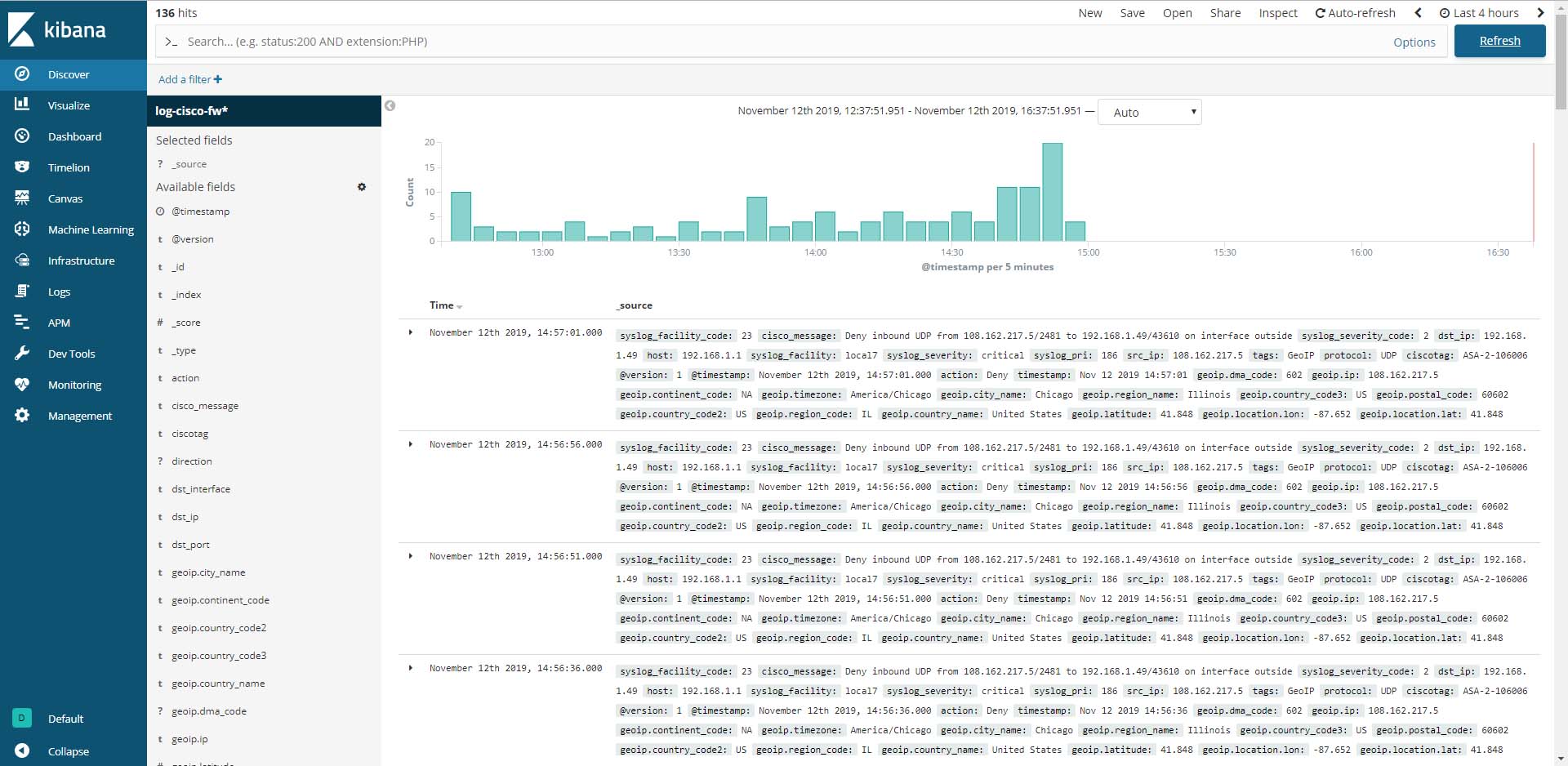


Make sure any changes you make get written to the ASA before you exit.

**Appendix D**

**Create a Dashboard in Kibana**

If all went well you should see events showing up in Kibana:

****

After you create your dashboard it should look something like this:

