**Automated ELK Stack Deployment Project**

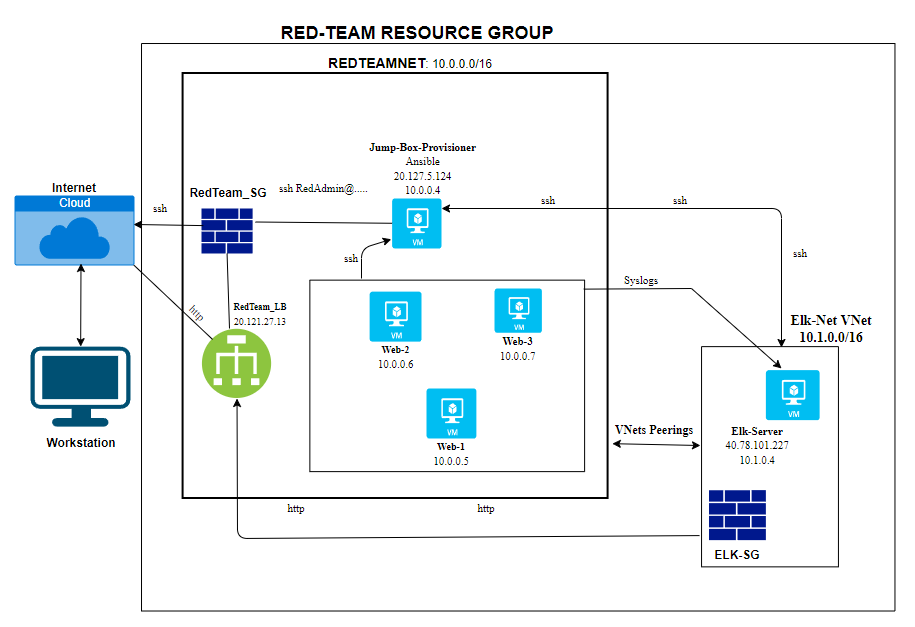
Below is the complete guide that shows how to quickly install and configure the Elk Stack. Elk Stack combines three open source tools (ElasticSearch, Logstash, Kibana) allowing the storage of large amounts of data. Elk stack provides a unified logging used to search through all the logs from multiple servers in a single place.

Highlights from this Project:

* Network Topology
* Load Balancing
* Access Policies
* How to Use the Ansible Configurations
* Elk Stack Configuration
  + Beats in Use
  + Monitoring Web Applications with Beats

**Description of the Network Topology**

The files in this repository were used to configure the network depicted below.



The main purpose of this network is to simulate a cloud based network hosting load-balanced vulnerable web applications using docker and configuration management systems.

Table 1.

| **Name** | **Function** | **IP Address** | **Operation System** |
| --- | --- | --- | --- |
| Jump Box | Gateway | 10.0.0.4 | Linux Ubuntu 18.04 |
| Web-1 | Web Server | 10.0.0.5 | Linux Ubuntu 18.04 |
| Web-2 | Web Server | 10.0.0.6 | Linux Ubuntu 18.04 |
| Web-3 | Web Server | 10.0.0.7 | Linux Ubuntu 18.04 |
| Elk-Server | Monitoring | 10.1.0.4 | Linux Ubuntu 18.04 |

**Load Balancing**

Load balancing is the process of distributing network traffic across multiple servers. It makes sure that no individual server takes too much demand or traffic which can cause an overload. Also, by distributing the network, load balancing ensures that the application will be highly **available**, in addition to restricting inbound access to the network.

A load balancer was used in the above network topology (Fig 1.). Three redundant web servers (Web-1, Web-2, Web-3) grouped together logically in a backend pool. Hence even if one of the servers is taken down, or flooded the load balancer will evenly distribute traffic to ensure that network is always **Available.** Load balancer’s targets are organized into different availability zones.

***Availability: West (US) Web-1, Web-2, Web-3***

***East (US) Elk-Server***

Adding a load balancer is very useful in mitigating the risk of a distributed denial-of-service attack (DDoS), spamming attacks, and other risks on Availability.

**Access Policies**

Access control policies ensure that only authorized users - namely, ourselves- will be able to connect in the first place. A network security group (RedTeamSG and Elk-SG) employed to ensure the confidentiality of the network, fulfilling the component of the CIA triad. Security rules were used to allow inbound traffic from the jump box provisioner for the adminitrator.

Using a jump box provisioner, only the provisioner can accept connections from the internet. Access is only allowed from the admin’s IP address. In addition, all the servers were configured to allow access from the administrator through port 22 (ssh) using the administrator’s ssh keys.

Also it is apparent to note that all but one (jump box) machines in the network cannot be publicly accessed.

**Automated Deployment and Elk Configuration**

Ansible was used to automate configuration of the machines. Ansible containers allow a user to build, service, and deploy **containers.** **Docker Containers** are an abstraction at the app layer that packages up code and all its dependencies so that the application runs quickly and reliably from one computing environment to another. Multiple containers can share resources and operating systems of a virtual machine.

Ansible playbooks are essentially framework or language by which ansible works. Ansible playbooks use YMAL (Yet Another Markup Language) which is a to-do list for Ansible that contains a list of all tasks. This enables easy deployment to multiple servers, lowering costs, creating uniformity and reducing errors.

***Installing Docker on the jump box for use***

* SSh into the jumpbox from a terminal using the admin and ssh key configured: ssh RedAdmin@20.120.114.21
* Update and install docker: sudo apt update then sudo apt install docker.io
* Check status of docker: sudo systemctl status docker
* Start docker: sudo systemctl start docker
* Pull the ansible container: sudo docker pull cyberxsecurity/ansible
* Launch ansible: docker run -it cybersecurity/ansible:latest bash
* To know the docker container name run: sudo docker container list -a
* Connect to the docker container: sudo docker start <<container\_name>> then: sudo docker attach <<container\_name>>

To run ansible playbooks to deploy web applications to the web servers, Ansible host file and ansible.cfg file must be updated.

Run: cd /etc/ansible and then

Use: nano hosts to uncomment and edit the host IP addresses of webservers: <<10.0.0.5, 10.0.0.6, 10.0.0.7>> to the line ansible\_python\_interpreter=/usr/bin/python3 respectively.

# Ex 2: A collection of hosts belonging to the 'webservers' group

[webservers]

10.0.0.5 ansible\_python\_interpreter=/usr/bin/python3

10.0.0.6 ansible\_python\_interpreter=/usr/bin/python3

10.0.0.7 ansible\_python\_interpreter=/usr/bin/python3

[elk]

10.1.0.4 ansible\_python\_interpreter=/usr/bin/python3

#alpha.example.org

#beta.example.org

Use: nano ansible.cfg to uncomment and remote\_user= <<root\_useradmin\_name>>

# default user to use for playbooks if user is not specified

# (/usr/bin/ansible will use current user as default)

remote\_user = RedAdmin

# logging is off by default unless this path is defined

# if so defined, consider logrotate

#log\_path = /var/log/ansible.log

# default module name for /usr/bin/ansible

#module\_name = command

Next task is to create an Ansible playbook that installs and configures all the virtual webservers with the DVWA web app.

Run: root@contianer\_ID:/etc/ansible/# nano playbook\_name.yml

Ansible playbook code used to install Docker and create the DVWA contianiers:

---

- name: Configure web vms with docker

hosts: webservers

become: true

tasks:

- name: docker.io

apt:

update\_cache: yes

name: docker.io

state: present

- name: Install pip3

apt:

name: python3-pip

state: present

- name: Install python docker module

pip:

name: docker

state: present

- name: download and launch our DVWA web container

docker\_container:

name: dvwa

image: cyberxsecurity/dvwa

state: started

restart\_policy: always

published\_ports: 80:80

- name: enable docker service

systemd:

name: docker

enabled: yes

To execute the playbook, run: bash ansible-playbook <<playbook\_name>>.yml  
This will execute all the commands in the playbook on each server as identified in the hosts file in the ansible directory. Successful completion would yield:

root@75937df21273:/etc/ansible/roles# ansible-playbook install.playbook.yml

[WARNING]: ansible.utils.display.initialize\_locale has not been called, this may result in incorrectly calculated text

widths that can cause Display to print incorrect line lengths

PLAY [Configure web vms with docker] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

TASK [Gathering Facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [10.0.0.5]

ok: [10.0.0.6]

ok: [10.0.0.7]

TASK [docker.io] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [10.0.0.5]

ok: [10.0.0.6]

ok: [10.0.0.7]

TASK [Install pip3] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [10.0.0.5]

ok: [10.0.0.6]

ok: [10.0.0.7]

TASK [Install python docker module] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [10.0.0.5]

ok: [10.0.0.6]

ok: [10.0.0.7]

TASK [download and launch our DVWA web container] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

[DEPRECATION WARNING]: The container\_default\_behavior option will change its default value from "compatibility" to

"no\_defaults" in community.docker 2.0.0. To remove this warning, please specify an explicit value for it now. This

feature will be removed from community.docker in version 2.0.0. Deprecation warnings can be disabled by setting

deprecation\_warnings=False in ansible.cfg.

ok: [10.0.0.5]

ok: [10.0.0.6]

ok: [10.0.0.7]

TASK [enable docker service] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [10.0.0.6]

ok: [10.0.0.5]

ok: [10.0.0.7]

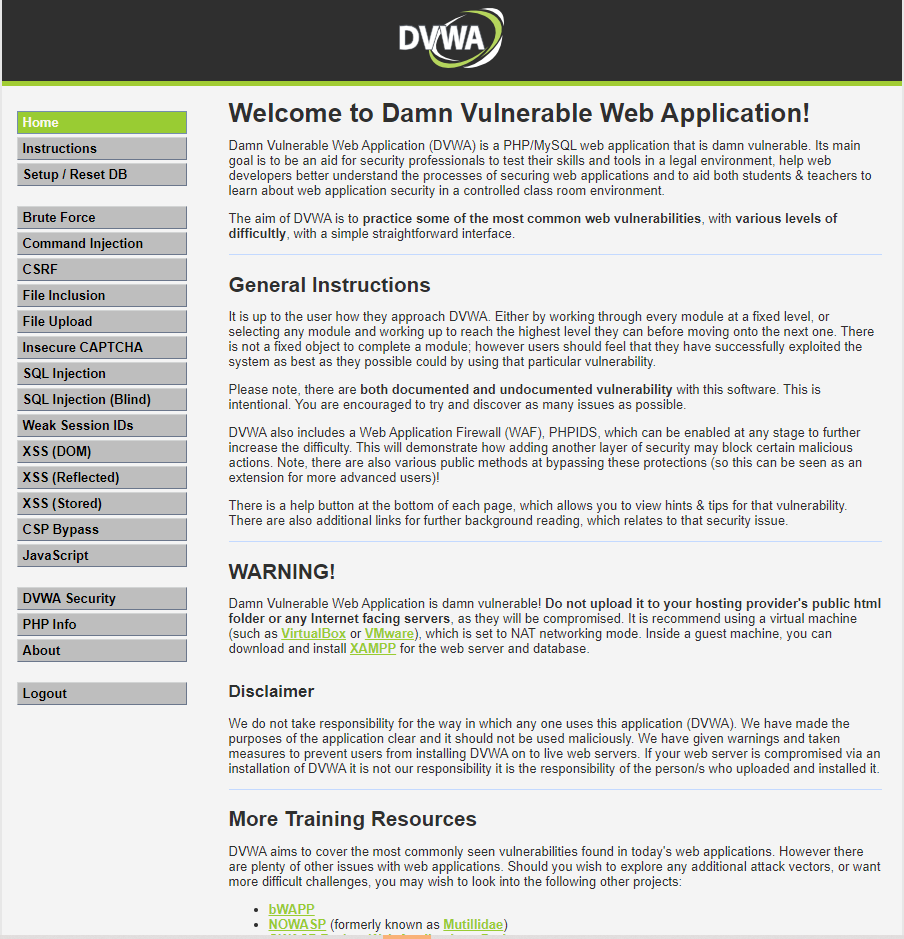
PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

10.0.0.5 : ok=6 changed=0 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0

10.0.0.6 : ok=6 changed=0 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0

10.0.0.7 : ok=6 changed=0 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0

**Note:** Make sure web servers are all turned on before running ansible. Also, web servers hosting the DVWA do not have a public IP, the public IP address of the load balancer is used to access the DVWA <<load\_balancer\_publicIP>>.



**Elk-Server Configuration**

This section covers the deploying of the Elk Server image using Ansible.

As stated before the Elk Stack combines three open source tools (ElasticSearch, Logstash, Kibana) allowing the storage of large amounts of data.

Logstash is a log aggregator that deals with the collection of data from all input sources and processes it to various supported output destinations.

ElasticSearch is a full-text search analysis engine, based on the Apache Lucene search engine that indexes and stores the data. Kibana is a visualization interface for querying and analyzing data. Also, Elk stack beats are open platforms used for single purpose data shippers to Logstash or Elasticsearch.

Steps to connect Elk server to the ansible container are as below:

Update **hosts** file in ansible, a separate section for Elk Server with the IP address for the Elk-server (10.1.0.4)

# Ex 2: A collection of hosts belonging to the 'webservers' group

[webservers]

10.0.0.5 ansible\_python\_interpreter=/usr/bin/python3

10.0.0.6 ansible\_python\_interpreter=/usr/bin/python3

10.0.0.7 ansible\_python\_interpreter=/usr/bin/python3

[elk]

10.1.0.4 ansible\_python\_interpreter=/usr/bin/python3

Create a separate ansible playbook to deploy and configure the Elk server, with the instructions below. The playbook addresses the configuration to run on the IP address added to the Elk section in the hosts file. It also includes the Elk Docker container configuration (sebp/elk:761) to provide the three components of the Elk Stack through a published port. Lastly, system requirement for running the Elk contontainer by increasing the memory. More information can be found on: <https://elk-docker.readthedocs.io/#prerequisites>

---

- name: Configure Elk VM with Docker

hosts: elk

remote\_user: RedAdmin

become: true

tasks:

# Use apt module

- name: Install docker.io

apt:

update\_cache: yes

force\_apt\_get: yes

name: docker.io

state: present

# Use apt module

- name: Install python3-pip

apt:

force\_apt\_get: yes

name: python3-pip

state: present

# Use pip module (It will default to pip3)

- name: Install Docker module

pip:

name: docker

state: present

# Use command module

- name: Increase virtual memory

command: sysctl -w vm.max\_map\_count=262144

# Use sysctl module

- name: Use more memory

sysctl:

name: vm.max\_map\_count

value: '264144'

state: present

reload: yes

# Use docker\_container module

- name: download and launch a docker elk container

docker\_container:

name: elk

image: sebp/elk:761

state: started

restart\_policy: always

# Please list the ports that ELK runs on

published\_ports:

- 5601:5601

- 9200:9200

- 5044:5044

# Use systemd module

- name: Enable service docker on boot

systemd:

name: docker

enabled: yes

Run the playbook to complete the deployment.

root@contianer\_ID:/etc/ansible# ansible-playbook elk\_playbook.yml

[WARNING]: ansible.utils.display.initialize\_locale has not been called, this may result in incorrectly calculated text

widths that can cause Display to print incorrect line lengths

PLAY [Configure Elk VM with Docker] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

TASK [Gathering Facts] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [10.1.0.4]

TASK [Install docker.io] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [10.1.0.4]

TASK [Install python3-pip] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [10.1.0.4]

TASK [Install Docker module] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [10.1.0.4]

TASK [Increase virtual memory] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [10.1.0.4]

TASK [Use more memory] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [10.1.0.4]

TASK [download and launch a docker elk container] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

[DEPRECATION WARNING]: The container\_default\_behavior option will change its default value from "compatibility" to

"no\_defaults" in community.docker 2.0.0. To remove this warning, please specify an explicit value for it now. This

feature will be removed from community.docker in version 2.0.0. Deprecation warnings can be disabled by setting

deprecation\_warnings=False in ansible.cfg.

ok: [10.1.0.4]

TASK [Enable service docker on boot] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

ok: [10.1.0.4]

PLAY RECAP \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

10.1.0.4 : ok=8 changed=0 unreachable=0 failed=0 skipped=0 rescued=0 ignored=0

**Target Machines and Beats**

An integrated Elk-server can be used to give the ability to aggregate logs from all systems. . Monitor systems to detect vulnerabilities, as well as find opportunities for improvement using beats. Beats are data shippers that gather data from sources to Elasticsearch.

* Filebeats and metricbeats have been employed to monitor machines. Filebeats monitors changes to file systems.
* Metricbeats can be used to monitor system metrics such as sudo escalation failures, CPU usage, attempted SSH logins, etc.

The playbook below installs filebeat on the target hosts.

**Filebeat Playbook**

---

- name: Installing and Launch Filebeat

hosts: webservers

become: yes

tasks:

- name: Download filebeat .deb file

command: curl -L -O https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-7.4.0-amd64.deb

- name: Install filebeat .deb

command: dpkg -i filebeat-7.4.0-amd64.deb

- name: Drop in filebeat.yml

copy:

src: /etc/ansible/files/filebeat-config.yml

dest: /etc/filebeat/filebeat.yml

- name: Enable and Configure System Module

command: filebeat modules enable system

- name: Setup filebeat

command: filebeat setup

- name: Start filebeat service

command: service filebeat start

- name: Enable service filebeat on boot

systemd:

name: filebeat

enabled: yes

Create a playbook for metricbeats. The playbook for installing metricbeat is not included, but looks essentially identical — simply replace filebeat with metricbeat, and it will work as expected. For reference refer to:

Run:

ansible-playbook <<filebeat\_playbook\_name>>.yml

ansible-playbook <<metricbeat\_playbook\_name>>.yml

Create a directory <<files>> in the ansible directory. The following commands below were used to download the configuration files for filebeats and metricbeats respectively.

curl https://gist.githubusercontent.com/slape/5cc350109583af6cbe577bbcc0710c93/raw/eca603b72586fbe148c11f9c87bf96a63cb25760/Filebeat > /etc/ansible/files/filebeat-config.yml

curl https://gist.githubusercontent.com/slape/58541585cc1886d2e26cd8be557ce04c/raw/0ce2c7e744c54513616966affb5e9d96f5e12f73/metricbeat

To verify Filebeat was correctly installed on the web servers use the following steps:

* Navigate to the ELK server using the IP address through port 5601:   
  curl http://<<Elk-server >>:5601/app/kibana

**Congratulations - you have completed the deployment of the Elk Stack monitoring project**