## Table des matières

Introduction and Goal .................................................................3

Installation of the Elastic Detector Vulnerability Scanner ...............4

Installation of the Elastic Stack SIEM ........................................5

1. Creation of the Virtual Machine (VM) ......................................5
2. Installation of Elastic Stack ....................................................5
3. Basic configuration of Elastic Stack .......................................6
4. Wrapping-up the installation of Elastic Stack ............................8

Configuring Elastic Detector .....................................................11

Configuring Elastic Stack .........................................................11

Conclusion ..................................................................................15

Further Reading ........................................................................16
Building your first SOC with Elastic Detector and Elastic Stack

Introduction and Goal:

The goal of this document is to explain how to build a first SOC (Security Operation Center) using Elastic Detector and the open source Elastic Stack (Elasticsearch, Kibana, Logstash, Beats).

Nowadays, organisations have a NOC (Network Operation Center) in order to monitor its network and to ensure the availability of the services. These NOCs are usually built on top of monitoring tools or solutions (for network, system, application performance and website) such as Nagios, Icinga, Centreon, Zabbix, ...

Once you monitor your network, servers and applications, the next step is to monitor security.

In order to do so, the very first thing is to know the security posture of your IT. Most part of organizations opt for vulnerability assessments, pentests and security audits done one-shot.

Because there were in average during 2015 more than 18 new vulnerabilities everyday and attacks are widespread, best practices prone for continuous vulnerability assessment and continuous log analysis.

A simple and efficient way to achieve this goal is to create a first SOC that could give you the overview of your security status and centralize the logs gathered across your IT. Your first SOC will comprise the two following tools:

- A continuous vulnerability assessment scanner
- A SIEM (System Information and Event Management)

The following sections will explain how to put in place such a first SOC using Elastic Detector (as a Vulnerability Scanner) and Elastic Stack (as a SIEM). We will install Elastic Detector Virtual Appliance in a VMware environment and build an Elastic Stack Virtual Machine in the same VMware environment. It is possible to do the same on public clouds such as AWS, Azure or GCE.
Installation of the Elastic Detector Vulnerability Scanner

1. Download Elastic Detector VM from SecludIT’s download server and deploy the OVF template.

2. Create your administration account and configure Elastic Detector to continuously monitor the VMware environment by creating a VMware vSphere connector with a vCenter service account.
Installion of the Elastic Stack SIEM

1. Creation of the Virtual Machine (VM)
Create a virtual Machine with 4GB RAM and 4 CPUs and 80 GB hard drive.
Download Debian Jessie netinstall and install a basic Debian system.

2. Installation of Elastic Stack
Follow instructions on Elastic web site in order to install:
   - Elasticsearch (current release 2.3.4)
   - Logstash (current release 2.3.4)
   - Kibana (current release 4.5.3)

Add Elastic GPG key using the following command line:
wget -qO - https://packages.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -

Add Elasticsearch, Logstash and Kibana repositories using the following command line:
echo "deb http://packages.elastic.co/elasticsearch/2.x/debian stable main" | sudo tee -a /etc/apt/sources.list.d/elasticsearch-2.x.list
echo "deb http://packages.elastic.co/logstash/2.3/debian stable main" | sudo tee -a /etc/apt/sources.list.d/logstash-2.3.list
echo "deb http://packages.elastic.co/kibana/4.5/debian stable main" | sudo tee -a /etc/apt/sources.list.d/kibana-4.5.list

Update package list using the following command:
apt-get update

Install Elasticsearch, Logstash, Kibana using the following command line:
apt-get install elasticsearch logstash kibana

Install Marvel or Elastic-HQ for better visualization, using the following command line:
/usr/share/elasticsearch/bin/plugin install license
/opt/kibana/bin/kibana plugin --install elasticsearch/marvel/latest

NB:
   - Depending on the version you might have to fix some permission for Kibana to start properly. For instance, I have to fix it using the following command line:
     chown kibana:root /opt/kibana/optimize/.babelcache.json
   - Depending on the version you might have to fix some permission for Logstash to start properly. For instance, I have to fix it using the following command line:
     usermod -G adm logstash
3. Basic configuration of Elastic Stack

Configure Elasticsearch by editing its main configuration file "/etc/elasticsearch/elasticsearch.yml" and set the parameters according to your deployment and your needs.

Here is our sample configuration file:

```yaml
cluster.name: SecludIT-SIEM
node.name: ELK-Kibana
node.master: true
node.data: true
path.data: /var/lib/elasticsearch
path.log: /var/log/elasticsearch
bootstrap.mlockall: true
network.host: 192.168.13.146
```

Configure Kibana by editing its main configuration file "/opt/kibana/config/kibana.yml" and set the parameters according to your deployment and your needs.

Here is our sample configuration file:

```yaml
```

Configure Logstash by adding basic filter in its configuration directory "/etc/logstash/conf.d/".

Here is our sample configuration file for the input:

```ruby
input {
  file {
    path => [ "/var/log/auth.log" ]
    type => "syslog"
  }
}
```

Here is our sample configuration file for the output:

```ruby
output {
  elasticsearch { hosts => "192.168.13.146" }
  stdout { codec => rubydebug }
}
```

Here is a sample status of the Elastic Stack installation.
Status: Green

- Heap Total (MB): 84.47
- Heap Used (MB): 77.84
- Load: 0.06, 0.04, 0.05
- Response Time Avg (ms): 11.72
- Response Time Max (ms): 82.08
- Requests Per Second: 0.85

Installed Plugins

- plugin.litmus
- plugin.elasticsearch
- plugin.maven
- plugin.ssh
- plugin.xs
- plugin.xs
- plugin.ys
- plugin.zs

Marvel - Your Elasticsearch Monitor

<table>
<thead>
<tr>
<th>Name</th>
<th>Nodes</th>
<th>Indices</th>
<th>License</th>
</tr>
</thead>
<tbody>
<tr>
<td>SecludIT-SIEM</td>
<td>1</td>
<td>4</td>
<td>Trial</td>
</tr>
</tbody>
</table>
4. Wrapping-up the installation of Elastic Stack

Marvel is a powerful plugin but it requires a license. You can use Elastic-HQ instead:

- [http://www.elastichq.org/](http://www.elastichq.org/)

In order to install Elastic-HQ, you can follow the instructions on Github that can be found here:

- [https://github.com/royrusso/elasticsearch-HQ](https://github.com/royrusso/elasticsearch-HQ)

In our case we choose not to install it as a plugin but as a separate website, so that we can better control its security without impacting Elastic Search.

In order to do so, please do as follows:

- Download the website archive from Github
- Unzip it to “/opt/” directory
  ```
  cd /opt/; unzip /root/royrusso-elasticsearch-HQ-v2.0.3-2-gfe18bc4.zip
  ```
- Install nginx web server using debian package manager, as follows
  ```
  apt-get install nginx
  ```
- Configure nginx and restart it

Here is our basic configuration for nginx that we put in “/etc/nginx/sites-available/elastic-HQ” file.

```
server {
  listen 8080;
  server_name 192.168.13.146;
  root /opt/royrusso-elasticsearch-HQ-fe18bc4/;
}
```

Update nginx configuration to enable this new site, as follow:

```bash
ln -s /etc/nginx/sites-available/elastic-HQ /etc/nginx/sites-enabled/elastic-HQ
```
In order to be able to connect to Elasticsearch from Elastic-HQ, you need to enable “CORS” in Elasticsearch. This can be done by adding the following directives at the end of Elasticsearch configuration file and restarting Elasticsearch.

http.cors.enabled: true
http.cors.allow-origin: "/.*"
Everything, has been configured and Elastic Stack is now operational. For instance, you can visualize that logs are properly send to the Stack, by accessing Kibana web interface on default port 5601.

For instance you can monitor SSH access to your platform to detect invalid SSH access or SSH brute force attack as shown below.
NB: Additional steps are required to secure the installation, such as:
- Using nginx as reverse proxy for serving Kibana
- Adding a secure communication channel for the web site in nginx
- Adding user authentication for web site in nginx
- Adjusting firewall rules to avoid any unwanted access to Elastic Stack REST API or web site

Configuring Elastic Detector

Elastic Detector logs can be sent to Elastic Stack in order to create dashboards for monitoring IT security. Therefore, any security event or alert raised internally by Elastic Detector can be shown in the Elastic Stack dashboard.

In order to do so, we just have to configure the logger of Elastic Detector to redirect logs to Elastic Stack. Elastic Detector is using a centralized log system based on syslog-ng.

Here is a the sample configuration we put in place to redirect “auth” logs to Elastic Stack (configuration file “/etc/syslog-ng/syslog-ng.conf”).

```bash
# SecludIT: Elastic Stack syslog log redirection
destination d_elastic_stack_syslog { udp("192.168.13.146" port(514)); }; log { source(s_src); filter(f_auth); destination(d_elastic_stack_syslog); };
```

Using the same method we can redirect internal logs of Elastic Detector to Elastic Stack in order to send security information and be able to track security events in Elastic Stack.

Configuring Elastic Stack

Elastic Stack logging system can be configured to accept logs from others sources and can redirect them in a specific file. We choose to configure “rsyslogd” daemon instead of adding a new input in logstash configuration, because we can then backup the log file in its raw format (without any modification).

Here is the sample configuration we put in place in file “/etc/rsyslog.d/elastic-detector.conf”:

```bash
########## FILTER ######
if $fromhost-ip startswith '192.168.13.43' then /var/log/elastic-detector/ED-192.168.13.43.syslog.log
& ~
```
Once we have Elastic Detector logs, we can add it to logstash in order to show information and event in Elastic Stack and especially Kibana.

To do so, we have created an input configuration for Logstash with the following content:

```ruby
input {
  file {
    path => [ "/var/log/elastic-detector/ED-192.168.13.43-syslog.log" ]
    type => "syslog"
  }
}
```

For instance you can monitor SSH access to any of your platform to detect invalid SSH access or SSH brute force attack as shown below.

Here is a sample syslog filter for parsing “auth” log for SSH login attempt:

```ruby
# Filtering for SSH logins either failed or successful
filter {
  if [type] == "syslog" {
    if [syslog_program] == "sshd" {
      if "Failed password" in [message] {
        grok {
          break_on_match => false
          match => [
            "message", "invalid user %{DATA:UserName} from %{IP:src_ip}"
          ]
        }
        mutate {
          add_tag => [ "SSH_Failed_Login" ]
        }
      }
      if "Accepted password" in [message] {
        grok {
          match => [
            "message", "for %{DATA:UserName} from %{IP:src_ip}" ]
        }
        mutate {
          add_tag => [ "SSH_Successful_Login" ]
        }
      }
      if "Too many authentication failures for invalid user" in [message] {
        grok {
          match => [
            "message", "for invalid user %{DATA:UserName} from %{IP:src_ip}" ]
        }
        mutate {
          add_tag => [ "SSH_BruteForce_Attempt" ]
        }
      }
    }
  }
  else if "Too many authentication failures for" in [message] {
    grok {
      match => [
        "message", "for %{DATA:UserName} from %{IP:src_ip}" ]
    }
    mutate {
      add_tag => [ "SSH_Erronous_Password" ]
    }
  }
}
```
We then have created a dashboard for reporting SSH connection attempts.

If we want to forward Nagios log to Elastic Stack, we have to put in place a specific parsing as Nagios format is different from syslog format.

Here is a sample configuration that can be put in place:

```
# SecludIT: Elastic Stack nagios log redirection
source s_nagios { file("/var/log/nagios3/nagios.log"); };
destination d_elastic_stack_nagios { udp("192.168.13.146" port(5514)); };
log { source(s_nagios); destination(d_elastic_stack_nagios); };
```

As Elastic Detector is using Nagios logging internally, we can retrieve Nagios logs and events by creating a new input in Logstash and parsing the Nagios logs.

In order to create an input in Logstash we have added the following configuration:

```
input { 
  udp { 
    port => 5514 
    type => "nagios" 
  } 
}
```

We have also added a Logstash filter to properly parse Nagios logs. Here is the Logstash filter:

```
filter { 
  if [type] == "nagios" { 
    grok { 
      match => { "message" => "%{NAGIOSLOGLINE}" } 
    } 
    date { 
      locale => en 
      match => ["nagios_epoch", "UNIX"] 
    } 
  } 
}
```
We have created an additional nagios filter to properly report the AutoDiscovery functionality of Elastic Detector.

Here is the sample nagios filter:

```
# Filtering Elastic Detector Auto-Discovery
filter {
  if [type] == "nagios" {
    if [nagios_type] == "SERVICE NOTIFICATION" {
      if "Found new Server" in [message] {
        grok {
          match => [ "message", "Found new Server: %{DATA:ServerID}" ]
        }
        mutate {
          add_tag => [ "Server_Added" ]
        }
      } else if "Found new Firewall Policy" in [message] {
        grok {
          match => [ "message", "Found new Firewall Policy: %{DATA:FWPolicyID}" ]
        }
        mutate {
          add_tag => [ "FWPolicy_Added" ]
        }
      } else if "Server Terminated" in [message] {
        grok {
          match => [ "message", "Server Terminated: %{DATA:ServerID}" ]
        }
        mutate {
          add_tag => [ "Server_Removed" ]
        }
      } else if "Firewall Policy Removed" in [message] {
        grok {
          match => [ "message", "Firewall Policy Removed: %{DATA:FWPolicyID}" ]
        }
      }
    }
  }
}
```
We have then created a specific Dashboard for Elastic Detector and its AutoDiscovery module.

Conclusion

Elastic Detector embeds a syslog server and is using standard logging format, thus making it easy to integrate in any SIEM such as Elastic Stack or other commercial product such as IBM QRadar, HP ArcSight, Splunk, AlienVault OSSIM, etc…

We have showed how you can create your first SOC using Elastic Detector and Elastic Stack. You can centralize your logs in Elastic Stack and build any dashboard according to Key Indicators that are the most relevant to you and your organization. Elastic Detector brings security information and events that you can filter in Elastic Stack in order to build the dashboard with your own Security Key Indicators.
Further Reading

- SOC Definition:

- NOC Definition:

- SIEM Definition:

- Elastic Stack:
  - https://www.elastic.co

- Elastic Detector:
  - https://secludit.com