

# Lab Exercises: Deploying, Managing, and Leveraging Honeypots in the Enterprise using Open Source Tools

Fill in the details of your MHN Server info. If you don't have this, ask your instructor. These details will be used throughout the rest of the lab.

MHN Server	domain: root password:
Honeypot Server	ip: root password:

MHN Webapp Credentials. You Choose these.

email address	
password	

## Exercise 1: Deploy and Configure Modern Honey Network

In this lab we are going to deploy MHN and configure it to use HTTPS. This lab should take 15-30 minutes total.

### Steps:

1. Login to your MHN Server using ssh.

```
ssh root@<mhn-server-domain-name>
```

2. Then run these commands:

```
cd /opt/  
git clone https://github.com/threatstream/mhn.git  
cd mhn/  
./install.sh
```

You will see lines of log output and a few prompts.

3. If prompted regarding submitting package stats to the golang repos, select <No>.

4. When you see “ MHN Configuration”, answer the questions in the following ways. Notice the https and :8443 on some of the inputs. These are important. Use the information from the tables at the top of this page to fill these in. <ENTER> means no input required, press ENTER to use the defaults.

```
Do you wish to run in Debug mode?: y
Superuser email: YOUR-EMAIL-ADDRESS
Superuser password: A PASSWORD OF YOUR CHOOSING
Server base url ["http://104.236.42.232"]:
https://<mhn-server-domain-name>
Honeymap url ["https://mhn-server-domain-name:3000"]:
https://<mhn-server-domain-name>:8443
Mail server address ["localhost"]: <ENTER>
Mail server port [25]: <ENTER>
Use TLS for email?: y/n y
Use SSL for email?: y/n y
Mail server username [""]: <ENTER>
Mail server password [""]: <ENTER>
Mail default sender [""]: <ENTER>
Path for log file ["/var/log/mhn/mhn.log"]: <ENTER>
```

After this, the installation will start to download and load snort rules from emerging threats. Be patient. This can take several minutes. Wait until you are prompted again.

5. Do not integrate with Splunk or ELK. We are going to do these steps later.

```
Would you like to integrate with Splunk? (y/n) n
...
Would you like to install ELK? (y/n) n
```

When finished, you should see a message like this:

```
[Wed Jul 29 21:27:52 EDT 2015] Completed Installation of all MHN packages
```

6. Lastly, run these two commands to fix a bug that has not been fixed yet:

```
chown www-data /var/log/mhn/mhn.log
supervisorctl restart mhn-celery-worker
```

You should see this:

```
mhn-celery-worker: ERROR (not running)
mhn-celery-worker: started
```

Now, run this command as a sanity check.

```
supervisorctl status
```

All processes should be RUNNING. Example:

geoloc	RUNNING	pid 31721, uptime 0:10:00
honeymap	RUNNING	pid 31722, uptime 0:10:00
hpfeeds-broker	RUNNING	pid 12916, uptime 0:12:36
mhn-celery-beat	RUNNING	pid 691, uptime 0:01:17
mhn-celery-worker	RUNNING	pid 782, uptime 0:00:33
mhn-collector	RUNNING	pid 685, uptime 0:01:17
mhn-uwsgi	RUNNING	pid 683, uptime 0:01:17
mnemosyne	RUNNING	pid 30843, uptime 0:10:35

## 5. Configure MHN to use HTTPS

This mhn server image you are using has some pre-deployed nginx config files. We need to enable these now that MHN is installed. Run these commands:

```
cd /etc/nginx/sites-enabled
ln -s /etc/nginx/sites-available/mhn-https
ln -s /etc/nginx/sites-available/honeymap-https
rm /etc/nginx/sites-enabled/default
/etc/init.d/nginx restart
```

Now, as a sanity check, run this command.

```
netstat -luntp | grep nginx
```

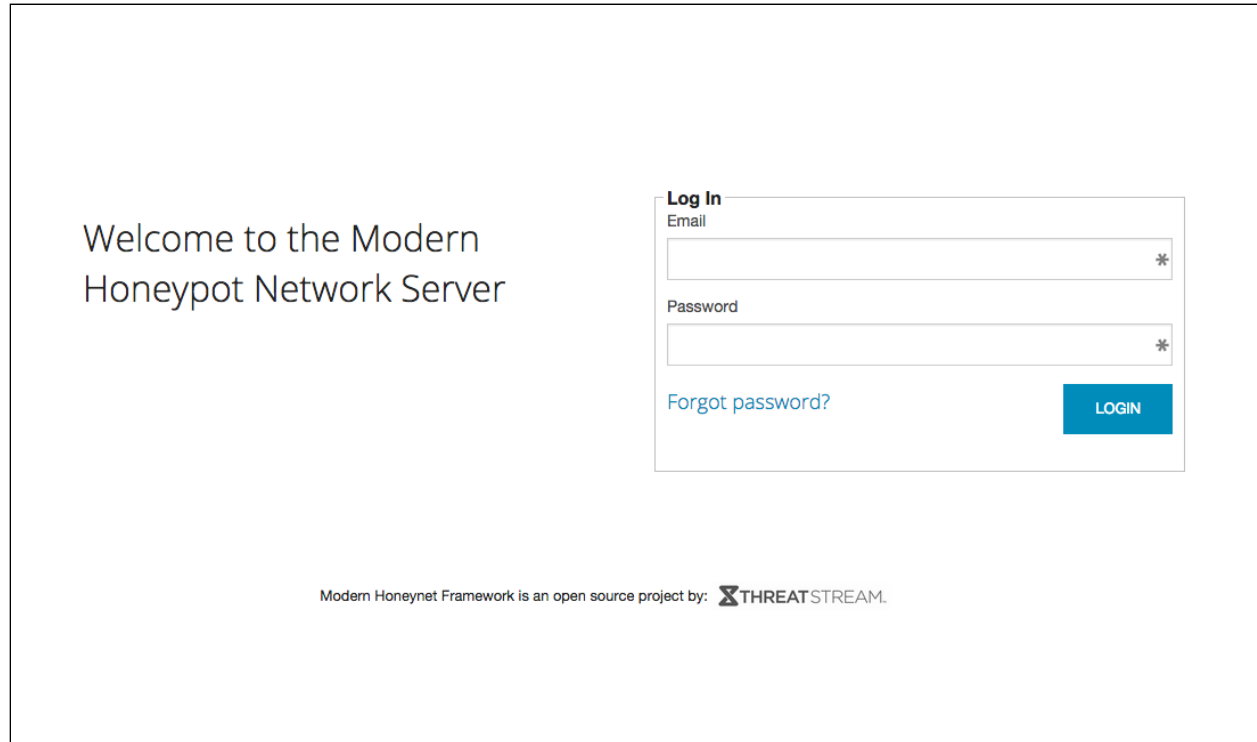
You should get output like this:


tcp	0	0 0.0.0.0:80	0.0.0.0:*	LISTEN	15766/nginx
tcp	0	0 0.0.0.0:8443	0.0.0.0:*	LISTEN	15766/nginx
tcp	0	0 0.0.0.0:443	0.0.0.0:*	LISTEN	15766/nginx
tcp	0	0 0.0.0.0:8001	0.0.0.0:*	LISTEN	15766/nginx

Now, open a web browser and visit:

**https://<your-mhn-server-domain-name>**

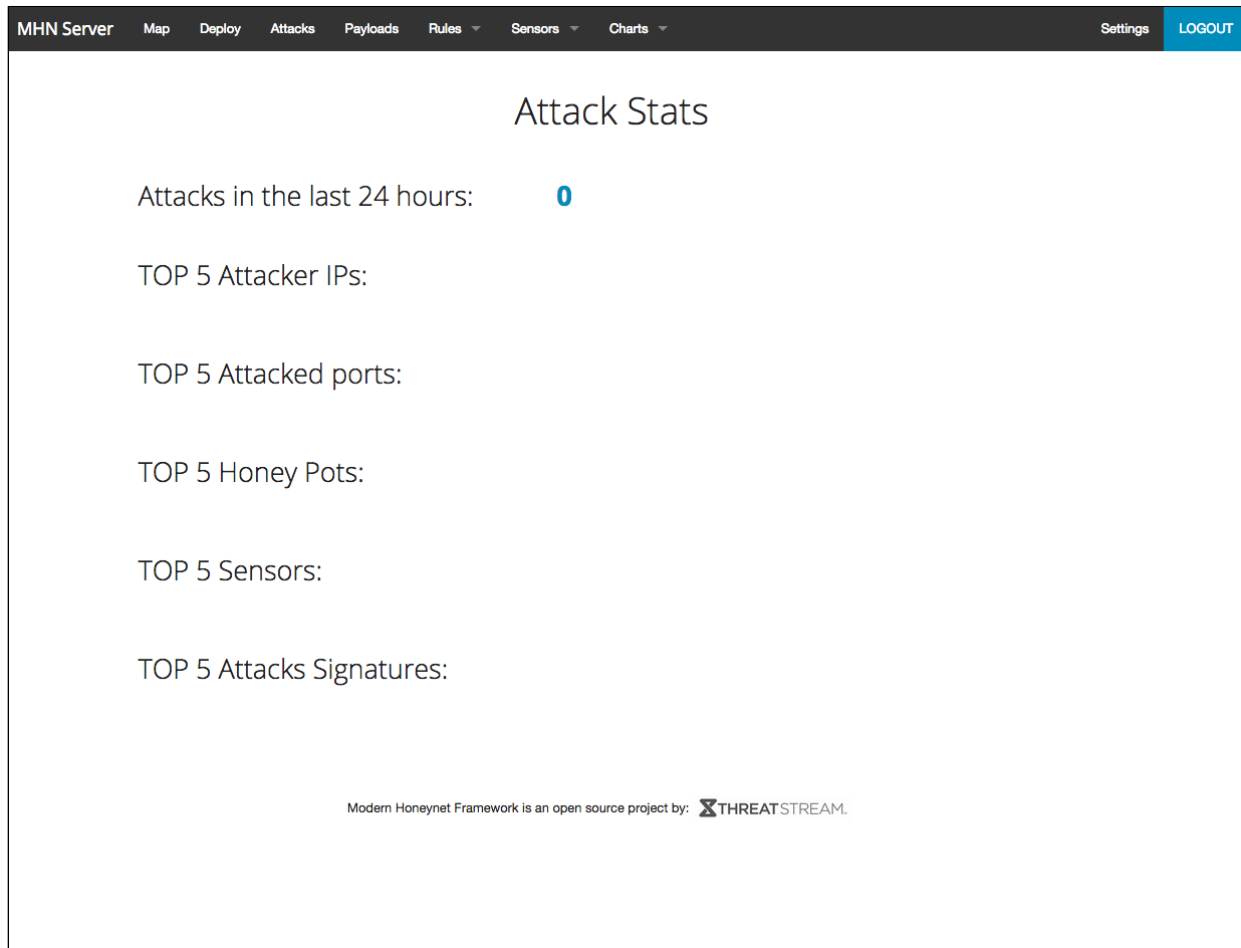
You should see a page like this:



The screenshot shows a web browser window displaying the login page for the Modern HoneyNet Network Server. On the left, the text "Welcome to the Modern HoneyNet Network Server" is displayed. On the right, there is a "Log In" form with two input fields: "Email" and "Password", each followed by an asterisk (\*). Below the "Password" field is a link that says "Forgot password?". At the bottom right of the form is a blue button labeled "LOGIN". At the bottom center of the page, there is a line of text: "Modern HoneyNet Framework is an open source project by:  THREATSTREAM."

Login with email address and password you provided during the MHN install (and that you hopefully wrote down at the beginning of the lab or you remember).

After logging in, you should see the MHN Server dashboard. It looks like this.



Click around the various tabs and explore the UI. There will be no data yet so expect all the tables and charts to be blank.

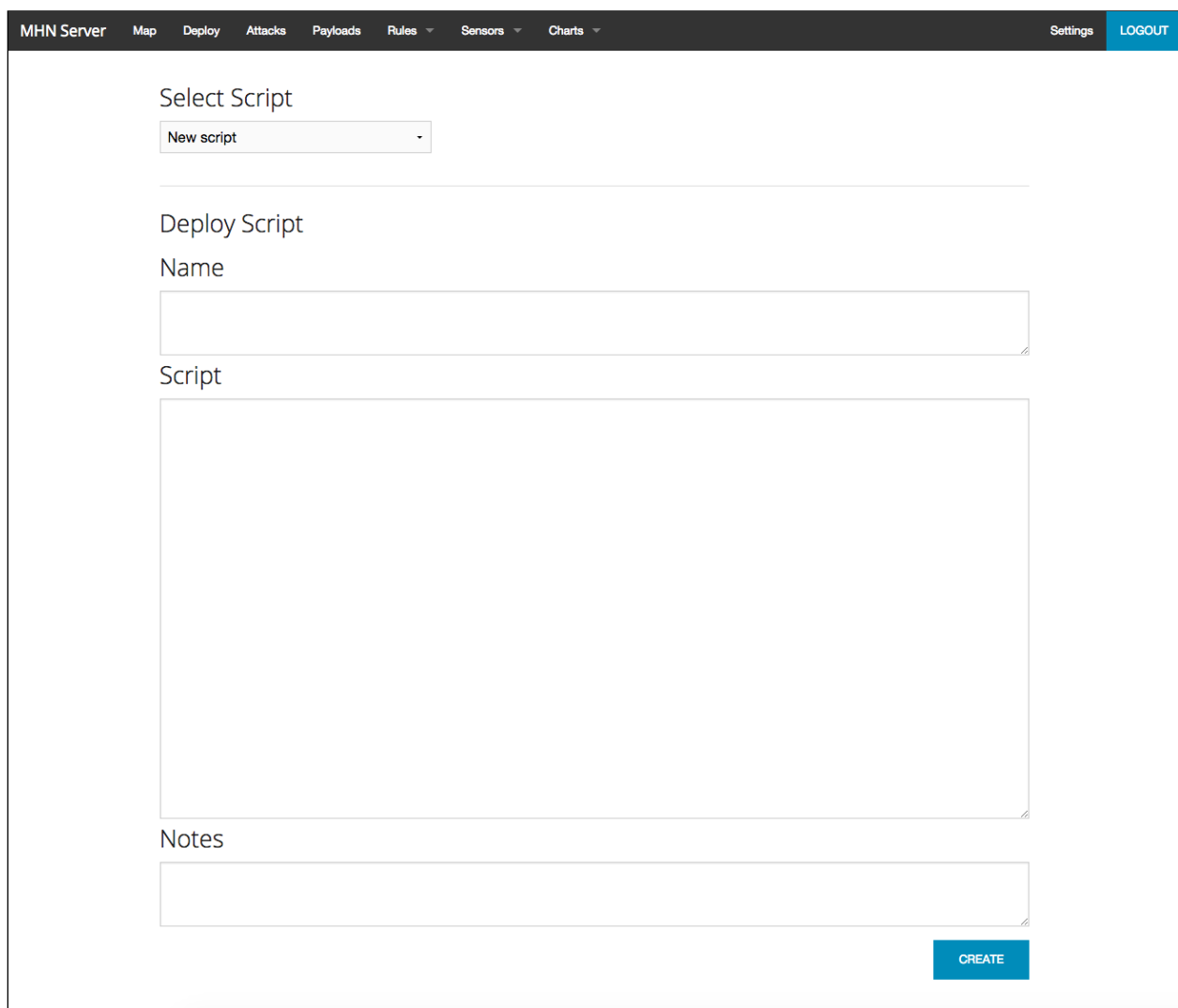
- **Map** - Realtime map visualization of events coming from various honeypots and sensors.
- **Deploy** - Deployment management page
- **Attacks** - Simple UI for exploring the Attacks data from various honeypots and sensors.
- **Payloads** - Simple UI for viewing some of the payload data
- **Rules** - Simple UI for exploring viewing and managing the snort rules MHN deploys.
- **Sensors** - Simple UI for viewing the installed sensors.
- **Charts** - Some charts and analytics about Kippo data (if Kippo is deployed)
- **Settings** - A page for adding new users and getting your API key.

## Exercise 2: Deploy Honeypots (Dionaea + Kippo + p0f + Snort)

Now that you have your MHN server up and running, it is time to deploy some sensors. In this exercise we are going to install several honeypots and network sensors to the same server. Several packages can be co-deployed together and provide useful data. This exercise should take 15-30 minutes.

### Steps:

1. Visit the “Deploy” tab of your MHN Web app. You should see a page like this:

The screenshot shows the 'Deploy' tab of the MHN Web app. The top navigation bar includes 'MHN Server', 'Map', 'Deploy', 'Attacks', 'Payloads', 'Rules', 'Sensors', 'Charts', 'Settings', and 'LOGOUT'. The main content area is titled 'Select Script' and features a dropdown menu currently set to 'New script'. Below this is a 'Deploy Script' section with three input fields: 'Name', 'Script', and 'Notes'. Each field has a small icon in the bottom right corner, likely for pasting or clearing. A blue 'CREATE' button is positioned at the bottom right of the form.

2. Look in the “**Select Script**” drop down menu and select “**Ubuntu - Dionaea**”. Now copy the deploy command. It should look something like this:

```
wget "https://villian.mhn-server.com/api/script/?text=true&script_id=12" -O
deploy.sh && sudo bash deploy.sh https://villian.mhn-server.com
XXXXXXXXXX
```

### 3. Login to your honeypot server using SSH

```
ssh root@<your-honeypot-ip>
```

### 4. paste the deploy command, press <ENTER> and wait for it to finish.

NOTE. You may be prompted by patch with this. If so, input the lines seen in blue.

```
Ignoring potentially dangerous file name /etc/dionaea/dionaea.conf
can't find file to patch at input line 3
Perhaps you used the wrong -p or --strip option?
The text leading up to this was:
-----
|--- /etc/dionaea/dionaea.conf
|+++ /etc/dionaea/dionaea.conf.new
-----
File to patch: /etc/dionaea/dionaea.conf
patching file /etc/dionaea/dionaea.conf
Ignoring potentially dangerous file name /usr/lib/dionaea/python/dionaea/ihandlers.py
can't find file to patch at input line 55
Perhaps you used the wrong -p or --strip option?
The text leading up to this was:
-----
|
|--- /usr/lib/dionaea/python/dionaea/ihandlers.py
|+++ /usr/lib/dionaea/python/dionaea/ihandlers.py.new
-----
File to patch: /usr/lib/dionaea/python/dionaea/ihandlers.py
```


### 5. After it finishes run this command as a sanity check:


```
supervisorctl status
```

You should see dionaea RUNNING like this

```
dionaea                                RUNNING      pid 3441, uptime 0:06:15
```

In your MHN web app, visit the “Sensors” tab. You should now see a sensor appear. It should look something like this (your honey name, IP, and UUID will be different).

MHN Server   Map   Deploy   Attacks   Payloads   Rules   Sensors   Charts   Settings   LOGOUT						
Sensors						
	Name	Hostname	IP	Honeypot	UUID	Attacks
1- 	<input type="text" value="loki-honeypot-dionaea"/>	loki-honeypot	104.131.119.192	dionaea	2e341022-377a-11e5-be94-040161165601	0

Modern Honeynet Framework is an open source project by:  THREATSTREAM.

6. Now, return to the “**Deploy**” tab and follow the same procedure with “**Ubuntu - Snort**”. After it finishes run this command as a sanity check: `supervisorctl status`

You should see dionaea and snort RUNNING like this:

```
dionaea                RUNNING      pid 3441, uptime 0:06:15
snort                  RUNNING      pid 27439, uptime 0:00:06
```

Another sensor should appear in your “**Sensors**” page.

7. Now follow the same procedure with “**Ubuntu - p0f**”. After it finishes run this command as a sanity check: `supervisorctl status`

You should see dionaea, snort, and p0f RUNNING like this:

```
dionaea                RUNNING      pid 3441, uptime 0:07:22
p0f                    RUNNING      pid 27583, uptime 0:00:08
snort                  RUNNING      pid 27439, uptime 0:01:13
```

Another sensor should appear in your “**Sensors**” page.






8. Now follow the same procedure with “**Ubuntu - Kippo**”. **NOTE:** after this is run, **IF YOU LOGOUT** of your honeypot, you will have to use a different SSH command to log back in (`ssh root@<your-honeypot-ip> -p 2222`). After it finishes run this command as a sanity check: `supervisorctl status`

You should see dionaea, snort, kippo, and p0f RUNNING like this:



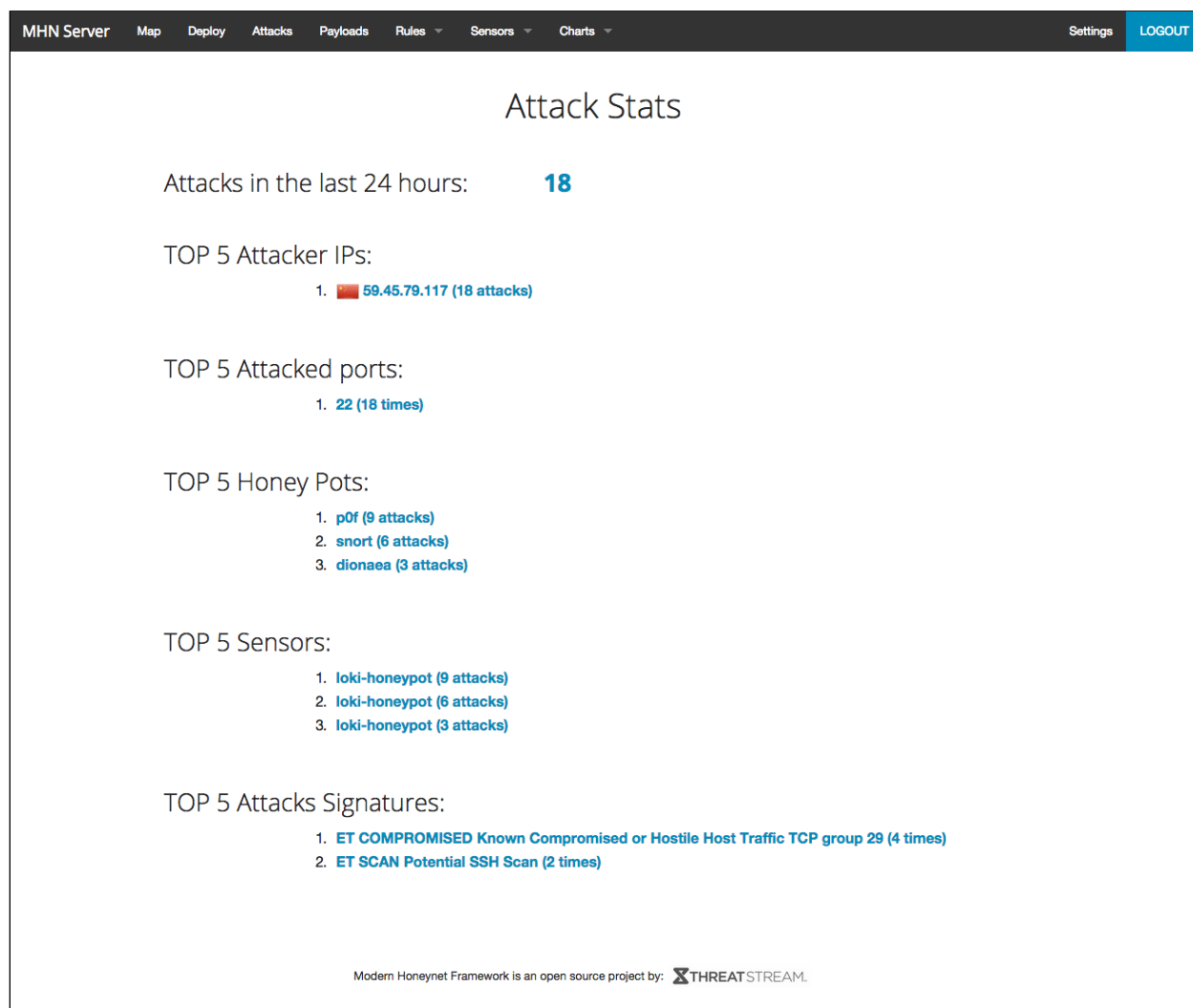
```
dionaea                RUNNING      pid 3441, uptime 0:10:23
kippo                  RUNNING      pid 31186, uptime 0:00:05
p0f                    RUNNING      pid 27583, uptime 0:03:09
snort                   RUNNING      pid 27439, uptime 0:04:14
```

Another sensor should appear in your “**Sensors**” page. It should look something like this.

MHN Server   Map   Deploy   Attacks   Payloads   Rules   Sensors   Charts   Settings   LOGOUT						
Sensors						
	Name	Hostname	IP	Honeypot	UUID	Attacks
1- 	<input type="text" value="loki-honeypot-dionaea"/>	loki-honeypot	104.131.119.192	dionaea	2e341022-377a-11e5-be94-040161165601	3
2- 	<input type="text" value="loki-honeypot-snort"/>	loki-honeypot	104.131.119.192	snort	c0892ac0-377a-11e5-be94-040161165601	6
3- 	<input type="text" value="loki-honeypot-p0f"/>	loki-honeypot	104.131.119.192	p0f	58af9d20-377b-11e5-be94-040161165601	9
4- 	<input type="text" value="loki-honeypot-kippo"/>	loki-honeypot	104.131.119.192	kippo	be1993f0-377b-11e5-be94-040161165601	0
Modern Honeynet Framework is an open source project by:  THREATSTREAM.						

Congratulations, you should now have a fully functional honeypot with additional network sensors deployed and collecting data. By now, you may even have events from your honeypot

being scanned. Visit the “MHN Server” tab (dashboard). Hopefully it has some events and should look something like this:



## Exercise 3: Integrate with Splunk

In this exercise we are going to integrate our MHN app with Splunk so we could search and explore the honeypot data using the open source MHN Splunk app.

### Steps:

1. Run the following commands **on your MHN Server** (not on the honeypot server).

```
cd /opt/mhn/scripts/  
./install_hpfeeds-logger-splunk.sh  
./install_splunk_universalforwarder.sh 127.0.0.1 9997
```

When prompted, enter the following information.

```
ERROR: The mgmt port [8089] is already bound.  Splunk needs to use  
this port.
```

```
Would you like to change ports? [y/n]: y
```

```
Enter a new mgmt port: 8091
```

2. Login to your splunk webapp. Splunk is preconfigured on your MHN server. Here are the credentials.

username	admin
password	MHNtR@in1ng (Note: that is a one, not an L)

Make sure you can login. **<https://<your-mhn-server-domain-name>:8001/>**

If you set everything up properly you should have some data in splunk already. Click the “Search and Reporting” link on the right side. And then type “\*” into the search box and press <ENTER>. You should see something like this:

**splunk** | App: Search & Reporting ▾ | Administrator ▾ | Messages ▾ | Settings ▾ | Activity ▾ | Help ▾ | Find

---

Search | Pivot | Reports | Alerts | Dashboards

**Search & Reporting**

---

## New Search

[Save As ▾](#) | [Close](#)

All time ▾

✓ 751 events (before 7/31/15 8:12:28.000 AM)

Events (751) | Patterns | Statistics | Visualization

Job ▾ || ■ ↗ ⬇ ⚙ Smart Mode ▾

Format Timeline ▾    — Zoom Out    + Zoom to Selection    × Deselect    1 month per column

List ▾   Format ▾   20 Per Page ▾

< Hide Fields	All Fields	i	Time	Event
		>	7/31/15 12:09:17.241 PM	2015-07-31T12:09:17.241024 src="80.82.79.104", direction="inbound", protocol="ip", ids_type="network", dionaea_action="reject", type="dionaea.connections", app="dionaea", dest="104.131.119.192", vendor_product="Dionaea", dest_port="8118", signature="Connection to Honeybot", src_port="42948", sensor="2e341022-377a-11e5-be94-040161165601", transport="tcp", severity="high"
		host = loki.mhn-server.com   source = /var/log/mhn/mhn-splunk.log   sourcetype = mhn-splunk-too_small		
		>	7/31/15 12:09:17.202 PM	2015-07-31T12:09:17.202570 direction="inbound", protocol="ip", ids_type="network", dest="104.131.119.192", app="p0f", transport="tcp", dest_port="8118", src="80.82.79.104", src_port="42948", severity="informational", vendor_product="p0f", type="p0f.events", p0f_os="???", signature="Packet Observed by p0f", sensor="58af9d20-377b-11e5-be94-040161165601"
		host = loki.mhn-server.com   source = /var/log/mhn/mhn-splunk.log   sourcetype = mhn-splunk-too_small		
		>	7/31/15 8:12:27.381 AM	[2015-07-31 08:12:27,381: WARNING/Worker-2] Imported 2000 rules so far...
		host = loki.mhn-server.com   source = /var/log/mhn/mhn-celery-worker.err   sourcetype = err-3		
		>	7/31/15 8:12:25.104 AM	[2015-07-31 08:12:25,104: WARNING/Worker-2] Imported 1500 rules so far...
		host = loki.mhn-server.com   source = /var/log/mhn/mhn-celery-worker.err   sourcetype = err-3		
		>	7/31/15 8:12:23.035 AM	[2015-07-31 08:12:23,035: WARNING/Worker-2] Imported 1000 rules so far...
		host = loki.mhn-server.com   source = /var/log/mhn/mhn-celery-worker.err   sourcetype = err-3		
		>	7/31/15 8:12:22.000 AM	[pid: 683][app: 0][req: 32/32] 37.252.238.106 () {34 vars in 730 bytes} [Fri Jul 31 08:12:22 2015] GET /ui/login/?next=%F => generated 2796 bytes in 8 msec (HTTP/1.0 200) 3 headers in 425 bytes (1 switches on core 0)
		host = loki.mhn-server.com   source = /var/log/mhn/mhn-uwsgi.err   sourcetype = err-too_small		
		>	7/31/15 8:12:21.018 AM	[2015-07-31 08:12:21,018: WARNING/Worker-2] Imported 500 rules so far...
		host = loki.mhn-server.com   source = /var/log/mhn/mhn-celery-worker.err   sourcetype = err-3		
		>	7/31/15 8:12:21.000 AM	[pid: 683][app: 0][req: 32/32] 37.252.238.106 () {32 vars in 438 bytes} [Fri Jul 31 08:12:21 2015] GET / => generated 245 bytes in 26 msec (HTTP/1.0 302) 4 headers in 415 bytes (1 switches on core 0)
		host = loki.mhn-server.com   source = /var/log/mhn/mhn-uwsgi.err   sourcetype = err-too_small		
		>	7/31/15 8:12:19.005 AM	2015-07-31 08:12:19,005 - /opt/mhn/server/mhn/tasks/rules.py - Bulk importing 17929 rules.
		host = loki.mhn-server.com   source = /var/log/mhn/mhn.log   sourcetype = mhn-too_small		
		>	7/31/15 8:12:19.005 AM	[2015-07-31 08:12:19,005: INFO/Worker-2] Bulk importing 17929 rules.
		host = loki.mhn-server.com   source = /var/log/mhn/mhn-celery-worker.err   sourcetype = err-3		
		>	7/31/15 8:12:19.005 AM	2015-07-31 08:12:19,005 - /opt/mhn/server/mhn/tasks/rules.py - Bulk importing 17929 rules.
		host = loki.mhn-server.com   source = /var/log/mhn/mhn-celery-worker.err   sourcetype = err-3		
		>	7/31/15 8:12:19.000 AM	INFO in rules [/opt/mhn/server/mhn/tasks/rules.py:76]: Bulk importing 17929 rules.
		host = loki.mhn-server.com   source = /var/log/mhn/mhn-celery-worker.err   sourcetype = err-3		

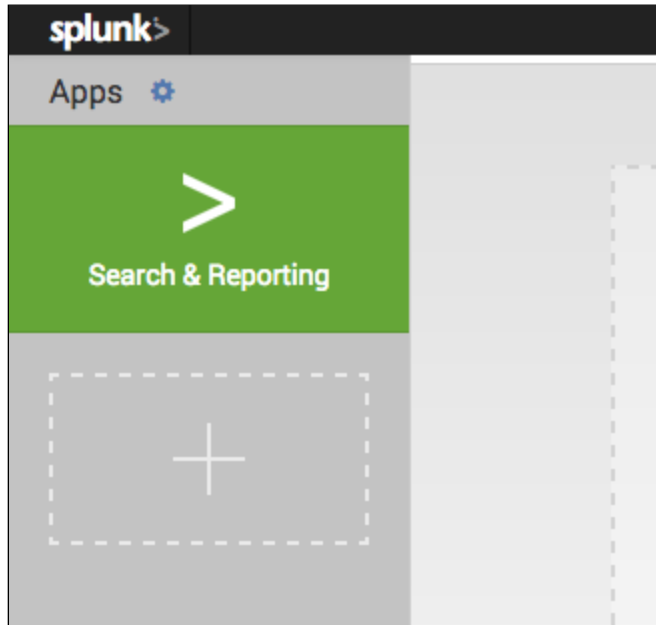
Selected Fields

- a host 1
- a source 13
- a sourcetype 7

Interesting Fields

- # date\_hour 4
- # date\_mday 2
- # date\_minute 47
- # date\_month 1
- # date\_second 59
- # date\_wday 2
- # date\_year 2
- # date\_zone 2
- # index 1
- # linecount 9
- # punct 70
- # splunk\_server 1
- # timeendpos 11
- # timestartpos 10

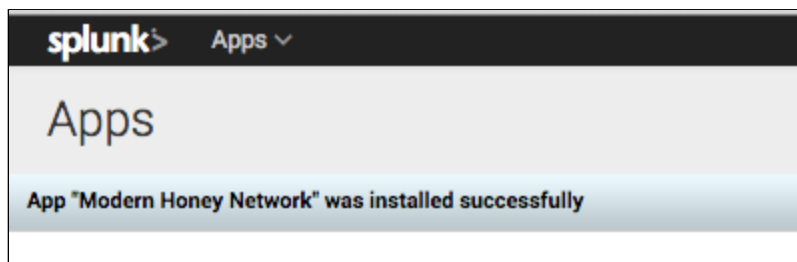
32 more fields  
[Extract New Fields](#)



Next click the button “**Install app from file**”

Click “**Choose File**” and select the modern-honey-network\_01.tgz file you downloaded earlier.

Then click “**Upload**”. You should then see the settings page with the following message:

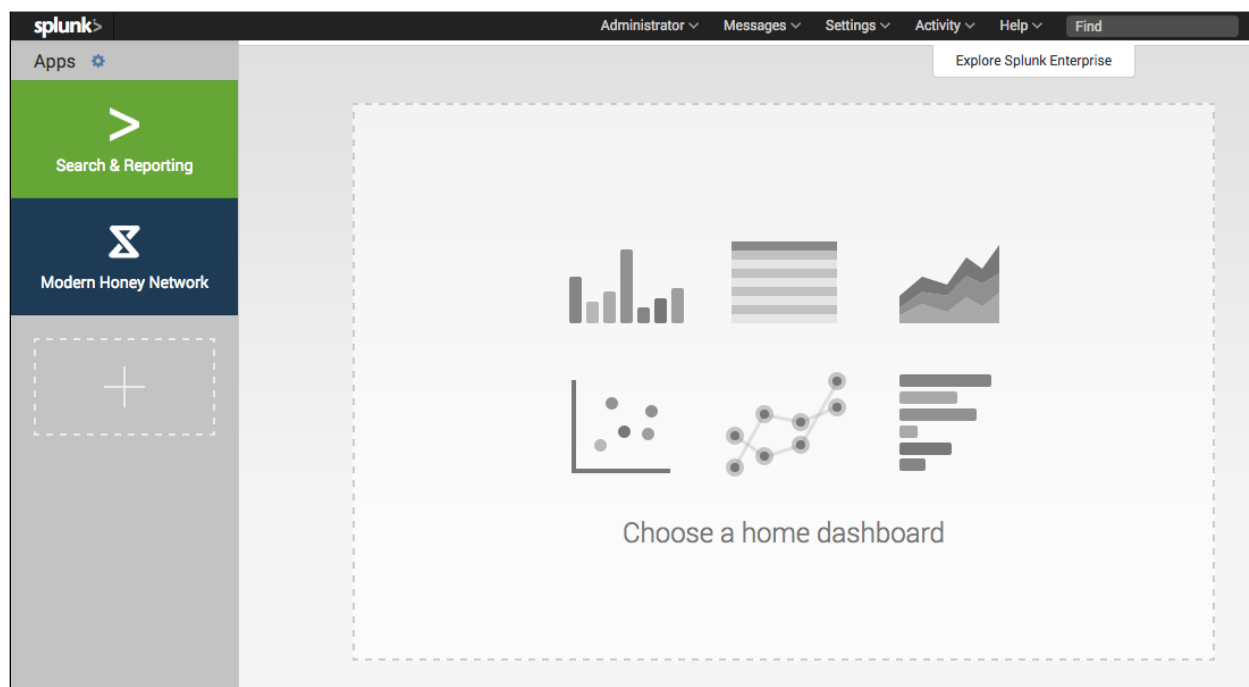


Go back to the command line terminal of your MHN Server and run this command:

```
splunk restart
```

**Note:** This step can be skipped, but some of the icons won’t load properly in the app.

Now, go back to your splunk web app home page; it will force you to log back in. You should now have an MHN App Icon.



Click the “**Modern Honey Network**” icon and explore. If some of the Dashboards are blank that shouldn’t be, change the time drop down at the top from “**Last 24 Hours**” to “**All Time**”.

## Exercise 4: Integrate with ELK

In this part of the exercise we are going to integrate MHN with Elasticsearch/Logstash/Kibana (ELK) so the data collected from our honeypots can be explored using Elasticsearch and Kibana.

### Steps:

1. Run these commands on your MHN Server

```
cd /opt/mhn/scripts/  
./install_elk.sh
```

When prompted, select <Okay> and then <yes> to accept the Oracle Java licenses.

Now lets configure Kibana to use HTTPS through nginx. Run these commands.

```
cd /etc/nginx/sites-available/  
sed 's/8001/5602/; s/8000/5601/' splunk-https > kibana-https  
cd /etc/nginx/sites-enabled  
ln -s /etc/nginx/sites-available/kibana-https  
/etc/init.d/nginx restart
```

Now, as a sanity check, run these commands:

```
supervisorctl status
```

You should see this (all processes RUNNING):

geoloc	RUNNING	pid 31721, uptime 1:07:55
honeymap	RUNNING	pid 31722, uptime 1:07:55
hpfeeds-broker	RUNNING	pid 12916, uptime 1:10:31
hpfeeds-logger-json	RUNNING	pid 8503, uptime 0:01:55
hpfeeds-logger-splunk	RUNNING	pid 2279, uptime 0:28:15
kibana	RUNNING	pid 9950, uptime 0:00:11
logstash	RUNNING	pid 9951, uptime 0:00:11
mhn-celery-beat	RUNNING	pid 691, uptime 0:59:12
mhn-celery-worker	RUNNING	pid 782, uptime 0:58:28
mhn-collector	RUNNING	pid 685, uptime 0:59:12
mhn-uwsgi	RUNNING	pid 683, uptime 0:59:12
mnemosyne	RUNNING	pid 30843, uptime 1:08:30

Then run:

```
/etc/init.d/elasticsearch status
```

You should see:

```
* elasticsearch is running
```

Lastly, lets see if there is any data in ES yet.

```
curl 'localhost:9200/mhn-*/_search?pretty&size=1'
```

Hopefully, you should see some like this:

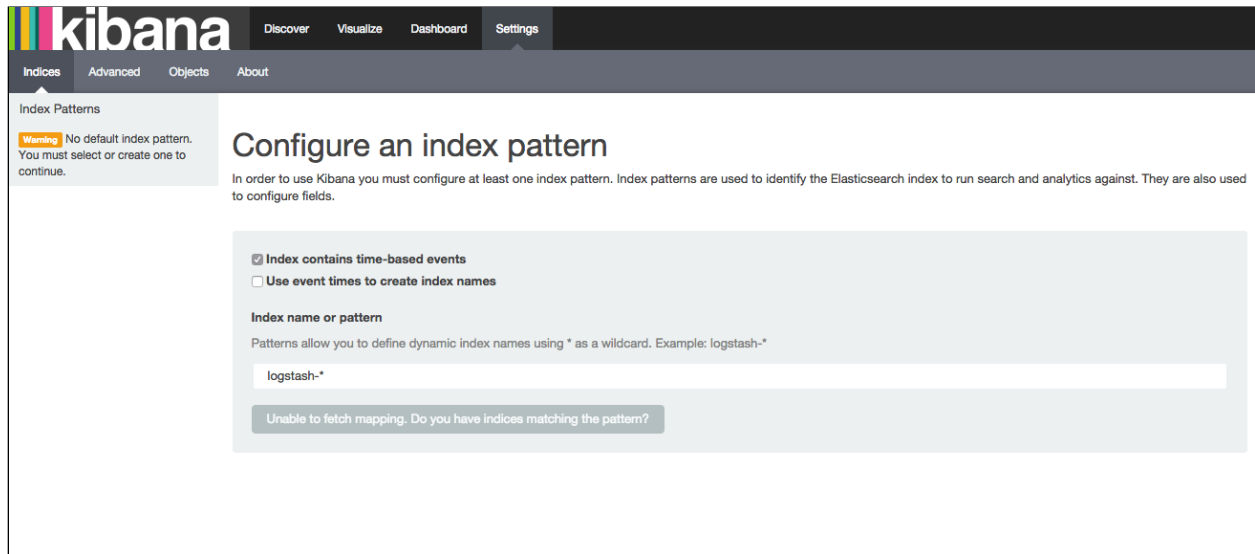
```
{
  "took" : 14,
  "timed_out" : false,
  "_shards" : {
    "total" : 5,
    "successful" : 5,
    "failed" : 0
  },
  "hits" : {
    "total" : 61,
    "max_score" : 1.0,
    "hits" : [ {
      "_index" : "mhn-201507311200",
      "_type" : "event",
      "_id" : "AU7kHgl3n-cYOCs_cySO",
      "_score" : 1.0,
      "_source": {"message": {"direction": "inbound", "protocol": "ip", "ids_type":
\\network\\", \"timestamp\": \"2015-07-31T12:37:29.959419\\", \"vendor_product\": \"Kippo\\",
\\type\": \"kipko.sessions\\", \"app\": \"kipko\\", \"src_ip\": \"218.65.30.61\\", \"dest_port\":
22, \"signature\": \"SSH session on kipko honeypot\\", \"ssh_version\": \"SSH-2.0-PUTTY\\",
\\src_port\": 48156, \"dest_ip\": \"127.0.0.1\\", \"sensor\":
\\\"be1993f0-377b-11e5-be94-040161165601\\", \"transport\": \"tcp\\", \"severity\":
\\\"high\\\"}, \"@version\": \"1\", \"@timestamp\": \"2015-07-31T12:37:30.570Z\", \"host\": \"loki.mhn-server.com\", \"p
ath\": \"/var/log/mhn/mhn-json.log\", \"direction\": \"inbound\", \"protocol\": \"ip\", \"ids_type\": \"network\", \"time
stamp\": \"2015-07-31T12:37:29.959419\", \"vendor_product\": \"Kippo\", \"type\": \"kipko.sessions\", \"app\": \"kipko
\", \"src_ip\": \"218.65.30.61\", \"dest_port\": 22, \"signature\": \"SSH session on kipko
honeypot\", \"ssh_version\": \"SSH-2.0-PUTTY\", \"src_port\": 48156, \"dest_ip\": \"127.0.0.1\", \"sensor\": \"be1993f0
-377b-11e5-be94-040161165601\", \"transport\": \"tcp\", \"severity\": \"high\", \"src_ip_geo\": {\"ip\": \"218.65.30.6
1\", \"country_code2\": \"CN\", \"country_code3\": \"CHN\", \"country_name\": \"China\", \"continent_code\": \"AS\", \"regio
n_name\": \"03\", \"city_name\": \"Nanchang\", \"latitude\": 28.550000000000001, \"longitude\": 115.93329999999997, \"
timezone\": \"Asia/Shanghai\", \"real_region_name\": \"Jiangxi\", \"location\": [115.93329999999997, 28.55000000
000001], \"coordinates\": [115.93329999999997, 28.550000000000001]}}
    } ]
  }
}
```



4. Ok now, lets build some Kibana Dashboards. Open Kibana by visiting:

<https://<mhn-server-domain-name>:5602/>

You should see a page like this:

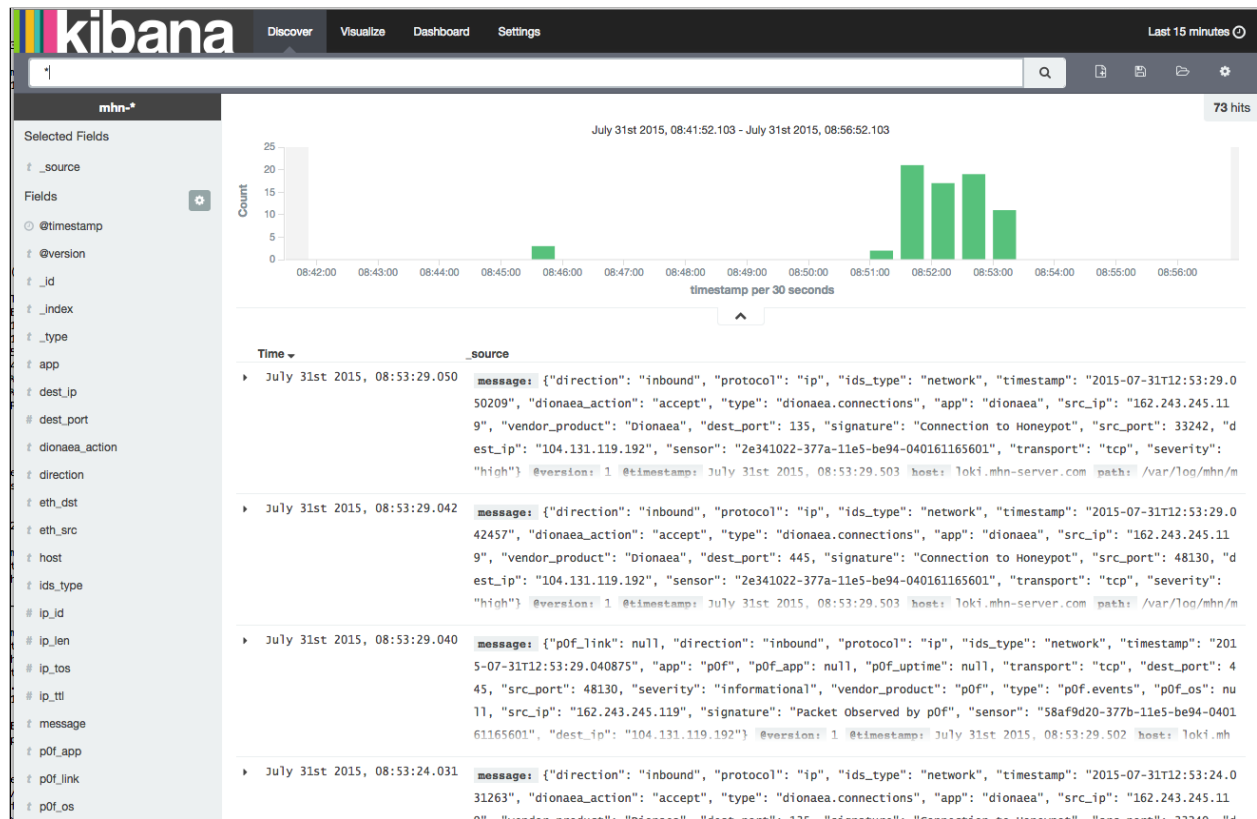


Change `logstash-*` to `mhn-*`

And select **“timestamp”** as the **“Time-field name”**

Click **“Create”**

As a sanity check, click the “Discover” tab at the top. You should see a page like this with data in it.



Ok, in order to build a Kibana Dashboard we need to create several visualization widgets including charts, graphs, metrics, and tables. We will later add these to a Dashboard.

## Create some Metric widgets

Click the “**Visualization**” tab.

Select Metric

Then select **From new Search** as the “**Select a search source**”

Then simply save this by clicking on the small Floppy Disk icon on the top right.

Name it “**Events Count**”

Click “**Save**”

Now, from the same page, change the search from “\*” to “severity:high” in the search bar. Click <ENTER>.

Then simply save this by clicking on the small Floppy Disk icon on the top right.

Name it “**High Severity Count**”

Click “**Save**”

Now, from the same page, change the search back to “\*” in the search bar. Click <ENTER>.

Expand the Metric drop menu

Change the Aggregation from “**Count**” to “**Unique Count**”

Select “src\_ip” as the field.

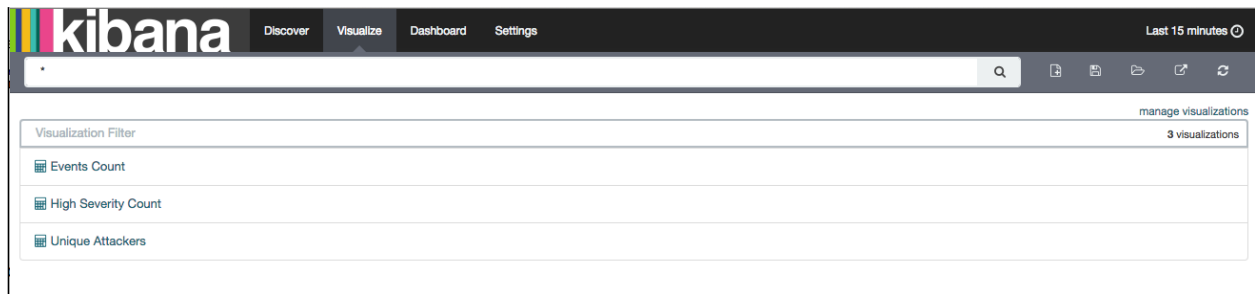
Click the Apply button at the bottom.

Then simply save this by clicking on the small Floppy Disk icon on the top right.

Name it “**Unique Attackers**”

Click “**Save**”

If you did everything correctly, you should be able to your saved visualizations listed after clicking the small folder icon on the right side of the Search bar. It should look like this:



## Create some Facet widgets

Click the “**Visualization**” tab.

Select **Table Data**

Then select **From new Search** as the “**Select a search source**”

Under the buckets section (on the left), click **split rows**

Then change Aggregation to Terms

Select “src\_ip” as the Field

Change size from 5 to 10

Click **Apply**

Then simply save this by clicking on the small Floppy Disk icon on the top right.

Name it “**Top Attackers**”

Click “**Save**”

Now, from the same page, expand the “**split rows**” widget and change the field from **src\_ip** to **src\_ip\_geo.country\_code2**

Click **Apply**

Then simply save this by clicking on the small Floppy Disk icon on the top right.

Name it “**Top Attacker Countries**”

Click “**Save**”

Now, from the same page, expand the “**split rows**” widget again and change the field from **src\_ip\_geo.country\_code2** to **dest\_port**

Click **Apply**

Then simply save this by clicking on the small Floppy Disk icon on the top right.

Name it “**Top Ports**”

Click “**Save**”

## Create an Activity Bar Chart

Click the “**Visualization**” tab.

Select **Vertical Bar Chart**

Then select **From new Search** as the “**Select a search source**”

Select **X-Axis** as the **bucket type**

Select **Date Histogram** for the **Aggregation**

Click **Apply**

Then simply save this by clicking on the small Floppy Disk icon on the top right.

Name it “**Activity Chart**”

Click “**Save**”

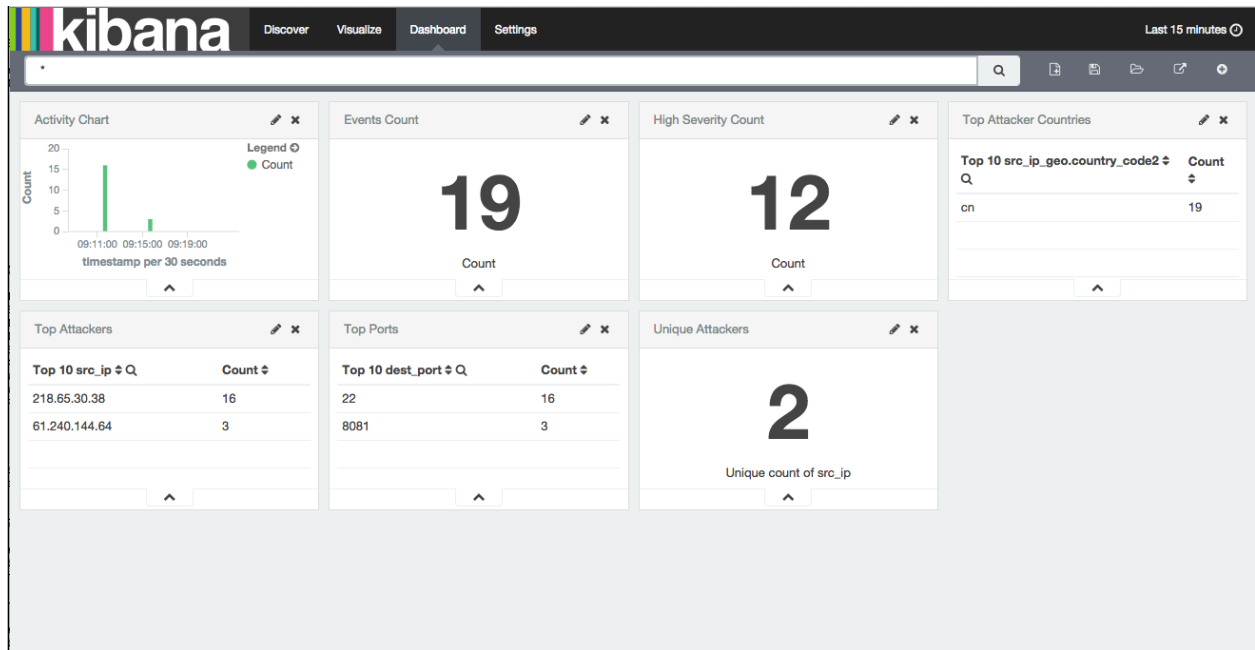
## Create the Dashboard

Click “**Dashboard**” at the top

Click the + icon on the right of the search bar

Then simply start clicking the names of each of the Widgets we created earlier. They will start appearing on the screen

When you're done, your screen should look like this:



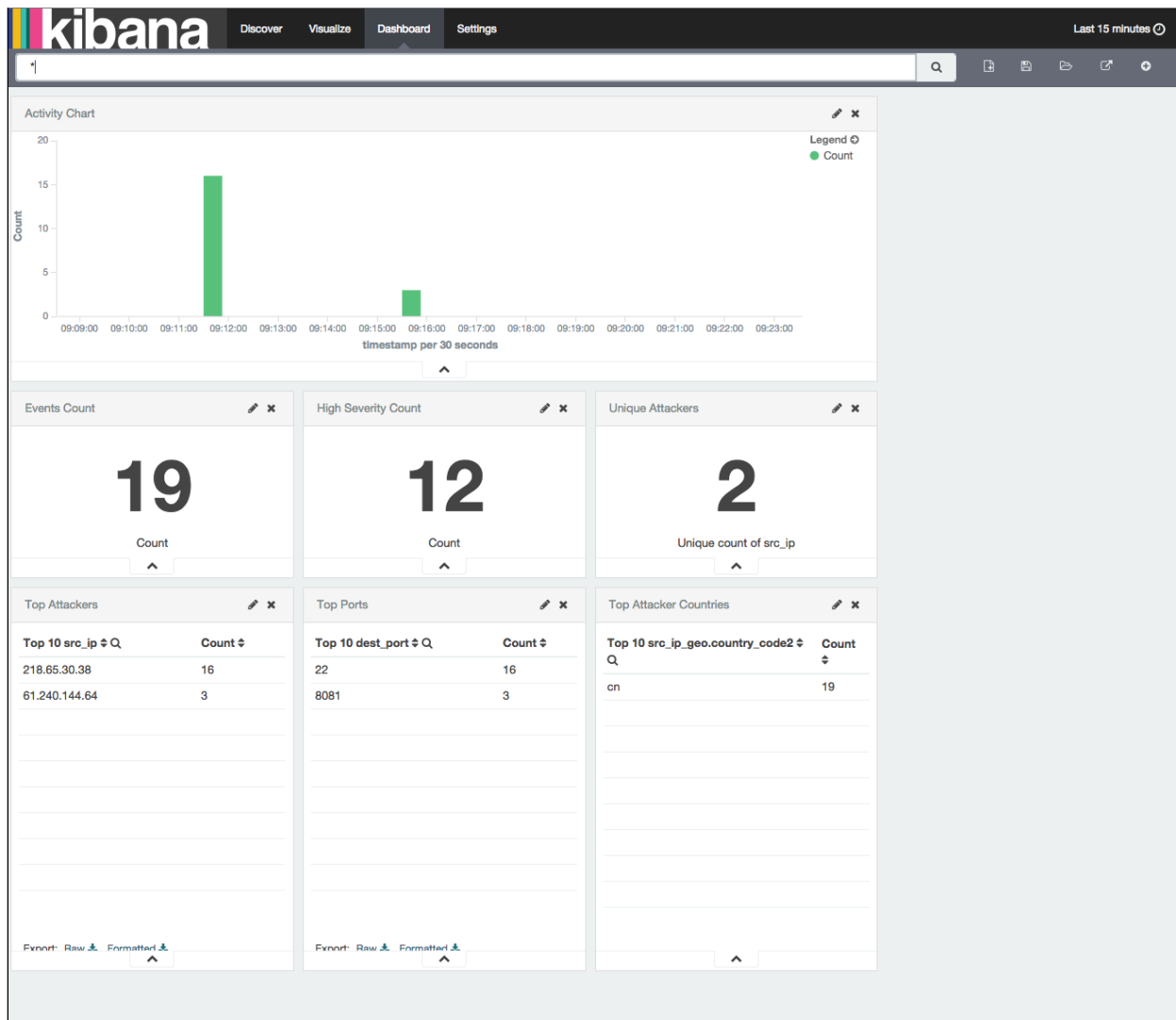
Now resize and rearrange to your liking. You can change the size of the widget by selecting its bottom right corner and dragging it. You can move the widget by clicking and dragging.

When you like how it looks, simply save this by clicking on the small Floppy Disk icon on the top right.

Name it **"MHN Dashboard"**

Click **"Save"**

Here's how mine looks:



## Appendix A: Enabling Swap

If you encounter issues where Elasticsearch keeps crashing immediately upon start up, but you are not seeing any `OutOfMemoryExceptions` in `/var/log/elasticsearch/elasticsearch.log`, AND you are seeing messages like this in `/var/log/syslog`:

```
Aug  3 11:23:51 loki kernel: [276082.892438] Out of memory: Kill process 9642 (java) score 171 or sacrifice child
Aug  3 11:23:51 loki kernel: [276082.892672] Killed process 9642 (java) total-vm:3502340kB, anon-rss:350364kB, file-rss:0kB
```

You may need to enable swap space (or stop other processes such as splunk).

### Steps to enable swap:

```
dd if=/dev/zero of=/swapfile bs=1M count=2048
chmod 600 /swapfile
mkswap /swapfile
swapon /swapfile
swapon -s
```