



ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΛΟΠΟΝΝΗΣΟΥ
ΣΧΟΛΗ ΟΙΚΟΝΟΜΙΑΣ ΚΑΙ ΤΕΧΝΟΛΟΓΙΑΣ
Τμήμα Πληροφορικής και Τηλεπικοινωνιών

1^η Εργασία

Ασφάλεια και οπτικοποίηση μεγάλων δεδομένων

Συγγραφείς:

Φώτης Παπαδήμας , 2022202204022 , dit2222 @go.uop.gr

Δημήτρης Σοφός , 2022202204013 , dit2213@go.uop.gr

Γιώργος Φασάκης , 2022202204011 , dit2211@go.uop.gr

Διδάσκοντες:

Νικόλαος Κολοκοτρώνης, Αναπληρωτής Καθηγητής

Νικόλαος Πλατής, Επίκουρος Καθηγητής

Ιούνιος 2023

Πίνακας περιεχομένων

1	Εισαγωγή.....	4
2	Zeek.....	5
2.1	Zeek Cluster	5
2.2	Network traffic slicing.....	6
3	ELK STACK.....	7
3.1	FileBeat	7
3.2	Elastic Search	8
3.3	Kibana	9
3.4	Configuration	11
3.4.1	Εγκατάσταση και ρύθμιση docker container	12
4	Εργαλεία ανίχνευσης εισβολών	14
4.1	ElastAlert (rule-based)	14
4.2	Elastic ML (anomaly detection)	16
5	Πραγματοποίηση επιθέσεων	19
5.1	Επανεκπομπή pcap αρχείων	19
5.2	Ανίχνευση και ανάλυση Εισβολών	21
6	Επίλογος.....	23
7	Βιβλιογραφία	24

Κατάλογος συντομογραφιών

IDS	Intrusion detection system
IoT	Internet of things
ELK	Elasticsearch, Logstash and Kibana
pcap	Application programming interface for capturing network traffic
ddos	Denial-of-service attack
ssh	Secure Shell

1 Εισαγωγή

Στα πλαίσια της παρούσας εργασίας κληθήκαμε να δημιουργήσουμε ένα κατακευματισμένο δίκτυο ανίχνευσης εισβολών (**IDS**). Η δομή του συστήματος αποτελείται από μια συλλογή εργαλείων, τα οποία δρουν συνεργατικά ώστε να επιτευχθεί επιτήρηση της δικτυακής κίνησης και μετέπειτα να εφαρμοστούν σε αυτή τεχνικές ανίχνευσης ανωμαλιών. Για το σκοπό αυτό δημιουργήθηκε ένα κατακευματισμένο **zeek cluster** αποτελούμενο από 3 **workers** που λειτουργούν στο ίδιο μηχάνημα. Οι **workers** κάνουν **monitor** τη δικτυακή κίνηση σε ένα κοινό **network-interface**. Στην συνέχεια τα **captured logs** στέλνονται στο **ELK stack** (**Elasticsearch**, **Filebeat** και **Kibana**). Το **ELK stack** αναλαμβάνει την συλλογή, οπτικοποίηση και ανάλυση των δεδομένων αυτών. Στη συνέχεια αξιοποιούνται δύο **plugins** του **ELK stack** (**ElastAlert**, **Elastic ML**) για την δημιουργία **alerts** και την ανίχνευση ανωμαλιών στα **logs**. Για να επιβεβαιώσουμε την ορθότητα του συστήματος, διενεργήθηκε επανεκπομπή **pcap** αρχείων από το **IoT Network intrusion dataset** [3].

Όλα τα **configuration files** και οδηγίες για την υλοποίηση βρίσκονται στο **github repo**:

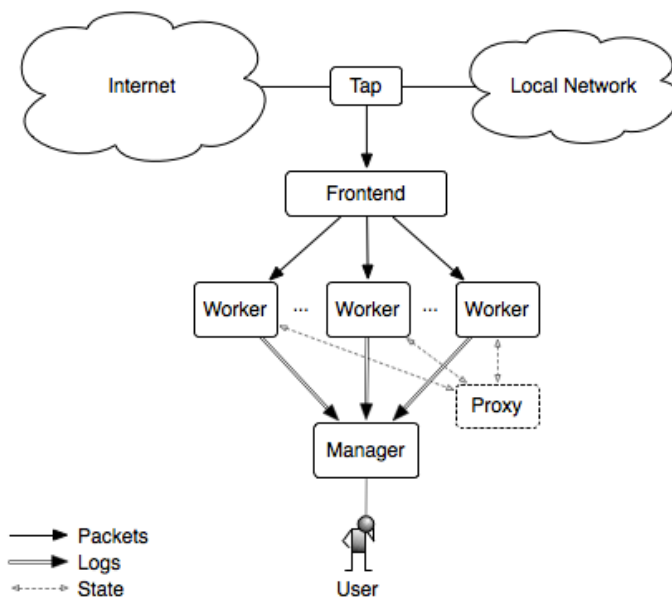
https://github.com/gfasa/Zeek_ELK/

2 Zeek

Το **Zeek** εγκαταστάθηκε στο default directory (/opt/zeek). Για να προσαρμόσουμε τη λειτουργία του, τροποποιήσαμε το configuration αρχείο που βρίσκεται στη διαδρομή /opt/zeek/etc/node.cfg. Στο αρχείο αυτό περιλαμβάνεται η επιλογή μεταξύ δύο τρόπων λειτουργίας του Zeek.

Ο πρώτος είναι σε λειτουργία standalone, δηλαδή το Zeek να τρέχει σε ένα πυρήνα του συστήματος και ο άλλος είναι σε **cluster** λειτουργία. Η δεύτερη λειτουργία είναι και η απαιτούμενη στα πλαίσια της τρέχουσας εργασίας.

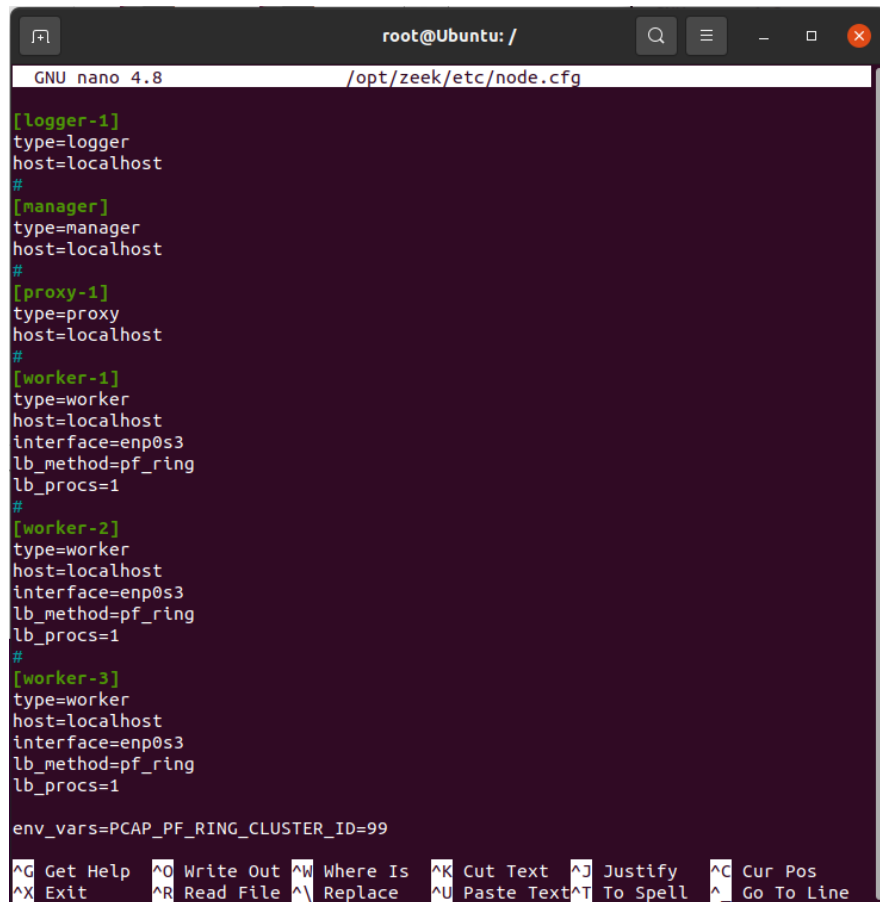
2.1 Zeek Cluster



Οι βασικοί κόμβοι που απαρτίζουν ένα **Zeek cluster** είναι οι **manager**, **logger**, **proxy** και ένας αριθμός από **workers**.

Ο κόμβος manager είναι υπεύθυνος για τη συλλογή logs από τους υπόλοιπους κόμβους και για τη ενοποίηση τους σε ένα κοινό log-file. Ο κόμβος logger μειώνει το φόρτος που θα αναλάμβανε ο manager. Στη συνέχεια, ο proxy κόμβος χρησιμοποιείται όταν υπάρχει ανάγκη το cluster να ισομοιράσει οποιαδήποτε μορφή φόρτου εργασίας και να απελευθερώσει πόρους των υπολοίπων κόμβων. Τέλος, οι worker κόμβοι είναι αυτοί που επιτελούν τη παρακολούθηση της δικτυακής κίνησης.

Παρακάτω δίνεται ένα στιγμιότυπο από τις ρυθμίσεις που επιλέχθηκαν.



```
GNU nano 4.8 /opt/zeek/etc/node.cfg

[logger-1]
type=logger
host=localhost
#
[manager]
type=manager
host=localhost
#
[proxy-1]
type=proxy
host=localhost
#
[worker-1]
type=worker
host=localhost
interface=enp0s3
lb_method=pf_ring
lb_procs=1
#
[worker-2]
type=worker
host=localhost
interface=enp0s3
lb_method=pf_ring
lb_procs=1
#
[worker-3]
type=worker
host=localhost
interface=enp0s3
lb_method=pf_ring
lb_procs=1

env_vars=PCAP_PF_RING_CLUSTER_ID=99

^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Paste Text ^T To Spell  ^_ Go To Line
```

Εικόνα 1. Zeek Configuration

2.2 Network traffic slicing

Για την επίτευξη του network slicing χρειάστηκε να εγκατασταθεί το **PF_RING**, ένα kernel socket το οποίο χρησιμοποιεί Direct NIC Access για να επιταχύνει τη συλλογή και μετάδοση πακέτων. Όπως φαίνεται και στην παραπάνω εικόνα επιλέξαμε όλους τους workers να παρακολουθούν το ίδιο network interface, να χρησιμοποιούν το **PF_RING** και να γίνεται δέσμευση μιας διεργασίας από κάθε worker. Συνεπώς, έχουμε workers που λειτουργούν ως τρεις ξεχωριστές διεργασίες και ο δικτυακός φόρτος μοιράζεται μεταξύ τους.

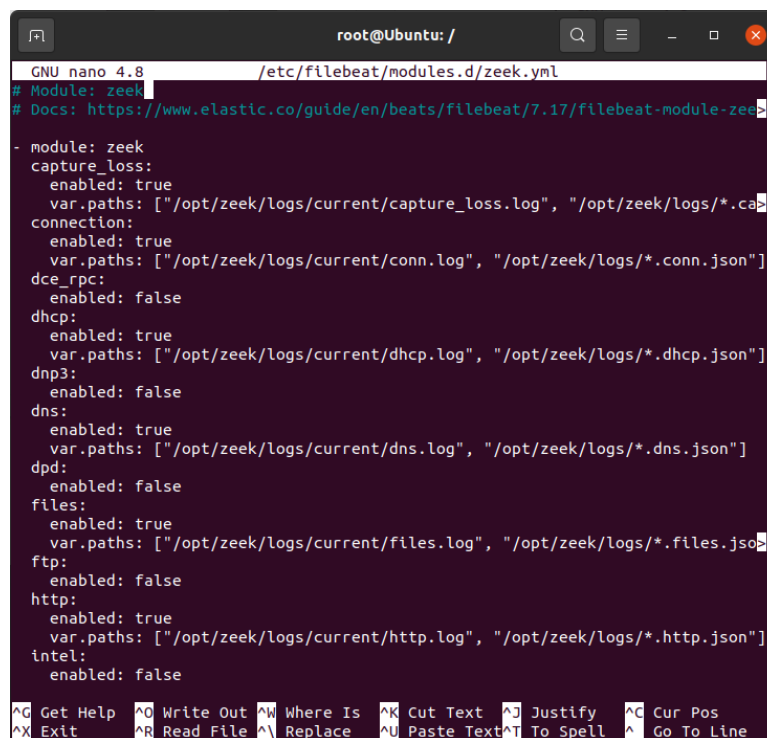
3 ELK STACK

Το **ELK Stack** είναι μια σουίτα λογισμικού η οποία συνδυάζει δυνατότητες για επιτήρηση, αποσφαλμάτωση και προστασία IT περιβαλλόντων μεταξύ άλλων.

Αποτελείται μεταξύ άλλων από 3 κύρια εργαλεία. Το **Elastic Search**, το **Filebeat** και το **Kibana**. Παρακάτω γίνεται αναφορά σε αυτά τα εργαλεία και παρατίθενται εικόνες από τα configuration files στα οποία έγιναν αλλαγές για να ρυθμιστεί η λειτουργία μεταξύ των εργαλείων.

3.1 FileBeat

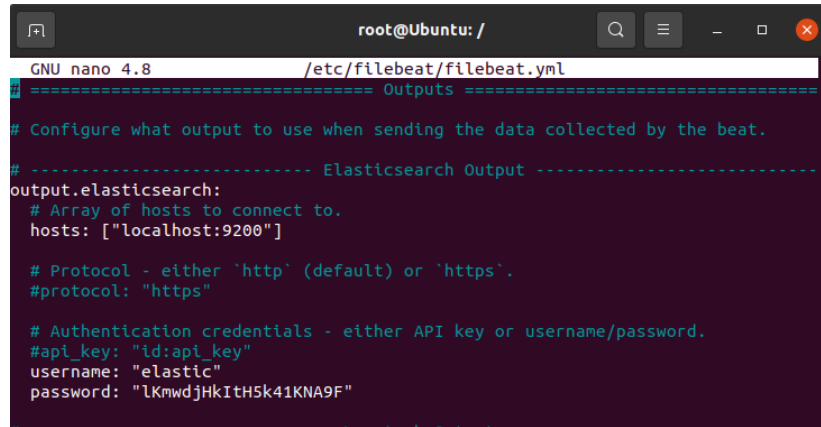
Το **Filebeat** είναι υπεύθυνο για τη συλλογή logs από το **Zeek**, τα οποία μετατρέπονται σε json μορφή από tsv ώστε να είναι δυνατή η ανάγνωσή και επεξεργασία τους από το **Elastic Search**. Το **Filebeat** επίσης ρυθμίστηκε να συγκεντρώνει αρχεία από το σημείο που τα αποθηκεύει το Zeek (/opt/zeek/logs/*). Για να επιτευχθεί αυτό τροποποιήσαμε το αντίστοιχο πεδίο του configuration file που βρίσκεται στο path /etc/filebeat/modules.d/zeek.yml.



```
root@Ubuntu: /  
GNU nano 4.8 /etc/filebeat/modules.d/zeek.yml  
# Module: zeek  
# Docs: https://www.elastic.co/guide/en/beats/filebeat/7.17/filebeat-module-zeek.html  
  
- module: zeek  
  capture_loss:  
    enabled: true  
    var.paths: ["/opt/zeek/logs/current/capture_loss.log", "/opt/zeek/logs/*.capture_loss.log"]  
  connection:  
    enabled: true  
    var.paths: ["/opt/zeek/logs/current/conn.log", "/opt/zeek/logs/*.conn.json"]  
  dce_rpc:  
    enabled: false  
  dhcp:  
    enabled: true  
    var.paths: ["/opt/zeek/logs/current/dhcp.log", "/opt/zeek/logs/*.dhcp.json"]  
  dnp3:  
    enabled: false  
  dns:  
    enabled: true  
    var.paths: ["/opt/zeek/logs/current/dns.log", "/opt/zeek/logs/*.dns.json"]  
  dpd:  
    enabled: false  
  files:  
    enabled: true  
    var.paths: ["/opt/zeek/logs/current/files.log", "/opt/zeek/logs/*.files.json"]  
  ftp:  
    enabled: false  
  http:  
    enabled: true  
    var.paths: ["/opt/zeek/logs/current/http.log", "/opt/zeek/logs/*.http.json"]  
  intel:  
    enabled: false  
  
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos  
^X Exit ^R Read File ^_ Replace ^U Paste Text ^T To Spell ^_ Go To Line
```

Εικόνα 2. FileBeat input logs from Zeek Configuration

Στο επόμενο screenshot φαίνεται το configuration file του **Filebeat** (/etc/filebeat/filebeat.yml) στο οποίο επιπλέον προσδιορίζουμε αν τα logs θα πάνε πρώτα στο **Logstash** ή αν θα σταλούν κατευθείαν στο **Elastic Search**. Στη συγκεκριμένη περίπτωση επιλέξαμε να ισχύει το δεύτερο.

A terminal window showing the configuration of the Filebeat output in the file /etc/filebeat/filebeat.yml. The configuration is set to output to Elasticsearch. The terminal title is 'root@Ubuntu: /'. The nano editor shows the following content:

```
GNU nano 4.8 /etc/filebeat/filebeat.yml
===== Outputs =====

# Configure what output to use when sending the data collected by the beat.

# ----- Elasticsearch Output -----
output.elasticsearch:
  # Array of hosts to connect to.
  hosts: ["localhost:9200"]

  # Protocol - either 'http' (default) or 'https'.
  #protocol: "https"

  # Authentication credentials - either API key or username/password.
  #api_key: "id:api_key"
  username: "elastic"
  password: "lKmwjdjHkItH5k41KNA9F"
```

Εικόνα 3. FileBeat output Configuration

3.2 Elastic Search

Το **Elastic Search** είναι ένα **NoSQL** σύστημα βάσεων δεδομένων το οποίο χρησιμοποιείται για την αποθήκευση, την αναζήτηση και ανάλυση των **logs**. Για τα analytics χρησιμοποιείται παράλληλα με τα υπόλοιπα μέρη του **ELK Stack**.

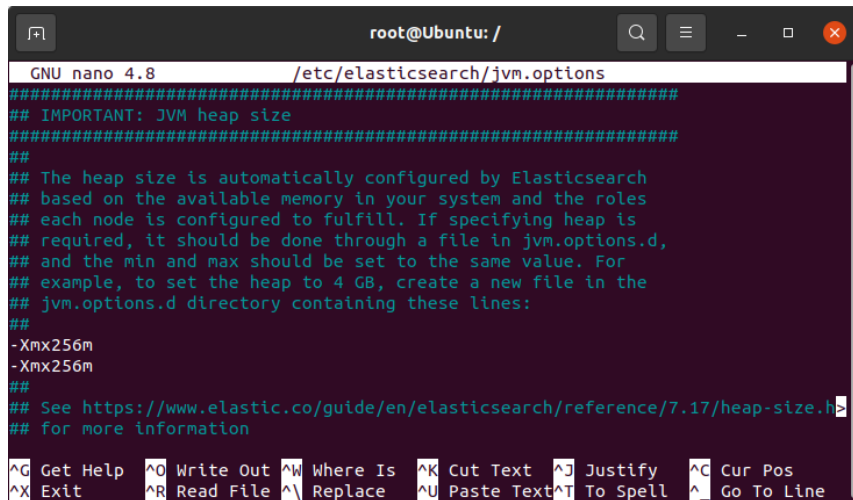
Για να προσαρμόσουμε τη λειτουργία του πρέπει να προσδιορίσουμε σε ποια διεύθυνση θα κάνει bind. Στη συγκεκριμένη περίπτωση προσδιορίζουμε να κάνει bind σε όλα τα διαθέσιμα interfaces για λόγους απλότητας. Επίσης, ρυθμίζουμε το port στο οποίο θα ακούει να είναι το **9200**, το οποίο είναι και το default port για το **Elastic Search**.

A terminal window showing the configuration of the Elastic Search network settings in the file /etc/elasticsearch/elasticsearch.yml. The terminal title is 'root@Ubuntu: /'. The nano editor shows the following content:

```
GNU nano 4.8 /etc/elasticsearch/elasticsearch.yml
# ----- Network -----
# By default Elasticsearch is only accessible on localhost. Set a different
# address here to expose this node on the network:
#
network.host: 0.0.0.0
#
# By default Elasticsearch listens for HTTP traffic on the first free port it
# finds starting at 9200. Set a specific HTTP port here:
#
http.port: 9200
#
# For more information, consult the network module documentation.
#
# ----- Discovery -----
#
# Pass an initial list of hosts to perform discovery when this node is started:
# The default list of hosts is ["127.0.0.1", "[:1]"]
#
discovery.type: single-node
#discovery.seed_hosts: ["host1", "host2"]
#
```

Εικόνα 4. Elastic Search Configuration

Στη συνέχεια θέτουμε το μέγεθος στο heap κομμάτι της μνήμης που θα δεσμεύσει το Java Virtual Machine να είναι **256 MB** ώστε να τρέξει ομαλά στο σύστημα μας



```
root@Ubuntu: /
GNU nano 4.8 /etc/elasticsearch/jvm.options
#####
## IMPORTANT: JVM heap size
#####
##
## The heap size is automatically configured by Elasticsearch
## based on the available memory in your system and the roles
## each node is configured to fulfill. If specifying heap is
## required, it should be done through a file in jvm.options.d,
## and the min and max should be set to the same value. For
## example, to set the heap to 4 GB, create a new file in the
## jvm.options.d directory containing these lines:
##
## -Xmx256m
## -Xms256m
##
## See https://www.elastic.co/guide/en/elasticsearch/reference/7.17/heap-size.html
## for more information
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^_ Replace ^U Paste Text ^T To Spell ^_ Go To Line
```

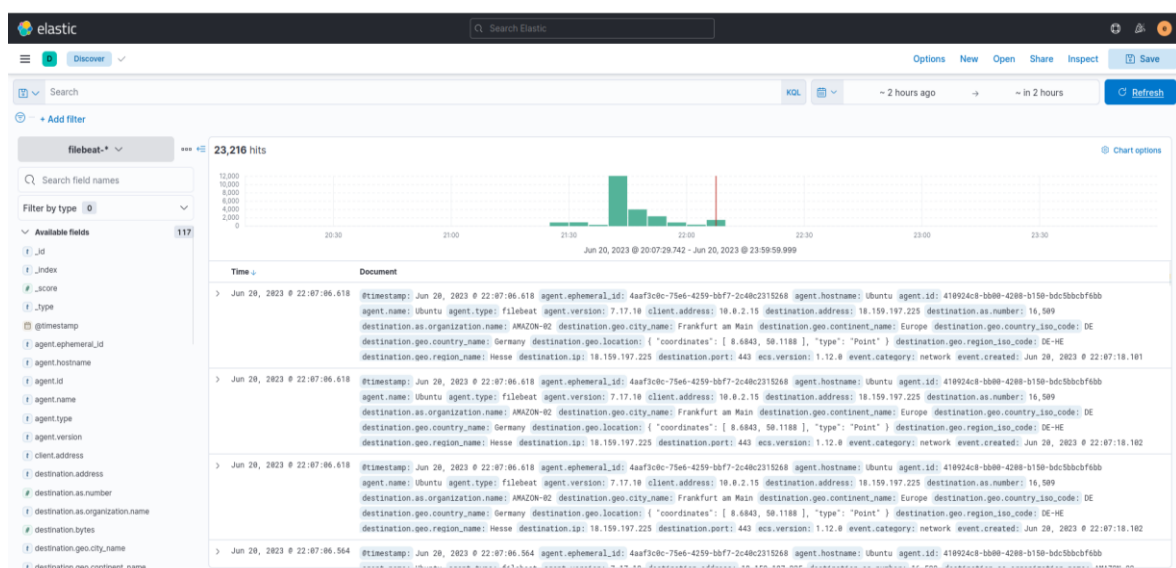
Εικόνα 5. Elastic Search JVM options

3.3 Kibana

Το **Kibana** αποτελεί ένα interface το οποίο αντλεί δεδομένα από το **Elastic Search** μέσω του **Filebeat** και δημιουργεί οπτικοποιήσεις. Παρέχει πληθώρα διαδραστικών εργαλείων που βοηθούν το χρήστη να εξερευνήσει δεδομένα που έχουν αποθηκευτεί από το Elastic Search με σκοπό τη καλύτερη κατανόηση τους. Σύμφωνα με το configuration μας τρέχει στην διεύθυνση **http://localhost:5601/**

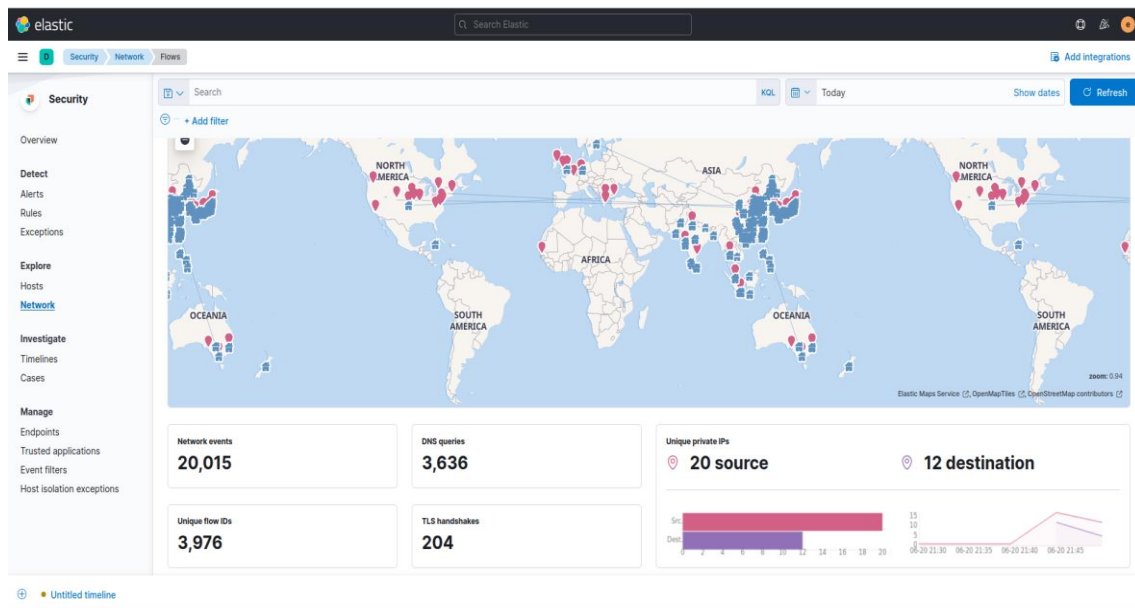
Μέσω του **Discover tab** μπορούμε να έχουμε μια εποπτική ματιά των **events** που ανιχνεύθηκαν από το **Zeek**. Θα ήταν πολύ πιο δύσκολο να ερμηνεύσουμε τα Events κατευθείαν από τα logs του **Zeek**.

Παρακάτω παρατίθεται ένα στιγμιότυπο από το **Discover**.

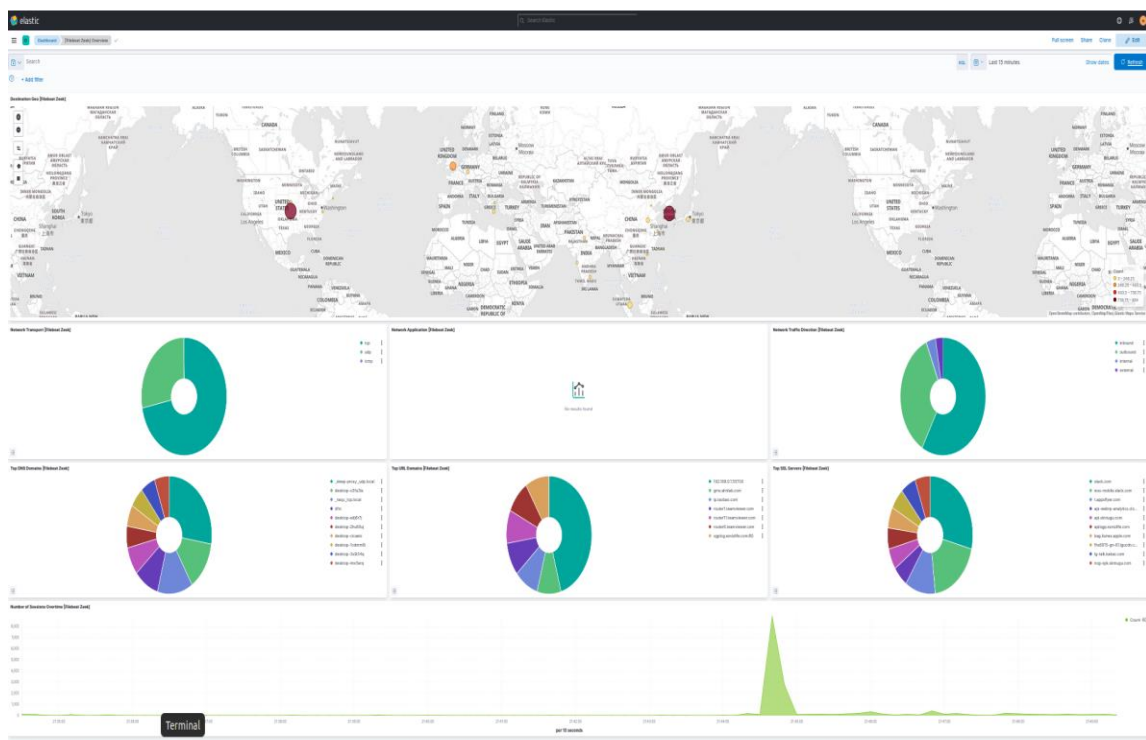


Εικόνα 6. Kibana Discover

Το το **Dashboard tab** του **Kibana** είναι ιδιαίτερα χρήσιμο καθώς διενεργεί αυτόματα **geolocation tracking**. Κατ' αυτό τον τρόπο είναι δυνατό να εντοπισθεί η πηγή κάποιας επίθεσης ή κάποιας ύποπτης σύνδεσης, όπως φαίνεται και στα ακόλουθα στιγμιότυπα.



Εικόνα 7. Kibana Security Network



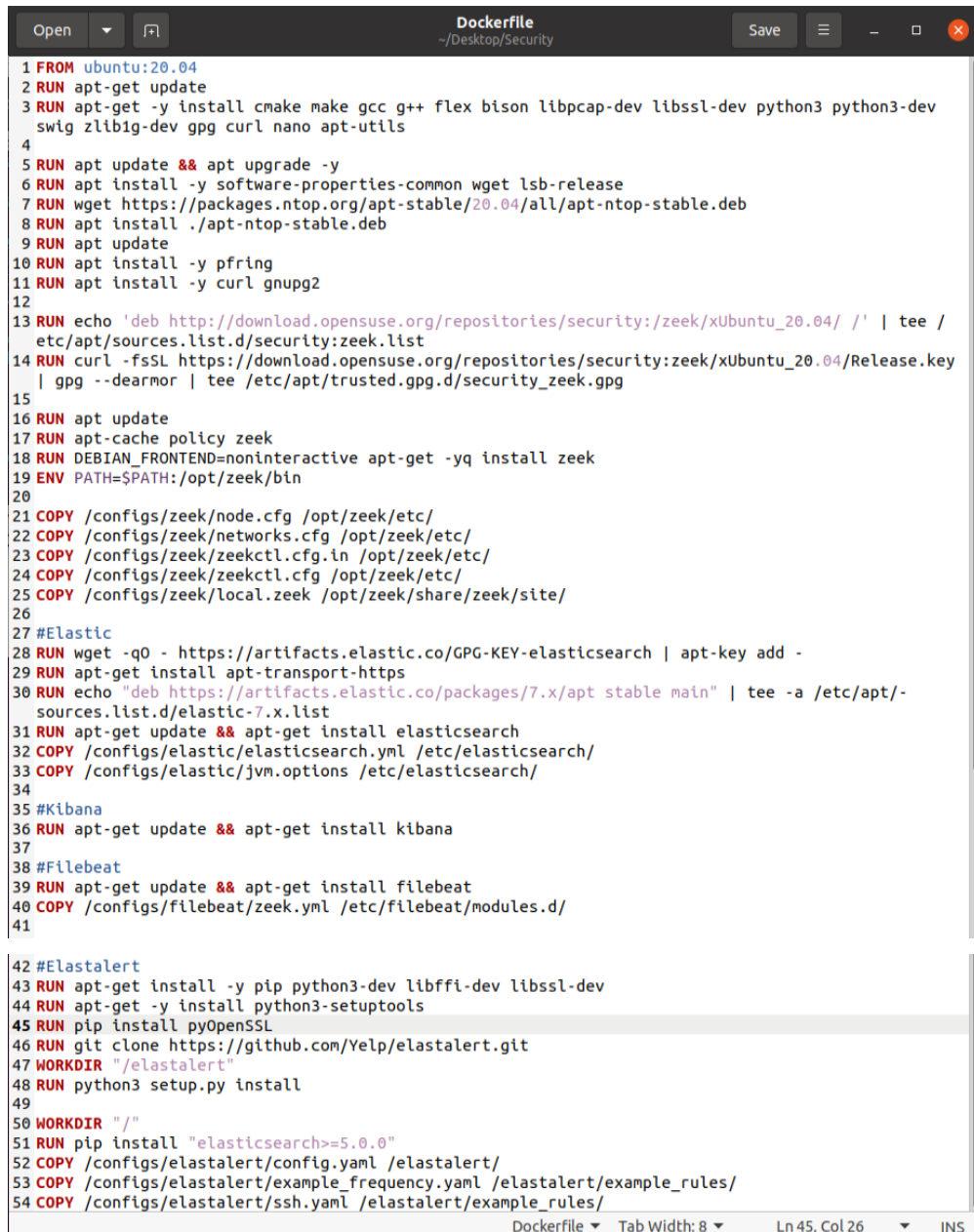
Εικόνα 8. Kibana Dashboard

3.4 Configuration

Η υλοποίηση των προηγούμενων βημάτων έγινε σε ένα **VM** που τρέχει **Ubuntu 20.04.06**.

Το **Zeek** και το **ELK Stack**, εγκαταστάθηκαν σε ένα **docker container** για καλύτερη διαχείριση πόρων στο σύστημα μας. [5][6][7]

Το directory της εργασίας περιέχει έναν φάκελο “**configs**” με όλες τις παραμετροποιήσεις που έχουμε για το **Zeek** και το **ELK stack** [1], το **Dockerfile** από το οποίο θα χτιστεί το image του container μας, τον φάκελο **pcaps-IoT** που περιέχει κάποια από τα **pcap** αρχεία από το **iot-network-intrusion-dataset** [3] .



```
1 FROM ubuntu:20.04
2 RUN apt-get update
3 RUN apt-get -y install cmake make gcc g++ flex bison libpcap-dev libssl-dev python3 python3-dev
  swig zlib1g-dev gpg curl nano apt-utils
4
5 RUN apt update && apt upgrade -y
6 RUN apt install -y software-properties-common wget lsb-release
7 RUN wget https://packages.ntop.org/apt-stable/20.04/all/apt-ntop-stable.deb
8 RUN apt install ./apt-ntop-stable.deb
9 RUN apt update
10 RUN apt install -y pfring
11 RUN apt install -y curl gnupg2
12
13 RUN echo 'deb http://download.opensuse.org/repositories/security:zeek/xUbuntu_20.04/ ' | tee /
  etc/apt/sources.list.d/security:zeek.list
14 RUN curl -fsSL https://download.opensuse.org/repositories/security:zeek/xUbuntu_20.04/Release.key
  | gpg --dearmor | tee /etc/apt/trusted.gpg.d/security_zeek.gpg
15
16 RUN apt update
17 RUN apt-cache policy zeek
18 RUN DEBIAN_FRONTEND=noninteractive apt-get -yq install zeek
19 ENV PATH=$PATH:/opt/zeek/bin
20
21 COPY /configs/zeek/node.cfg /opt/zeek/etc/
22 COPY /configs/zeek/networks.cfg /opt/zeek/etc/
23 COPY /configs/zeek/zeekctl.cfg.in /opt/zeek/etc/
24 COPY /configs/zeek/zeekctl.cfg /opt/zeek/etc/
25 COPY /configs/zeek/local.zeek /opt/zeek/share/zeek/site/
26
27 #Elastic
28 RUN wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | apt-key add -
29 RUN apt-get install apt-transport-https
30 RUN echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | tee -a /etc/apt/-
  sources.list.d/elastic-7.x.list
31 RUN apt-get update && apt-get install elasticsearch
32 COPY /configs/elastic/elasticsearch.yml /etc/elasticsearch/
33 COPY /configs/elastic/jvm.options /etc/elasticsearch/
34
35 #Kibana
36 RUN apt-get update && apt-get install kibana
37
38 #Filebeat
39 RUN apt-get update && apt-get install filebeat
40 COPY /configs/filebeat/zeek.yml /etc/filebeat/modules.d/
41
42 #Elastalert
43 RUN apt-get install -y pip python3-dev libffi-dev libssl-dev
44 RUN apt-get -y install python3-setuptools
45 RUN pip install pyOpenSSL
46 RUN git clone https://github.com/Yelp/elastalert.git
47 WORKDIR "/elastalert"
48 RUN python3 setup.py install
49
50 WORKDIR "/"
51 RUN pip install "elasticsearch>=5.0.0"
52 COPY /configs/elastalert/config.yml /elastalert/
53 COPY /configs/elastalert/example_frequency.yml /elastalert/example_rules/
54 COPY /configs/elastalert/ssh.yml /elastalert/example_rules/
```

Εικόνα 9. Docker File

3.4.1 Εγκατάσταση και ρύθμιση docker container

```
#install docker on host
#https://docs.docker.com/engine/install/ubuntu/

#move to folder
cd ~/Desktop/Security/

#build Zeek Container
sudo docker build -t zeek-base:1.0 .

#start container
sudo docker run -itd --name zeek --net=host -p 9200:9200 -p 5601:5601 zeek-base:1.0

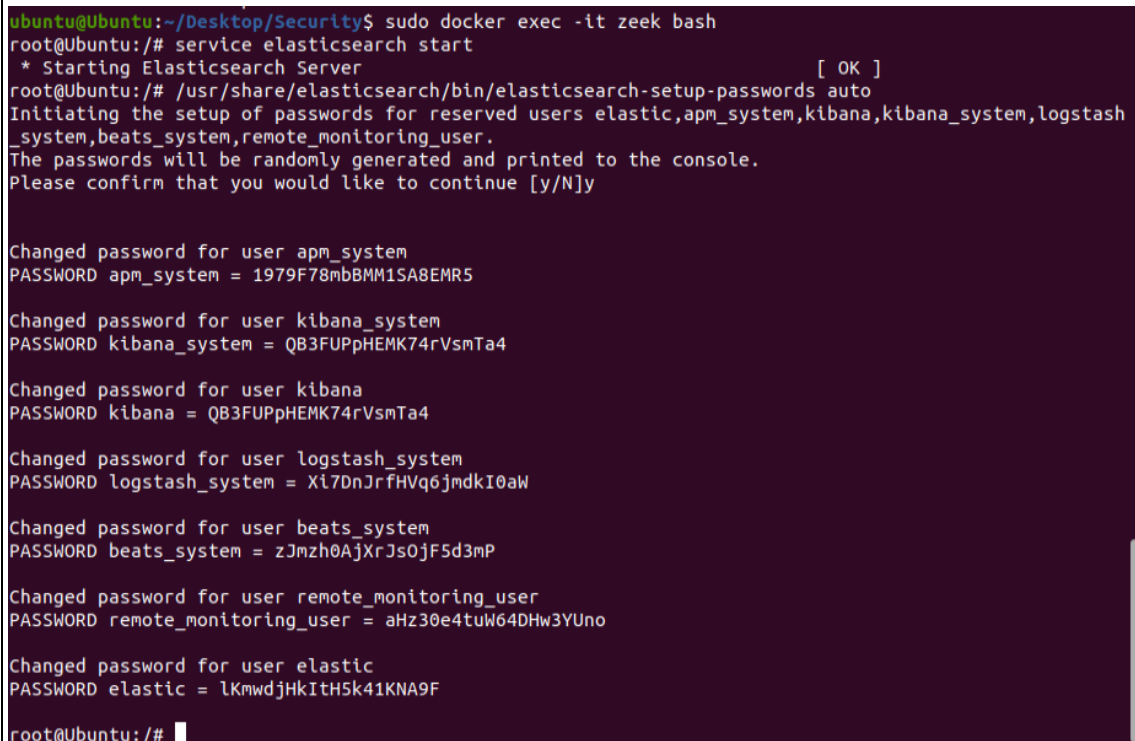
#open container terminal
sudo docker exec -it zeek bash

#-----
#in docker

service elasticsearch start

#due to error, run at host: sudo sysctl -w vm.max_map_count=262144

#produce password for the default elastic user
/usr/share/elasticsearch/bin/elasticsearch-setup-passwords auto
```



```
ubuntu@Ubuntu:~/Desktop/Security$ sudo docker exec -it zeek bash
root@Ubuntu:/# service elasticsearch start
 * Starting Elasticsearch Server [ OK ]
root@Ubuntu:/# /usr/share/elasticsearch/bin/elasticsearch-setup-passwords auto
Initiating the setup of passwords for reserved users elastic,apm_system,kibana,kibana_system,logstash_system,beats_system,remote_monitoring_user.
The passwords will be randomly generated and printed to the console.
Please confirm that you would like to continue [y/N]y

Changed password for user apm_system
PASSWORD apm_system = 1979F78mbBMM1SA8EMR5

Changed password for user kibana_system
PASSWORD kibana_system = QB3FUPpHEMK74rVsmTa4

Changed password for user kibana
PASSWORD kibana = QB3FUPpHEMK74rVsmTa4

Changed password for user logstash_system
PASSWORD logstash_system = Xl7DnJrFHVq6jmdkI0aW

Changed password for user beats_system
PASSWORD beats_system = zJmzh0AjXrJs0jF5d3mP

Changed password for user remote_monitoring_user
PASSWORD remote_monitoring_user = aHz30e4tuW64DHw3YUno

Changed password for user elastic
PASSWORD elastic = lKmwjdjHkItH5k41KNA9F

root@Ubuntu:/#
```

Εικόνα 10. Δημιουργία passwords για το xpack.security

```
#as per previous output edit elasticsearch.username, elasticsearch.password
sudo nano /etc/kibana/kibana.yml

#due to error on http://localhost:5601/, it is madatory to copy the ouput of this to kibana.yml
sudo /usr/share/kibana/bin/kibana-encryption-keys generate

#start kibana / check on localhost:5601
service kibana start

#start zeek
zeekctl deploy

#setup passwords for filebeat as per previous output
nano etc/filebeat/filebeat.yml

#start filebeat
filebeat setup
service filebeat start

#on host change time to UTC due to error

#config es_username, es_password, on elastalert
#also enter the slack webhook for the alerts
#https://hooks.slack.com/services/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
cd elastalert/
nano config.yaml

elastalert-create-index

#needed cause of an error
python3.8 -m pip install APScheduler --upgrade

#command to run only one rule for testing
python3 -m elastalert.elastalert --verbose --rule example_rules/example_frequency.yaml

#-----
```

Τελειώσαμε με το αρχικό configuration του Zeek-ELK container μας.

4 Εργαλεία ανίχνευσης εισβολών

Σε αυτήν την φάση θα δοκιμάσουμε δυο plugin του **ELK Stack**. Το **Elast Alert** που είναι μια rule based λύση για ανίχνευση εισβολών, καθώς και το **Elastic ML** που διαβάζοντας την δικτυακή κίνηση μπορεί να εντοπίσει anomalies και επίσης να δημιουργήσει alerts.

4.1 ElastAlert (rule-based)

Μέσα από το **docker** εκτελούμε τις παρακάτω εντολές για να πάρουμε δυο από τα έτοιμα templates που έχει το **Elast Alert** [2] και πάνω σε αυτά να χτίσουμε τους δικούς μας κανόνες.

```
cd elastalert/  
mkdir my_rules  
cp example_rules/example_frequency.yaml my_rules/example_frequency.yaml  
cp example_rules/ssh.yaml my_rules/ssh.yaml
```

*ο φάκελος **my_rules** έχει οριστεί ως ο default φάκελος για να διαβάζει rules το **ElastAlert** στο αρχικό configuration του container. Επίσης οι κανόνες που φτιάξαμε βρίσκονται στα config files του φακέλου της εργασίας μας.

Κάποια χαρακτηριστικά screenshots και η λογική των rules που φτιάξαμε:

example_frequency.yaml

```
# Rule name, must be unique  
name: Big Data Security and Visualization  
  
# (Required)  
# Type of alert.  
# the frequency rule type alerts when num_events events occur with timeframe time  
type: frequency  
  
# (Required)  
# Index to search, wildcard supported  
index: filebeat-*  
  
# (Required, frequency specific)  
# Alert when this many documents matching the query occur within a timeframe  
num_events: 20  
  
# (Required, frequency specific)  
# num_events must occur within this amount of time to trigger an alert  
timeframe:  
  hours: 1  
  
# (Required)  
# A list of Elasticsearch filters used for find events  
# These filters are joined with AND and nested in a filtered query  
# For more info: http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/query-dsl.html  
filter:  
  - term:  
      source.as.organization.name: "China Mobile Communications Group Co., Ltd."  
  
# (Required)  
# The alert is use when a match is found  
alert:  
  - "slack"  
  
# (required, email specific)  
# a list of email addresses to send alerts to  
slack:  
  slack_webhook_url: "https://hooks.slack.com/services/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"  
  slack_title: "China Mobile Communications Group Co., Ltd. EVENT!!!"
```

Εικόνα 11. Elast Alert Frequency rule

Από την ανάλυση των logs από το επόμενο βήμα όπως θα δούμε , αλλά και από το **Kibana Discovery tool**, φαίνεται ότι πολλά από τα επιτιθέμενα πακέτα έρχονται από το συγκεκριμένο source.organization.name. Οπότε για τις ανάγκες της εργασίας αποφασίσαμε να το θεωρήσουμε ως επικίνδυνο και να φτιάξουμε έναν **frequency** κανόνα που αν εντοπίσει 20 φορές σε διάστημα 1 ώρας κάποιο log με από αυτόν τον οργανισμό μας ειδοποιεί μέσω **Slack**.

ssh.yaml

```
# Alert when this many documents matching the query occur within a timeframe
num_events: 3

# num_events must occur within this amount of time to trigger an alert
timeframe:
  minutes: 1

# A list of elasticsearch filters used for find events
# These filters are joined with AND and nested in a filtered query
# For more info: http://www.elasticsearch.org/guide/en/elasticsearch/reference/current/query-dsl.html
filter:
- query:
    query_string:
      query: "event.type:authentication_failure"

index: filebeat-*

# When the attacker continues, send a new alert after x minutes
realert:
  minutes: 1

query_key:
- source.ip

include:
- host.hostname
- user.name
- source.ip

include_match_in_root: true

alert_subject: "SSH abuse on <{}> "
alert_subject_args:
- host.hostname

alert_text: |-
  An attack on {} is detected.
  The attacker looks like:
  User: {}
  IP: {}
alert_text_args:
- host.hostname
- user.name
- source.ip

# The alert is use when a match is found
alert:
- "slack"

slack:
  slack_webhook_url: "https://hooks.slack.com/services/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"
  slack_title: "SSH Attack Event!!"

alert_text_type: alert_text_only
```

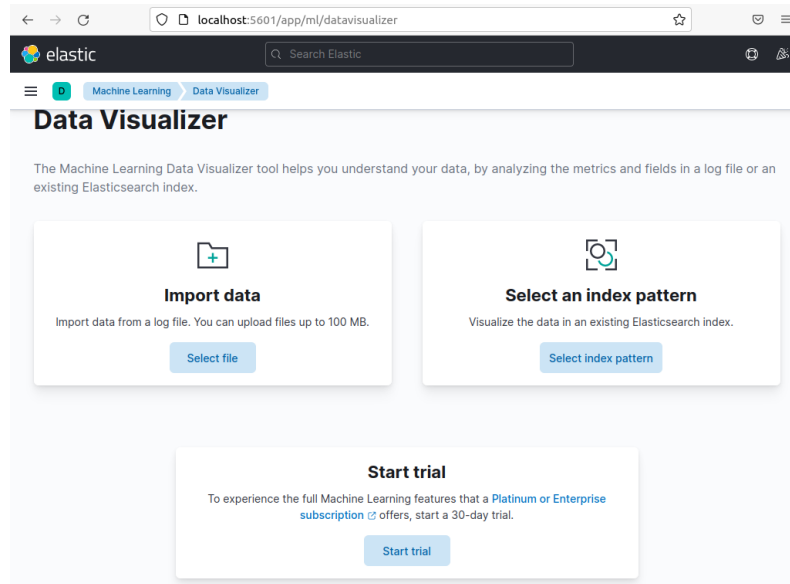
Εικόνα 12. Elast Alert SSH rule

Αυτός είναι ένας γενικός κανόνας που εντοπίζει πιθανές επιθέσεις **brute force** στο πρωτόκολλο **ssh**. Αν εντοπίσει 3 αποτυχημένες προσπάθειες σύνδεσης μέσω **ssh** σε 1 λεπτό μας ειδοποιεί μέσω **Slack**.

****Επιλέξαμε το slack για να μας έρχονται τα alerts αντί της default επιλογής μέσω email, λόγω ευκολίας στην παραμετροποίηση, και για να αποφύγουμε να στήσουμε έναν mail server από την αρχή στο vm μας.**

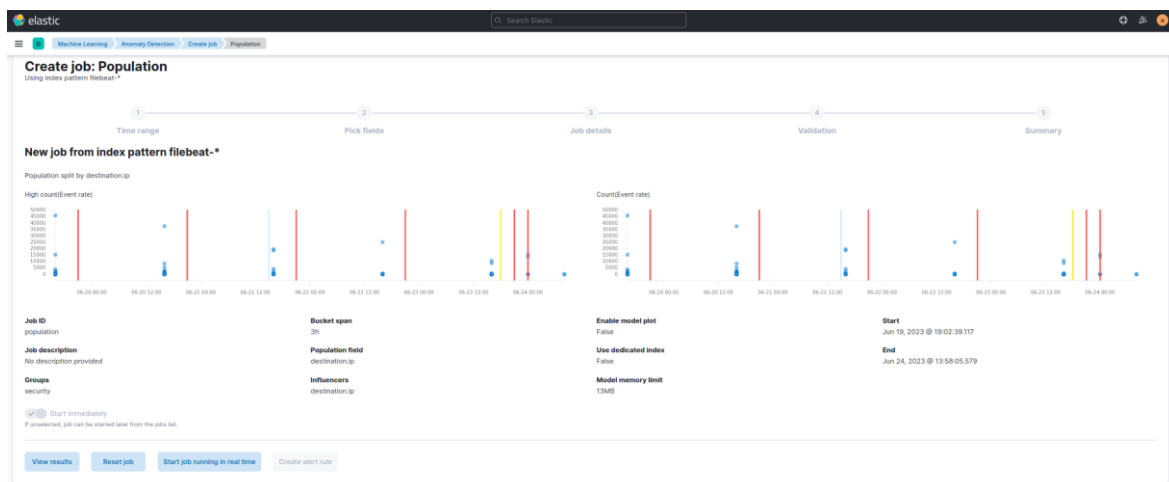
4.2 Elastic ML (anomaly detection)

Για να ενεργοποιήσουμε το **ElasticML** [4] εργαλείο που προσφέρεται από το **ELK stack**. Πηγαίνουμε στον browser “localhost:5601” στο **Analytics/Machine learning tab** και ενεργοποιούμε το trial όπως φαίνεται στην παρακάτω εικόνα:



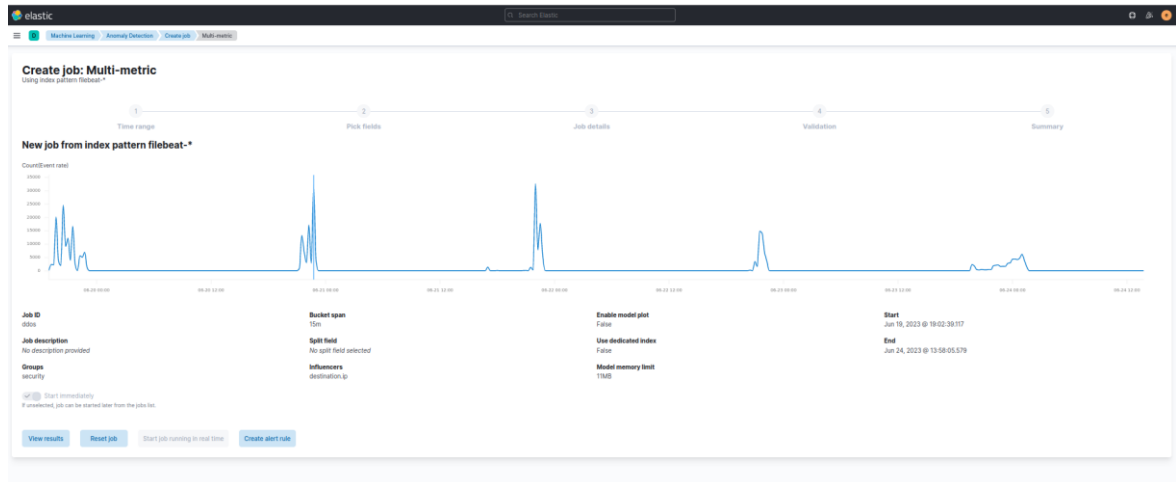
Εικόνα 12. Ενεργοποίηση trial Elastic ML

Στην συνέχεια θα φτιάξουμε ένα **Job** στο **Machine Learning tab** για να παρατηρήσουμε τον πληθυσμό των **events** σε κάποια από τα fields των logs όπως για παράδειγμα το **zeek.connection.state** που παράγει το **Zeek**, και τα **destination.ip** και **destination.port**.



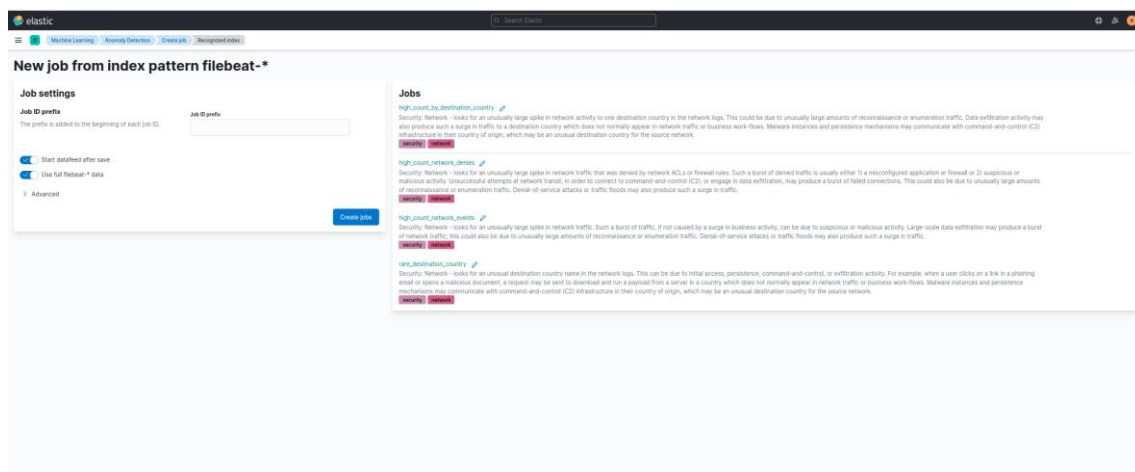
Εικόνα 13. Elastic ML – Population Job

Θα φτιάξουμε επίσης ένα **multi-metric Job** στο **Machine Learning tab** για να βρούμε πιθανές επιθέσεις **ddos** με παρόμοια λογική από τα πεδία **destination.ip** και τον αριθμό των **Events**.



Εικόνα 13. Elastic ML – Multi Metric Job

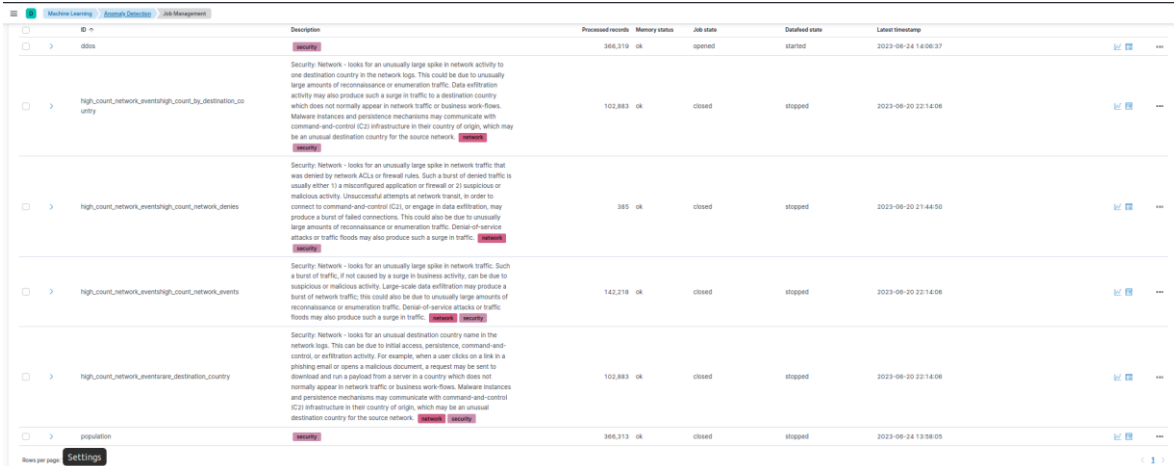
Ακόμα ενεργοποιήσαμε το έτοιμο πακέτο **Network Security** που εντοπίζει συχνά εμφανιζόμενες ύποπτες δικτυακές κινήσεις.



Εικόνα 13. Elastic ML – Network Security Job

Όλα τα **Jobs** που φτιάξαμε είναι τώρα διαθέσιμα στο tab **Machine Learning/Anomaly Detection/Job Management**.

Από εδώ μπορούμε να κάνουμε start /stop , να feedαρουμε καινούργια δεδομένα, αν κάποια δουλειά δεν τρέχει σε realtime, καθώς και να δούμε γραφικά τα αποτελέσματα ή να ρυθμίσουμε να μας έρχονται alerts σε περίπτωση ανίχνευσης ύποπτης κίνησης.



ID	Description	Processed records	Memory status	Job state	Defined state	Last timestamp
obis	Security Network - looks for an unusually large spike in network activity to one destination country in the network logs. This could be due to unusually large amounts of reconnaissance or enumeration traffic. Data exfiltration activity may also produce such a surge in traffic to a destination country which does not normally appear in network traffic or business work-flows. Malware instances and persistence mechanisms may communicate with command-and-control (C2) infrastructure in their country of origin, which may be an unusual destination country for the source network. Security	366,319	OK	opened	started	2023-08-24 16:08:37
high_count_network_events_high_count_by_destination_country	Security Network - looks for an unusually large spike in network traffic that was denied by network ACLs or firewall rules. Such a burst of denied traffic is usually either (1) a misconfigured application or firewall or (2) suspicious or malicious activity. Unsuccessful attempts at network transit, in order to connect to command-and-control (C2), or engage in data exfiltration, may produce a burst of failed connections. This could also be due to unusually large amounts of reconnaissance or enumeration traffic. Denial-of-service attacks or traffic floods may also produce such a surge in traffic. Security Security	102,883	OK	closed	stopped	2023-08-20 22:14:06
high_count_network_events_high_count_network_denies	Security Network - looks for an unusually large spike in network traffic. Such a burst of traffic, if not caused by a surge in business activity, can be due to suspicious or malicious activity. Large-scale data exfiltration may produce a burst of network traffic; this could also be due to unusually large amounts of reconnaissance or enumeration traffic. Denial-of-service attacks or traffic floods may also produce such a surge in traffic. Security Security	385	OK	closed	stopped	2023-08-20 21:44:50
high_count_network_events_high_count_network_events	Security Network - looks for an unusually large spike in network traffic. Such a burst of traffic, if not caused by a surge in business activity, can be due to suspicious or malicious activity. Large-scale data exfiltration may produce a burst of network traffic; this could also be due to unusually large amounts of reconnaissance or enumeration traffic. Denial-of-service attacks or traffic floods may also produce such a surge in traffic. Security Security	142,218	OK	closed	stopped	2023-08-20 22:14:06
high_count_network_events_are_destination_country	Security Network - looks for an unusual destination country name in the network logs. This can be due to initial access, persistence, command-and-control, or exfiltration activity. For example, when a user clicks on a link in a phishing email or opens a malicious document, a request may be sent to download and run a payload from a server in a country which does not normally appear in network traffic or business work-flows. Malware instances and persistence mechanisms may communicate with command-and-control (C2) infrastructure in their country of origin, which may be an unusual destination country for the source network. Security Security	102,883	OK	closed	stopped	2023-08-20 22:14:06
population	Security	366,313	OK	closed	stopped	2023-08-24 13:58:05

Εικόνα 13. Elastic ML – Job Management

5 Πραγματοποίηση επιθέσεων

5.1 Επανεκπομπή pcap αρχείων

Για την πραγματοποίηση των επανεκπομπών των **pcap** αρχείων, αρχικά πρέπει να εγκαταστήσουμε το λογισμικό **tcprelay** στο host μηχανήμα μας με την εντολή

```
apt-get install tcprelay
```

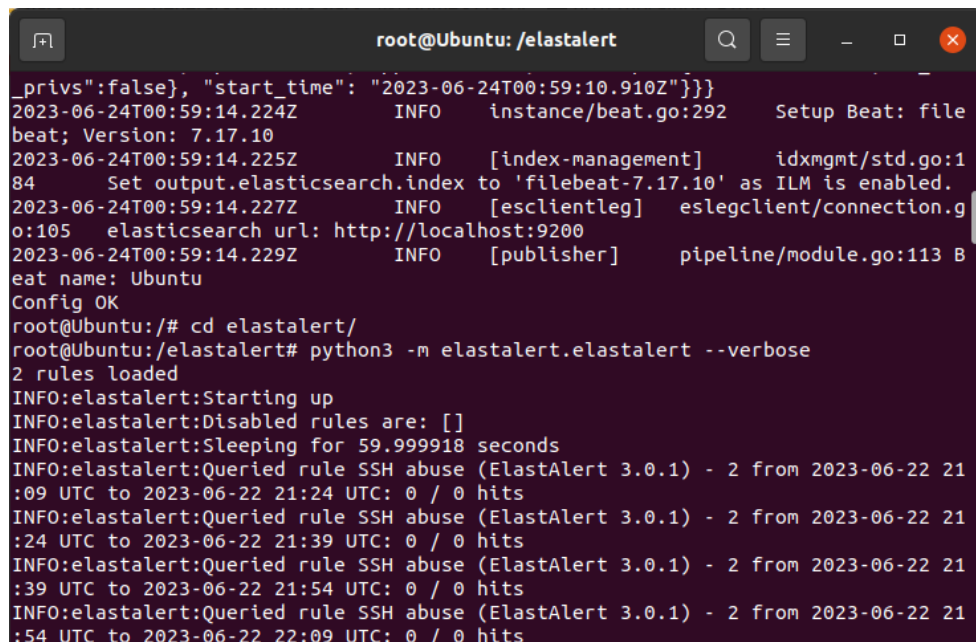
Έπειτα ξεκινάμε το Zeek container:

```
sudo docker start zeek  
sudo docker exec -it zeek bash
```

Τώρα μέσα από το **container**, ξεκινάμε όλα τα services και τρέχουμε το **ElastAlert** με τα rules που φτιάξαμε προηγουμένως:

```
service elasticsearch start  
service kibana start  
zeekctl deploy  
service filebeat start  
  
cd elastalert/  
#start elasticalert with my_rules  
python3 -m elastalert.elastalert --verbose
```

Επιβεβαιώνουμε ότι τα rules φορτωθήκαν επιτυχώς και ότι το **ElastAlert** τρέχει χωρίς errors



```
root@Ubuntu: /elastalert  
_privs":false}, "start_time": "2023-06-24T00:59:10.910Z"}}}  
2023-06-24T00:59:14.224Z INFO instance/beat.go:292 Setup Beat: file  
beat; Version: 7.17.10  
2023-06-24T00:59:14.225Z INFO [index-management] idxmgmt/std.go:1  
84 Set output.elasticsearch.index to 'filebeat-7.17.10' as ILM is enabled.  
2023-06-24T00:59:14.227Z INFO [esclientleg] eslegclient/connection.g  
o:105 elasticsearch url: http://localhost:9200  
2023-06-24T00:59:14.229Z INFO [publisher] pipeline/module.go:113 B  
eat name: Ubuntu  
Config OK  
root@Ubuntu:/# cd elastalert/  
root@Ubuntu:/elastalert# python3 -m elastalert.elastalert --verbose  
2 rules loaded  
INFO:elastalert:Starting up  
INFO:elastalert:Disabled rules are: []  
INFO:elastalert:Sleeping for 59.999918 seconds  
INFO:elastalert:Queried rule SSH abuse (ElastAlert 3.0.1) - 2 from 2023-06-22 21  
:09 UTC to 2023-06-22 21:24 UTC: 0 / 0 hits  
INFO:elastalert:Queried rule SSH abuse (ElastAlert 3.0.1) - 2 from 2023-06-22 21  
:24 UTC to 2023-06-22 21:39 UTC: 0 / 0 hits  
INFO:elastalert:Queried rule SSH abuse (ElastAlert 3.0.1) - 2 from 2023-06-22 21  
:39 UTC to 2023-06-22 21:54 UTC: 0 / 0 hits  
INFO:elastalert:Queried rule SSH abuse (ElastAlert 3.0.1) - 2 from 2023-06-22 21  
:54 UTC to 2023-06-22 22:09 UTC: 0 / 0 hits
```

Εικόνα 14. Elast Alert running

Τώρα έχοντας ένα δεύτερο **terminal** ανοιχτό από τον host τρέχουμε τις παρακάτω εντολές:

```
cd ~/Desktop/Security/pcaps-IoT/
```

```
sudo tcpreplay --intf1=enp0s3 scan-hostport-6-dec.pcap
sudo tcpreplay --intf1=enp0s3 dos-synflooding-3-dec.pcap
sudo tcpreplay --intf1=enp0s3 dos-synflooding-1-dec.pcap
sudo tcpreplay --intf1=enp0s3 mirai-udpflooding-1-dec.pcap
sudo tcpreplay --intf1=enp0s3 mitm-arp spoofing-1-dec.pcap
```

#or run all rules in my_rules folder at once:
sudo tcpreplay --intf1=enp0s3 *

```

Truncated packets: 0
Retried packets (ENOBUFS): 0
Retried packets (EAGAIN): 0
ubuntu@Ubuntu: ~/Desktop/Security/pcaps-IoT$ sudo tcpreplay --intf1=enp0s3 mirai-udpflooding-1-dec.pcap
Warning in replay.c:replay_file() line 137:
mirai-udpflooding-1-dec.pcap was captured using a snaplen of 1500 bytes. This may mean you have truncated packets.
Actual: 417863 packets (39980899 bytes) sent in 154.36 seconds
Rated: 258996.0 Bps, 2.07 Mbps, 2706.91 pps
Statistics for network device: enp0s3
Successful packets: 417863
Failed packets: 0
Truncated packets: 0
Retried packets (ENOBUFS): 0
Retried packets (EAGAIN): 0
ubuntu@Ubuntu: ~/Desktop/Security/pcaps-IoT$ sudo tcpreplay --intf1=enp0s3 *
Warning in send_packets.c:send_packets() line 644:
Unable to send packet: Error with PF_PACKET send() [637]: Message too long (errno = 90)
Warning in send_packets.c:send_packets() line 644:
Unable to send packet: Error with PF_PACKET send() [8344]: Message too long (errno = 90)
Warning in replay.c:replay_file() line 137:
mirai-udpflooding-1-dec.pcap was captured using a snaplen of 1500 bytes. This may mean you have truncated packets.

```

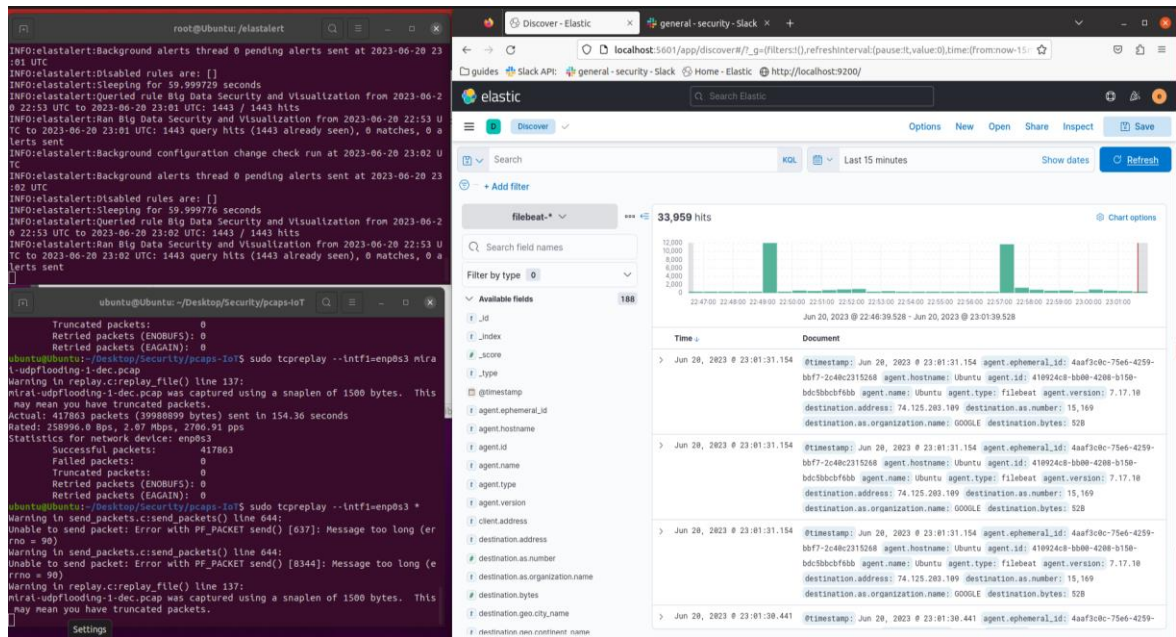
Εικόνα 15. Resend pcap with tcprelay

Τα **pcap** που επιλέξαμε περιέχουν διάφορων τύπων επιθέσεις:

File Name	Creation Date*	File Size (KB)	Target Device**	Category	Sub-category	Wireshark Rule to Filter Only Attack Packets	# Total Packets	# Attack Packets
mitm-arp spoofing-1-dec.pcap	31/05/2019	27,045	EZVIZ	Man in the Middle (MITM)	ARP Spoofing	eth.addr == f0:18:98:5e:ff:9f and (((ip.src == 192.168.0.16 and ip.dst == 192.168.0.13) or (ip.src == 192.168.0.13 and ip.dst == 192.168.0.16)) and (tcp and tcp) or (arp.src.hw_mac == f0:18:98:5e:ff:9f and (arp.dst.hw_mac == bc:1c:81:4b:ae:ba or arp.dst.hw_mac == 48:4b:ae:2c:d8:f9)))	65,768	34,855
dos-synflooding-1-dec.pcap	31/05/2019	5,815	EZVIZ	Denial of Service	SYN Flooding	ip.src == 222.0.0.0/8 and tcp.flags.syn == 1 and ip.dst == 192.168.0.13 and tcp.destination == 554 and tcp	40,788	18,703
dos-synflooding-3-dec.pcap	31/05/2019	2,548	EZVIZ	Denial of Service	SYN Flooding	ip.src == 111.0.0.0/8 and tcp.flags.syn == 1 and ip.dst == 192.168.0.13 and tcp.destination == 554 and tcp	26,334	12,538
scan-hostport-6-dec.pcap	11/07/2019	2,151	NUGU	Scanning	Host Discovery	(eth.src == f0:18:98:5e:ff:9f and arp and eth.dst == ff:ff:ff:ff:ff:ff) and frame.number < 1000	6,828	285
mirai-udpflooding-1-dec.pcap	01/08/2019	45,618	EZVIZ (performs external server (target))	Mirai Botnet	UDP Flooding	ip.dst == 210.89.164.90	417,863	404,863

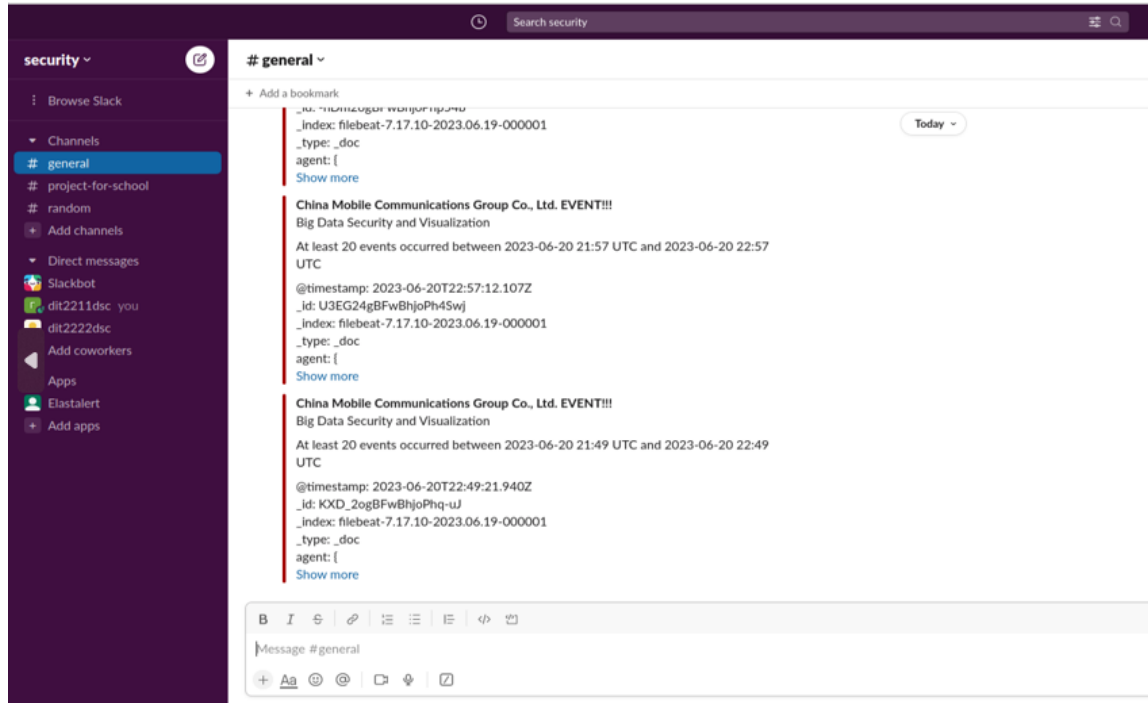
Εικόνα 16. pcap files info

5.2 Ανίχνευση και ανάλυση Εισβολών



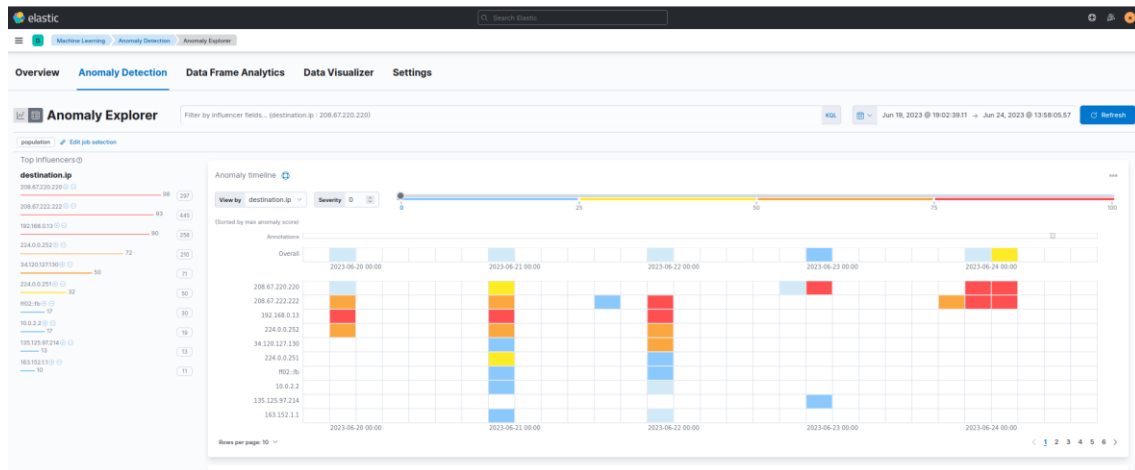
Εικόνα 17. During the attack

Έπειτα από λίγη ώρα και κατά την διάρκεια της επίθεσης μας έρχεται alert στο Slack.



Εικόνα 18. ElastAlert notification to slack

Έπειτα από τις επιθέσεις τρέξαμε μια ανάλυση μέσω του **Anomaly Detection** από το **Population Job** που φτιάξαμε στο προηγούμενο βήμα. Όπως φαίνεται από τις ακόλουθες εικόνες βρήκε αρκετές περιπτώσεις IP οι οποίες είχαν ξεπεράσει κατά πολύ την τυπική συμπεριφορά όπως φαίνεται και στις ακόλουθες εικόνες.



Εικόνα 19. ElastML – Anomaly Detection results 1

Anomalies								
Time	Severity	Detector	Found for	Influenced by	Actual	Typical	Description	Actions
> June 22nd 2023	98	count over "destination.ip"	208.67.220.220	destination.ip: 208.67.220.220	25084	20.3	⬆ More than 100x higher	
> June 22nd 2023	98	high_count over "destination.ip"	208.67.220.220	destination.ip: 208.67.220.220	25084	20.3	⬆ More than 100x higher	
> June 21st 2023	93	count over "destination.ip"	208.67.222.222	destination.ip: 208.67.222.222	18989	17.9	⬆ More than 100x higher	
> June 21st 2023	93	high_count over "destination.ip"	208.67.222.222	destination.ip: 208.67.222.222	18989	17.9	⬆ More than 100x higher	
> June 21st 2023	90	count over "destination.ip"	192.168.0.13	destination.ip: 192.168.0.13	18861	17.9	⬆ More than 100x higher	
> June 21st 2023	90	high_count over "destination.ip"	192.168.0.13	destination.ip: 192.168.0.13	18861	17.9	⬆ More than 100x higher	
> June 20th 2023	87	high_count over "destination.ip"	192.168.0.13	destination.ip: 192.168.0.13	37712	21.5	⬆ More than 100x higher	
> June 20th 2023	87	count over "destination.ip"	192.168.0.13	destination.ip: 192.168.0.13	37712	21.5	⬆ More than 100x higher	
> June 24th 2023	85	high_count over "destination.ip"	208.67.220.220	destination.ip: 208.67.220.220	15622	19.8	⬆ More than 100x higher	
> June 24th 2023	85	count over "destination.ip"	208.67.220.220	destination.ip: 208.67.220.220	15622	19.8	⬆ More than 100x higher	
> June 24th 2023	82	count over "destination.ip"	208.67.222.222	destination.ip: 208.67.222.222	13730	19.8	⬆ More than 100x higher	
> June 24th 2023	82	high_count over "destination.ip"	208.67.222.222	destination.ip: 208.67.222.222	13730	19.8	⬆ More than 100x higher	
> June 19th 2023	80	count over "destination.ip"	192.168.0.13	destination.ip: 192.168.0.13	15507	73.2	⬆ More than 100x higher	
> June 19th 2023	80	high_count over "destination.ip"	192.168.0.13	destination.ip: 192.168.0.13	15507	73.2	⬆ More than 100x higher	
> June 23rd 2023	78	count over "destination.ip"	208.67.220.220	destination.ip: 208.67.220.220	8824	19.8	⬆ More than 100x higher	
> June 23rd 2023	78	high_count over "destination.ip"	208.67.220.220	destination.ip: 208.67.220.220	8824	19.8	⬆ More than 100x higher	

Εικόνα 20. ElastML – Anomaly Detection results 2

6 Επίλογος

Από όσα εκθέσαμε παραπάνω, παρόλο που εφαρμόσαμε ένα υποτυπώδες configuration και χωρίς να χρησιμοποιήσουμε όλες τις δυνατότητες των εργαλείων, είναι εμφανές ότι ο συνδυασμός του **Zeek** ως εργαλείο **network monitoring** με το **ELK stack** δίνουν εκτεταμένες δυνατότητες συλλογής, ανάλυσης και οπτικοποίησης δεδομένων. Ιδιαίτερα στον τομέα της ασφάλειας, η πληθώρα επιλογών παραμετροποίησης του **ELK stack**, όπως τα **ElastAlert** και **Elastic ML** καθώς και η ύπαρξη εξειδικευμένων εργαλείων για ανίχνευση απειλών δίνει σαφές πλεονέκτημα σε ομάδες ασφαλείας που έχουν ως στόχο την αποτροπή κακόβουλων ενεργειών εις βάρος κάποιου οργανισμού.

7 Βιβλιογραφία

- [1] <https://www.elastic.co/blog/collecting-and-analyzing-zeek-data-with-elastic-security>
- [2] <https://github.com/Yelp/elastalert>
- [3] <https://ieee-dataport.org/open-access/iot-network-intrusion-dataset>
- [4] <https://www.elastic.co/guide/en/machine-learning/current/ml-getting-started.html>
- [5] <https://logz.io/learn/docker-monitoring-elk-stack/>
- [6] <https://devopscube.com/build-docker-image/>
- [7] <https://docs.docker.com/engine/install/ubuntu/>