Διπλωματική Εργασία

Μελέτη και κατηγοριοποίηση κυβερνοεπιθέσεων σε δημοφιλή λειτουργικά συστήματα και υπηρεσίες με τη χρήση Honeypots

Βογιατζής Μανώλης

Επιβλέπων καθηγητής: Δασυγένης Μηνάς

Εργαστήριο Ρομποτικής, Ενσωματωμένων και Ολοκληρωμένων Συστημάτων
http://arch.ece.uowm.gr

Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, 12/07/2022, Κοζάνη

Περιεχόμενα παρουσίασης

Σχετικά με την Κυβερνοασφάλεια

Με την συνεχή τεχνολογική πρόοδο, την εξέλιξη των συσκευών και την αύξηση των χρηστών στο διαδίκτυο αυξάνεται παράλληλα και ο αριθμός των κυβερνοεπιθέσεων. Κυβερνοεπίθεση είναι η κακόβουλη ενέργεια μέσω υπολογιστικών συστημάτων με σκοπό την υποκλοπή και την μη εξουσιοδοτημένη πρόσβαση στις πληροφορίες του χρήστη.

Κακόβουλοι χρήστες / Hackers

- Εγκληματικές οργανώσεις
- Ακτιβιστές
- Όφελος: Οικονομικό

Δεδομένα που υποκλέπτουν

- Κωδικοί Πρόσβασης
- Πιστωτικές Κάρτες
- Συνομιλίες, Φωτογραφίες

Τεχνικές Επίθεσης

#1 Brute Force Attack

#2 Denial-of-service (DoS)

#3 Man-in-the-middle attack

#4 Phising attack

#5 SQL injection

#6 Communication with Command-and-Control Server

#7 Spam emails

#8 Vulnerability Exploitation

Αντίμετρα Ασφάλειας διαδικτυακών συστημάτων

- Firewall
- Antivirus
- Honeypots
- Security Email Gateway
- Ανάλυση δικτυακής κίνησης (Cloud Monitor Analysis)
- Ανάλυση συμπεριφοράς χρήστη (User Behavior Analysis)
- Ανάλυση κακόβουλων προγραμμάτων (Sandbox)

Honeypot

Τα honeypot είναι συστήματα που έχουν στόχο την ανίχνευση, την ανάλυση και κάποιες φορές την εξουδετέρωση των κακόβουλων προσπαθειών. Χρησιμοποιώντας παραπλανητικά συστήματα αναγνωρίζουν, κατηγοριοποιούν και αναλύουν τις διαδικτυακές επιθέσεις.

Κατηγορίες Honeypot

- Χαμηλής αλληλεπίδρασης
- Μεσαίας αλληλεπίδρασης
- Υψηλής αλληλεπίδρασης

Σχετικές Εργασίες

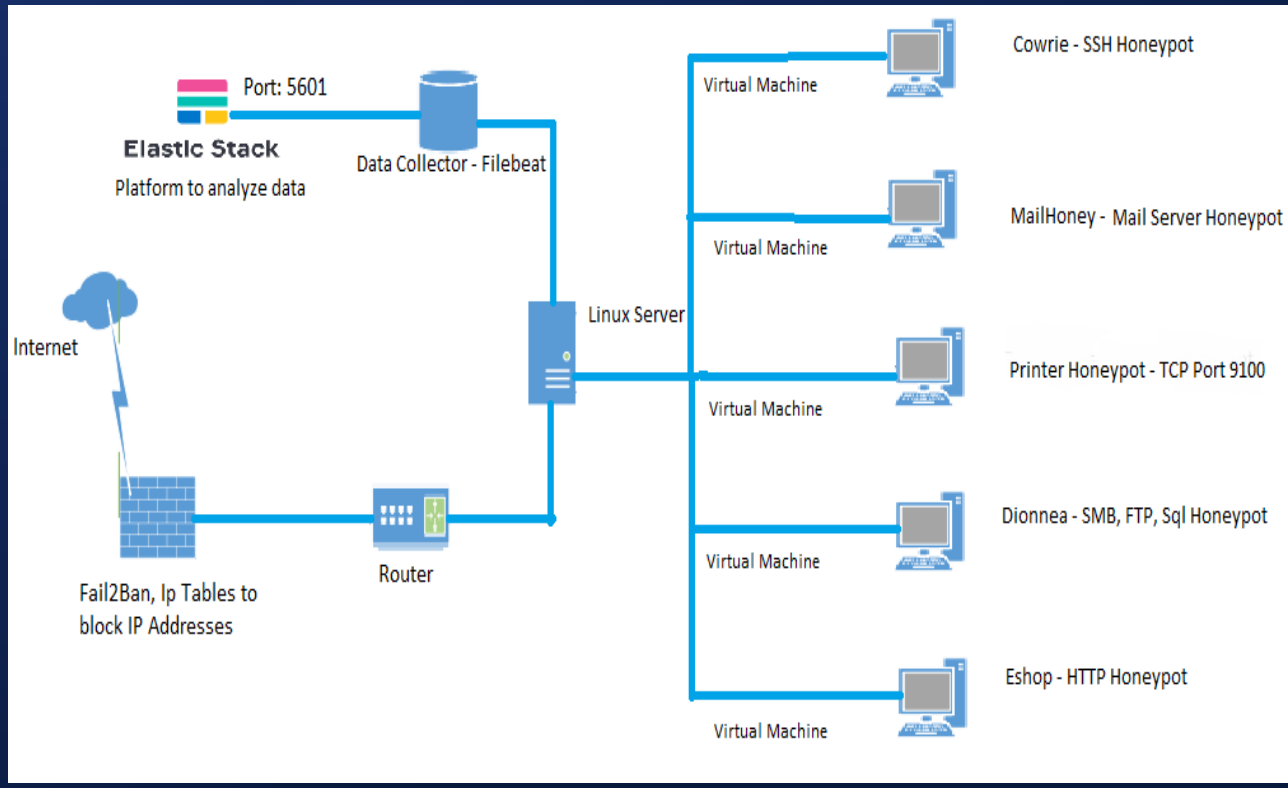
- T-Pot
- Cowrie
- Dionaea
- Glastopf
- Honeypot Kippo

Γιατί το δικό μας Honeypot είναι ξεχωριστό από τα παραπάνω?

Το υψηλής αλληλεπίδρασης Honeypot που έχουμε δημιουργήσει, αποτελεί σουίτα πολλών παραπλανητικών συστημάτων και αποσκοπεί στην αναγνώριση, στην εικονοποίηση σε διαγράμματα και στην εξουδετέρωσή των κυβερνοεπιθέσεων. Σε αντίθεση με τα υπόλοιπα Honeypot που πραγματοποιούν μόνο αναγνώριση και σπανίως εικονοποίηση των δεδομένων.

Αρχιτεκτονική του Honeypot

Το Honeypot μας αποτελείται από ένα παραπλανητικό σύστημα που δημιουργήθηκε από εμάς και ακόμη 4 δημοφιλή παραπλανητικά συστήματα. Το Filebeat συλλέγει τα δεδομένα τους και στην πλατφόρμα ELK stack εικονοποιούνται τα δεδομένα της επίθεσης. Το Fail2Ban και το IP Tables αποτελούν αντίμετρα ασφαλείας για την εξουδετέρωση συγκεκριμένων επιθέσεων όπως DoS.



Τοποθέτηση των Honeypots

Η τοποθέτηση στο δικό μας Honeypot είναι μπροστά από το τείχος προστασίας

- Τοποθέτηση μπροστά από το τείχος προστασίας
  - Για την ανίχνευση όλων των κυβερνοεπιθέσεων που δέχεται ο οργανισμός
- Τοποθέτηση πίσω από το τείχος προστασίας
  - Για την ανίχνευση των κυβερνοεπιθέσεων που δεν αποτρέπει το τείχος προστασίας
- Τοποθέτηση στο εσωτερικό δίκτυο
  - Για την ανίχνευση των κυβερνοεπιθέσεων που θα πλήξουν την παραγωγή του οργανισμού

Ανάλυση των παραπλανητικών Συστημάτων

Τα παραπλανητικά συστήματα / Honeypots δελεάζουν του κακόβουλους χρήστες να τα παραβιάσουν και συλλέγουν τα δεδομένα της επίθεσης. Έχουν υλοποιηθεί σε κοντέινερ με την βοήθεια της εφαρμογής Docker και αυτό μας διασφαλίζει ασφάλεια καθώς αν παραβιαστεί ένα παραπλανητικό σύστημα, ο κακόβουλος χρήστης δεν θα μπορέσει να συνεχίσει στις πραγματικές υπηρεσίες του διακομιστή.

#1

SSH Honeypot

Εξυπηρετείται στην πόρτα 22 και αναπαριστά ένα ψεύτικο περιβάλλον SSH στο οποίο οι κακόβουλοι χρήστες πραγματοποιούν επιθέσεις ωμής βίας.
Οι πληροφορίες που μας παρέχει είναι:

- Ονόματα χρηστών / Κωδικοί που χρησιμοποιούνται στις επιθέσεις
- Χώρα Προέλευσης της επίθεσης
- Διεύθυνση διαδικτυακού πρωτοκόλλου του κακόβουλου χρήστη

#2

Mailoney Honeypot

Εξυπηρετεί το πρωτόκολλο SMTP και αναπαριστά ένα Mail διακομιστή στον οποίο οι κακόβουλοι χρήστες στένουν ύποπτα email’s σε χρήστες.
Οι πληροφορίες που μας παρέχει είναι:

- Το θέμα, το περιεχόμενο και τα ύποπτα αρχεία των κακόβουλων email
- Χώρα Προέλευσης της επίθεσης
- Διεύθυνση διαδικτυακού πρωτοκόλλου του κακόβουλου χρήστη

Ανάλυση των παραπλανητικών Συστημάτων

#3

Dionaea Honeypot

Εξυπηρετείται πολλαπλές πόρτες όπως ftp, mysql, smb.
Οι πληροφορίες που μας παρέχει είναι:

- Ονόματα χρηστών / Κωδικοί που χρησιμοποιούνται στις επιθέσεις
- Κακόβουλα αρχεία που ανεβάζουν οι επιτιθέμενοι.
- Το payload/κώδικας της κυβερνοεπίθεσης

#4

Wordpot Honeypot

Εξυπηρετείται στην πόρτα 80 και αναπαριστά μία ιστοσελίδα που έχει την τεχνολογία Wordpress.
Οι πληροφορίες που μας παρέχει είναι:

- Ονόματα χρηστών / Κωδικοί που χρησιμοποιούνται στις επιθέσεις
- Χώρα Προέλευσης της επίθεσης
- Διεύθυνση διαδικτυακού πρωτοκόλλου του κακόβουλου χρήστη
- Εντολές SQL injection, RCE, Path traversal για την παραβίαση της ιστοσελίδας

#5

Printer Honeypot

Εξυπηρετείται στην πόρτα 9100 και αναπαριστά ένα εκτυπωτή που του στέλνουν αρχεία οι επιτιθέμενοι
Οι πληροφορίες που μας παρέχει είναι:

- Τα ονόματα ύποπτα αρχεία των κακόβουλων αρχείων
- Οι κακόβουλες αιτήσεις
- Διεύθυνση διαδικτυακού πρωτοκόλλου του κακόβουλου χρήστη

Εγκατάσταση Εργαλείων στο δικό μας Honeypot

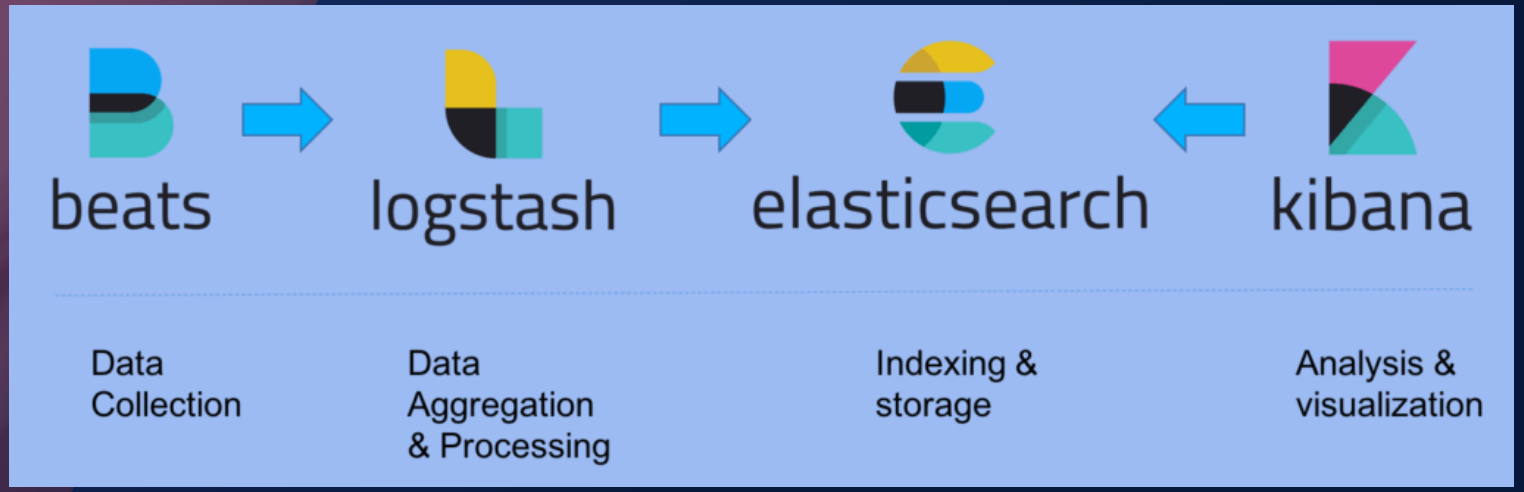
Εγκατάσταση εργαλείων για την εύρυθμη λειτουργία του Honeypot μας

- Γλώσσα προγραμματισμού Python
- Γλώσσα προγραμματισμού PHP
- Βιβλιοθήκη flask
- Τείχος προστασίας fail2ban
- Διαχειριστική πλατφόρμα Cockpit
- Υπηρεσία Filebeat
- Πρόγραμμα Docker

Συλλογή και εικονοποίηση δεδομένων

Για την συλλογή, την μεταφορά, την αποκωδικοποίηση και την οπτικοποίηση των δεδομένων χρησιμοποιήθηκε το ELK stack. Αυτό αποτελείται από 4 προγράμματα που το καθένα έχει διαφορετικό ρόλο.

- Το Filebeat ευθύνεται για την συλλογή δεδομένα από τα παραπλανητικά συστήματα
- Το Logstah ευθύνεται για τη μεταφορά των δεδομένων και την αποκωδικοποίηση τους σε ευκολονόητη μορφής τους προς τους χρήστες
- Το Elasticsearch ευθύνεται για την αποθήκευση, την αναζήτηση και την ανάλυση τεράστιων όγκων δεδομένων
- Το Kibana ευθύνεται για την οπτικοποίηση και την παρουσίαση των δεδομένων σε διαγράμματα.



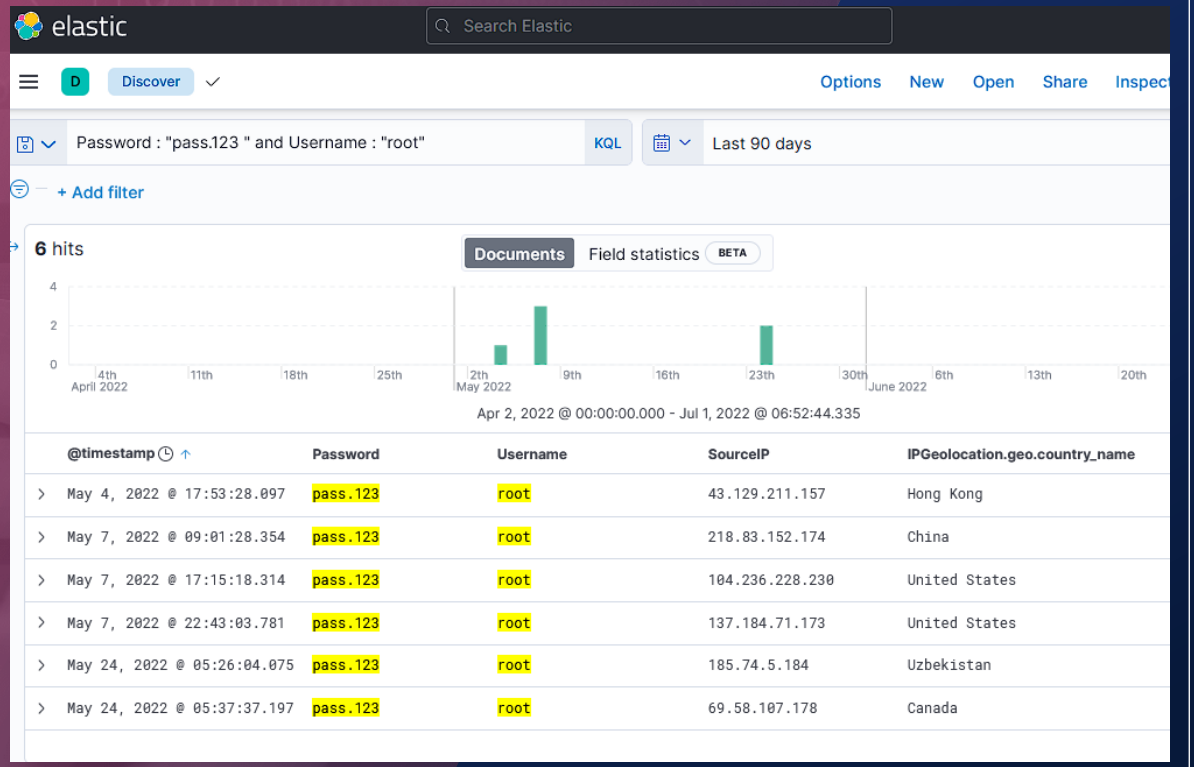
Συλλογή και εικονοποίηση δεδομένων

Το Logstash επεξεργάζεται, αποκωδικοποιεί και προωθεί στο Elasticsearch τα δεδομένα των επιθέσεων. Με την χρήση του συγκεκριμένου φίλτρου, που φαίνεται στην εικόνα, αποκωδικοποιείται το payload της επίθεσης και τα δεδομένα όπως IP Address του επιτιθέμενου, Κωδικός πρόσβασης, Όνομα χρήστη κτλ, κατηγοριοποιούνται σε στήλες. Για την υλοποίηση χρησιμοποιούνται regex εκφράσεις για την αντιστοίχιση της κάθε λέξης σε μία στήλη.

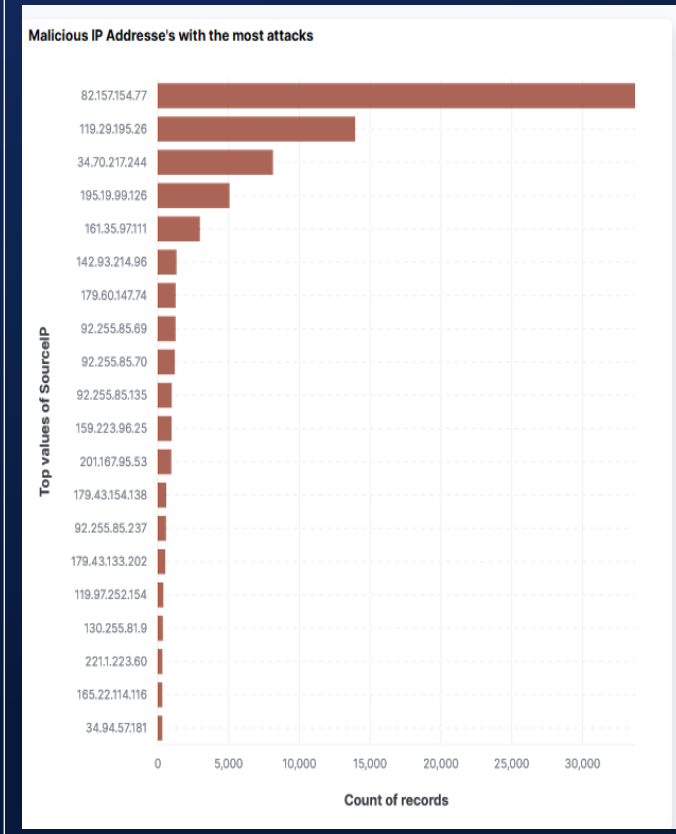
```
input { # πεδίο που ορίζει τις πόρτες διαδικτυακής κίνησης
        beats {
                port => 5044
        }
        tcp {
                port => 5000
        } }
## Add your filters / logstash plugins configuration here
filter { # πεδίο που ορίζει το φίλτρο αποκωδικοποίησης δεδομένων
      json { # ορίζουμε ότι τα δεδομένα θα έχουν json μορφή
        source => "message" # το πεδίο που θα αποκωδικοποιηθεί θα είναι το «message»
      }
      if [container][name] in "ssh_honeypot" { # λογική συνθήκη ΑΝ τα δεδομένα έρχονται από το honeypot ssh
        grok { # έκφραση regex για την εισαγωγή των δεδομένων σε στήλες
          match => { "log" => [ "\[%{DAY} %{MONTH} %{MONTHDAY} %{TIME} %{YEAR}\] %{IPV4:SourceIP} %{USERNAME:Username} %{GREEDYDATA:Password}" ] }
        }
        mutate { # φίλτρο για εισαγωγή πεδίου
         add_field => { "SrcIP" => "%{SourceIP}" } # εισαγωγή της IP address του επιτιθέμενου σε ξεχωριστό πεδίο με όνομα «SourceIP»
        }
        geoip { # φίλτρο geolocation για να βρει περισσότερες λεπτομερείς για την IP address του κακόβουλου χρήστη όπως χωρά προέλευσης
         source => "SourceIP"
         target => "IPGeolocation" #στήλη που θα εξάγει τις πληροφορίες
        } }
      if [container][name] in  "wordpot" { #  λογική συνθήκη αν τα δεδομένα προέρχονται από το honeypot wordpot
        grok { # φίλτρο για την αποκωδικοποίηση με grok μορφή
          match => { "log" => [ "\"%{GREEDYDATA:Request}\" " ] } # εξαγωγή των δεδομένων της επίθεσης στην στήλη με όνομα «requests»
        }
      if [container][name] in  "dionaea" #λογική συνθήκη
{
    date {
      match => [ "timestamp", "ISO8601" ]#μετατροπή της ημερομηνίας/ώρας σε πρότυπο ISO8601
    }
    mutate {
    rename => { # μετανομασία των στηλών
    "dst_port" => "dest_port"
    "dst_ip" => "dest_ip"
    }
    gsub => [
        "src_ip", "::ffff:", "",
        "dest_ip", "::ffff:", ""
] }
if [credentials] { # λογική συνθήκη αν υπάρχει η επίθεση περιέχει όνομα χρήστη και κωδικό
    mutate {
     add_field => { # προσθήκη  στηλών
     "username" => "%{#[credentials][username]}" # εξαγωγή του όνομα χρήστη στην στήλη "username"
     "password" => "%{#[credentials][password]}" # εξαγωγή του κωδικού πρόσβασης στην στήλη "password"
}
     remove_field => "[#credentials]" # Αν δεν περιέχει το όνομα χρήστη και τον κωδικού να γίνει απόκρυψη της στήλης "credentials"
} } }
if [container][name] in "mailoney" # λογική συνθήκη {
    date {
     match => [# "timestamp", "ISO8601" ] #μετατροπή ημερομηνίας/ώρας στο πρότυποISO
}
    mutate {
      add_field => { "dest_port" => "25" } # εισαγωγη στήλης  της πόρτας που εξυπηρετεί το honeypot
      grok {  match => { "log" => [ "\"%{GREEDYDATA:Emails}  %{IPV4:SourceIP} \" " ] }   } # έκφραση regex για την εξαγωγή των δεδομένων σε στήλες
} }
if [container][name] in  "printerhoneypot" { #  λογική συνθήκη αν τα δεδομένα προέρχονται από το honeypot printer
        grok { # φίλτρο για την αποκωδικοποίηση με grok μορφή
          match => { "message" => [ "\(%{IPV4:IPAddress} %{INTEGER:Port}\) " ] } # εξαγωγή των δεδομένων της επίθεσης στην στήλη με όνομα "IPAddres
        } } }
output { # πεδίο για την διάδοση δεδομένων στο elasticsearch
        elasticsearch {
                hosts => "X.X.X.X:9200" # ορισμός της IP address/πόρτας της υπηρεσίας Elasticsearch
              user => "user_of_logstash"  # ο λογαριασμός χρήστη που μεταφέρει τα δεδομένα από το logstash στο elasticsearch
              password => "password_of_logstashuser" } # κωδικός πρόσβασης του χρήστη
        stdout { codec => rubydebug } # εντολή να εμφανίζει τα μηνύματα λάθους και debugging στο πρόγραμμα logstash
```

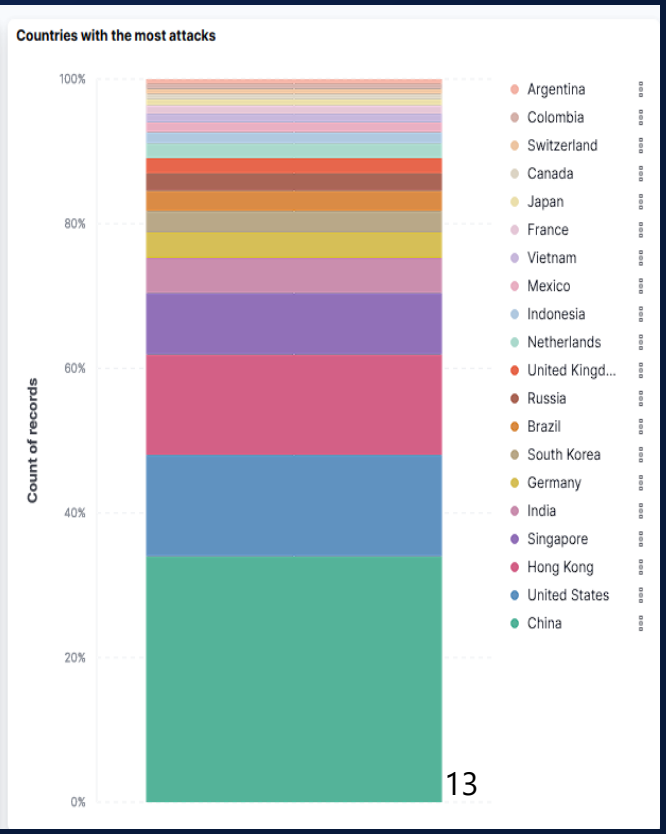
Συλλογή και εικονοποίηση δεδομένων

Το Elasticsearch αποθηκεύει τις καταγραφές των επιθέσεων και αποτελεί μία μηχανή αναζήτησης τεράστιων όγκου δεδομένων σε πραγματικό χρόνο. Χρησιμοποιώντας λογικές συνθήκες στο φίλτρο αναζήτησης μπορούμε να εστιάσουμε σε συγκεκριμένες επιθέσεις. Στην παρακάτω εικόνα γίνεται αναζήτηση των επιθέσεων, στις οποίες έχει χρησιμοποιηθεί ως κωδικός το “pass.123” και ως όνομα χρήστη το “root” .



Το Kibana αναπαριστά τα δεδομένα των επιθέσεων σε διαγράμματα, πίνακες και ιστογράμματα. Τα διαγράμματα περιέχουν στατιστικά στοιχεία για όλους του μήνες που δέχεται κυβερνοεπιθέσεις το Honeypot μας. Στην εικόνα 28 παρουσιάζονται οι IP addresses των κακόβουλων χρηστών με τις περισσότερες επιθέσεις και το ποσοστό των επιθέσεων πού προέρχεται από κάθε χώρα.





Printer Honeypot

Η υλοποίηση του Printer Honeypot πραγματοποιήθηκε για την ανίχνευση των επιθέσεων που αφορούν τους εκτυπωτές. Εξυπηρετείται στην πόρτα 9100 στο πρωτόκολλο IPP (Internet Printing Protocol) και η γλώσσα προγραμματισμού που αναπτύχθηκε είναι η Python. Το Honeypot υλοποιήθηκε σε κοντέινερ, για αυτό δημιουργήθηκε κατάλληλο αρχείο με το όνομα Dockerfile. Ο κώδικας από το παραπάνω αρχείο φαίνεται στην εικόνα.

```
FROM python

ADD . /honeyprint

WORKDIR /honeyprint

RUN cd /usr/local/lib/python*/site-packages/pkipplib/ -w pkipplib.py

EXPOSE 9100

CMD ["python3","server.py", "-i", "172.17.0.4", "-p", "9100"]
```

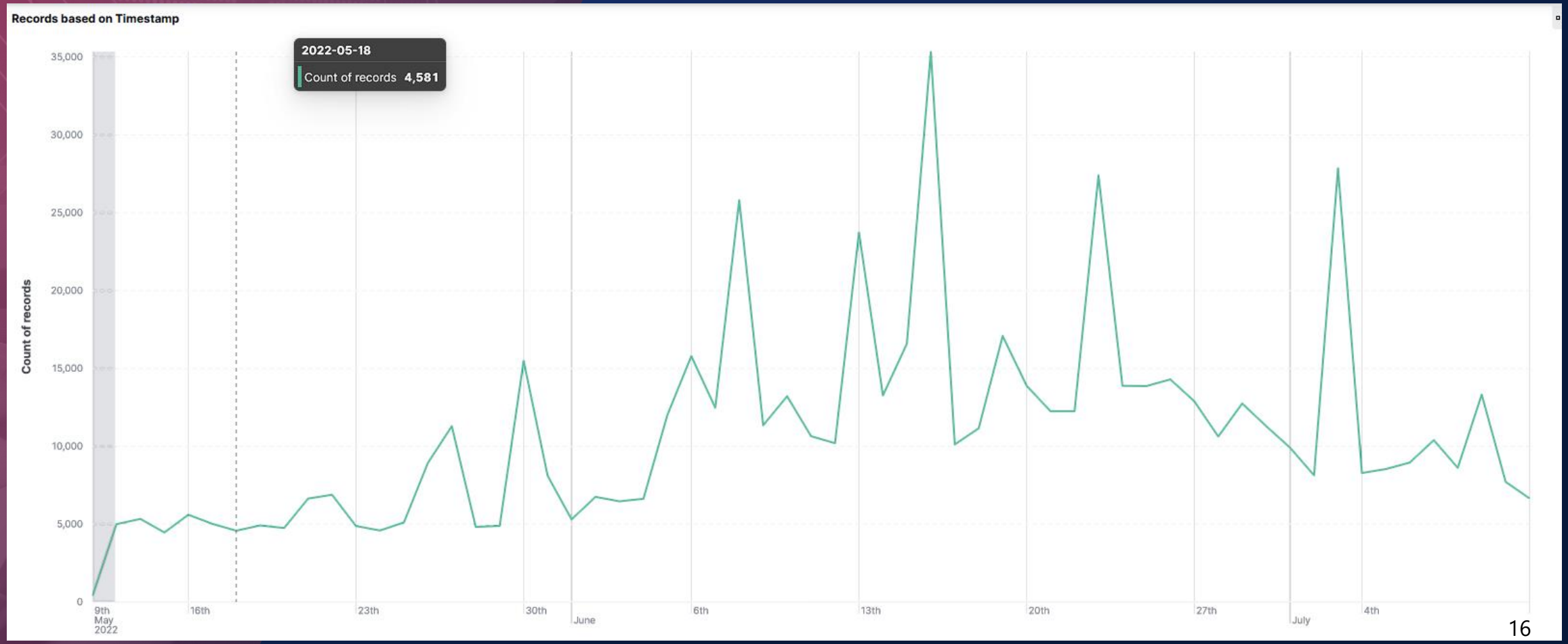
Printer Honeypot

Στην υλοποίηση του printer Honeypot χρησιμοποιήθηκαν βιβλιοθήκες της Python όπως η “pkipplib” και η “gevent”. Η “pkipplib” μετατρέπει τα αιτήματα IPP (Internet Printing Protocol) σε μορφή τέτοια ώστε στην συνέχεια να σταλούν στο κοντέινερ. Η βιβλιοθήκη “gevent” διαχειρίζεται τα αιτήματα που δέχεται και τα συγχρονίζει με το διακομιστή.

```
import sys # Εισαγωγή βιβλιοθηκών
import logging
import argparse
from pkipplib import pkipplib # Εισαγωγή βιβλιοθηκών

from gevent.server import StreamServer # Εισαγωγή βιβλιοθηκών

class PrintServer(object):

    def __init__(self):
        pass

    def handle(self, sock, address): # function για την διαχείρηση των αιτημάτων
        print(address) #εκτύπωση IP Address
        data = sock.recv(8192) # το μέγιστο μέγεθος που μπορεί να επεξεργαστεί το συστημα είναι 8192 bytes
        print(repr(data)) #εντολή για την ωραία εμφάνιση των δεδομένων
        try:
            body = data.split('\r\n\r\n', 1)[1] # εντολή για να μην είναι τα δεδομένα σε μία γραμμή
        except IndexError:
            body = data
        request = pkipplib.IPPRequest(body) #μετατροπη του αιτήματος σε μορφή ικανή για να κατανοήσει ο διακομιστής εκτυπωτής
        request.parse()
        print(request) #εκτύπωση αιτήματος
        request = pkipplib.IPPRequest #μετατροπη του αιτήματος σε μορφή ικανή για να κατανοήσει ο διακομιστής εκτυπωτής
        request.operation["attributes-charset"] = ("charset", "utf-8") #ορίζεται η μορφή του αιτήματος που είναι utf-8
        request.operation["attributes-natural-language"] = ("naturalLanguage", "en-us") #ορίζεται η γλώσσα του αιτήματος
        sock.send(request.dump())

    def get_server(self, host, port):
        connection = (host, port) #IP Address και πόρτα
        server = StreamServer(connection, self.handle)
        return server

if __name__ == "__main__": # για να εκτελεστούν τα modules "pkipplib" και "gevent"
    print_address="172.17.0.4" # IP Address του συστήματος
    print_port=9100 # Πόρτα του συστήματος
    parser.add_argument('-i','--serveraddress', const='172.17.0.4', required=False) #ορισμός της παραμέτρου "-i"
    parser.add_argument('-p','--port', const=9100, required=False) #ορισμός της παραμέτρου "-p"
    args = vars(parser.parse_args()) #παρσάρει τις νέες παραμέτρους αν πληκτρολογηθούν αλλιώς συνεχίζει με τα δεδομένα που έχει ως μεταβλητές στο "print_address" και στο "print_port"
    if args['serveraddress']:
            print_address=args['serveraddress']
    if args['port']:
            print_port=int(args['port'])

    # Start print server
    ps = PrintServer()
    print_server = ps.get_server(print_address, print_port)
    print(f'Network Printer')
    try:
        print_server.serve_forever() # module απο την βιβλιοθήκη gevent
    except KeyboardInterrupt as e: #να σταματάει το σύστημα οταν πατηθεί κάτι απο το πληκτρολόγιο
        print('Corrupted')
        sys.exit(0) #έξοδος απο το σύστημα.
```

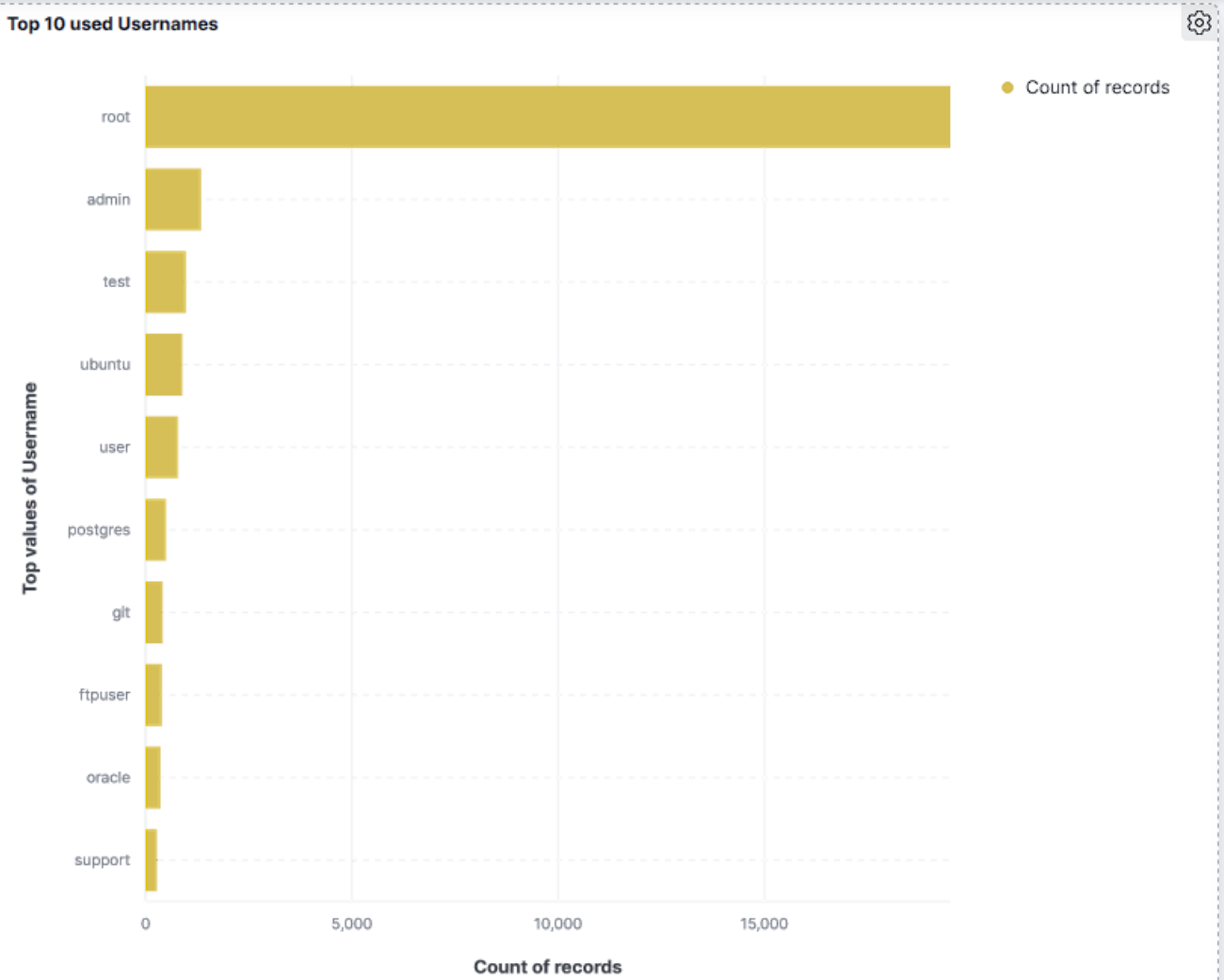
Αποτελέσματα

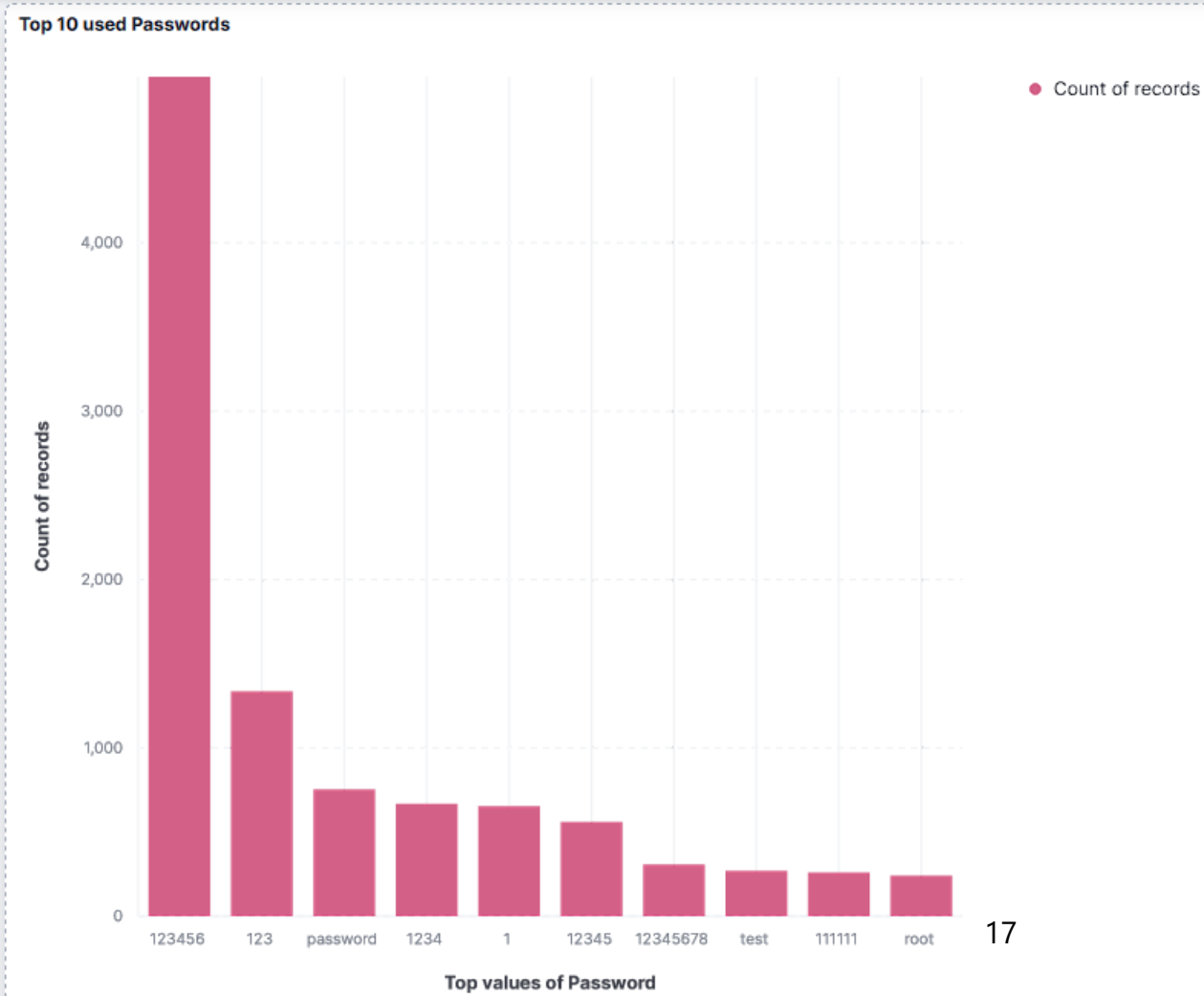
Κατά τη φάση κατασκευής και ανάπτυξης του Honeypot, πολλοί επιτιθέμενοι άρχισαν να πραγματοποιούν κυβερνοεπιθέσεις σε αυτό προκειμένου να ανακαλύψουν τις υπηρεσίες που είναι εκτεθειμένες και να τις παραβιάσουν. Το υψηλής αλληλεπίδρασης Honeypot ήταν σε πλήρη λειτουργία από τη 1 Μαρτίου 2022 έως τη 1 Ιουνίου 2022 και οι επιθέσεις που δέχτηκε φτάνουν κοντά στις 500.000. Το παρακάτω διάγραμμα παρουσιάζει τις επιθέσεις που δέχεται κάθε μέρα το Honeypot μας.



Αποτελέσματα

Κατά την καταγραφή των επιθέσεων, με την βοήθεια του Logstash βάλαμε σε στήλες τους κωδικούς πρόσβασης και τα ονόματα χρηστών που δημιουργήθηκαν. Στο αριστερό διάγραμμα παρουσιάζονται οι 10 πιο συχνοί κωδικοί και στο δεξί παρουσίαζονται τα 10 πιο συχνά ονόματα χρηστών.





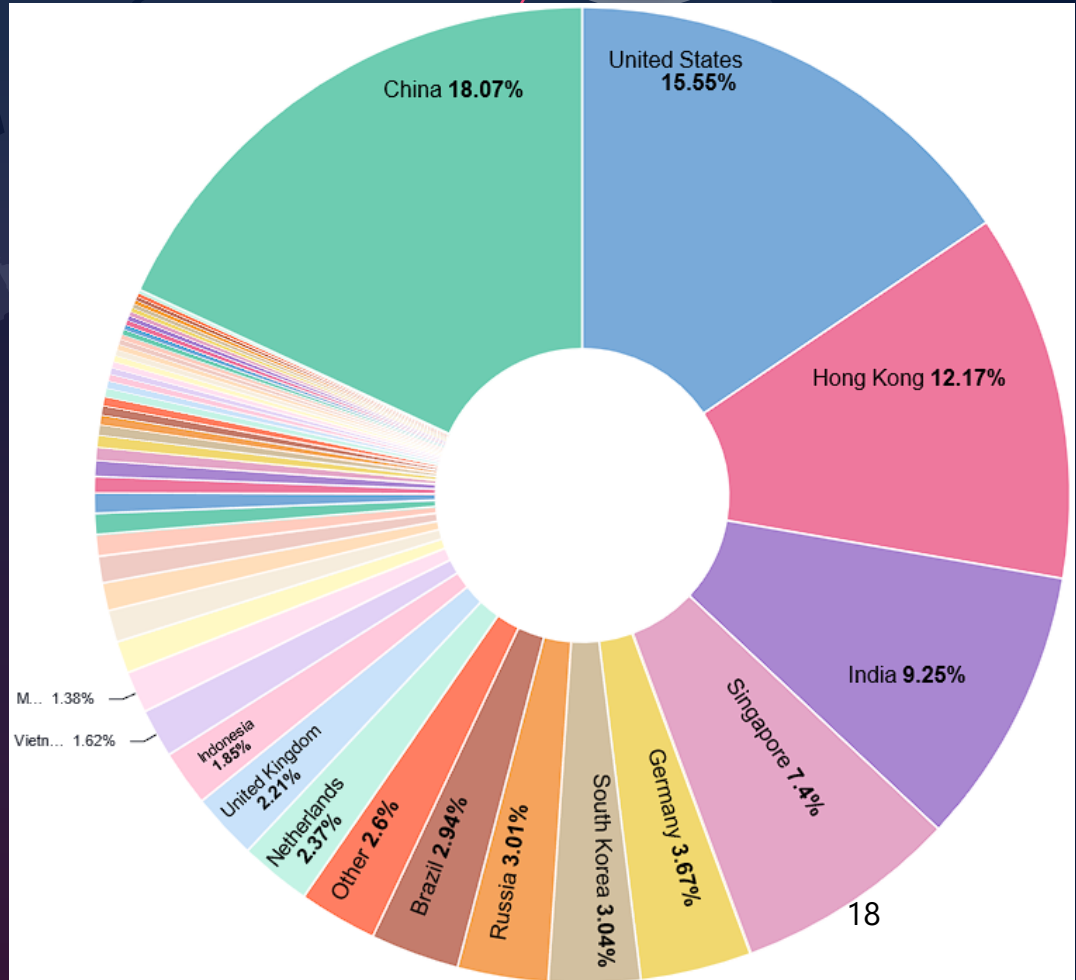
Αποτελέσματα – Οι χώρες προέλευσης των επιθέσεων

ΚΙΝΑ
18%

ΗΠΑ
18%

ΓΕΡΜΑΝΙΑ
3%

ΒΡΑΖΙΛΙΑ
2%



Συμπεράσματα και Μελλοντικές Επεκτάσεις

Στόχος της διπλωματικής εργασίας είναι η δημιουργία ενός υψηλής αλληλεπίδρασης Honeypot, το οποίο θα ανιχνεύσει και θα αποτρέψει τις κυβερνοεπιθέσεις στα υπολογιστικά συστήματα μιας εταιρίας ή ενός οργανισμού και θα βοηθήσει τους ερευνητές στην καλύτερη ανάλυση των διαδικτυακών απειλών.

Το συγκεκριμένο εργαλείο έχει δυνατότητες ανάπτυξης προσθέτοντας περισσότερη τεχνητή νοημοσύνη. Με την εφαρμογή κανόνων, που ανιχνεύουν την ύποπτη δραστηριότητα από τους χρήστες και ορίζουν την κρισιμότητα της επίθεσης, το Honeypot μετατρέπεται σε SOC (Security Operation Center) και αυξάνεται η ασφάλεια του δικτύου σε έναν οργανισμό.

Σας ευχαριστώ!
Καλό καλοκαίρι!