Ev Tipi Tehdit İstihbaratı

written by Mert SARICA | 1 October 2019
If you are looking for an English version of this article, please visit here.

Yazılarımı okuyanlarınız, Esaretten Kaçış başlıklı yazımda güvenliğini kendinizin sağlayabildiği, güvenlik özellikleri ile dopdolu bir yönlendirici (router) kullanmanın avantajlarından büyük bir mutlulukla bahsettiğimi hatırlayacaklardır. Yazıda da bahsettiğim üzere DNS trafiğini şifreli (Dns over HTTPS – DoH) hale getirmek için dnscrypt-proxy aracını kullanmaya başlamıştım.

Termostatların akıllandığı (smart), akıllı televizyonların kameralarla donatıldığı, elektrikli su ısıtıcılarının, ütülerin casuslaştırıldığı günümüzde, ev ağımıza bağlı olup internete bağlanan güvensiz nesneler (IoT), enfekte olmuş, zararlı yazılım barındıran sistemler, cihazlar güvenliğimiz, mahremiyetimiz için büyük risk teşkil ediyorlar. Hacklenmiş, enfekte olmuş, arka kapı içeren ev ağımızdaki sistemleri nasıl tespit edebileceğim üzerine düşünürken dnscrypt-proxy aracı sayesinde ev ağına bağlı tüm sistemler, cihazlar, aygıtlar tarafından gerçekleştirilen DNS isteklerini de kayıt altına alabileceğimi hatırladım.

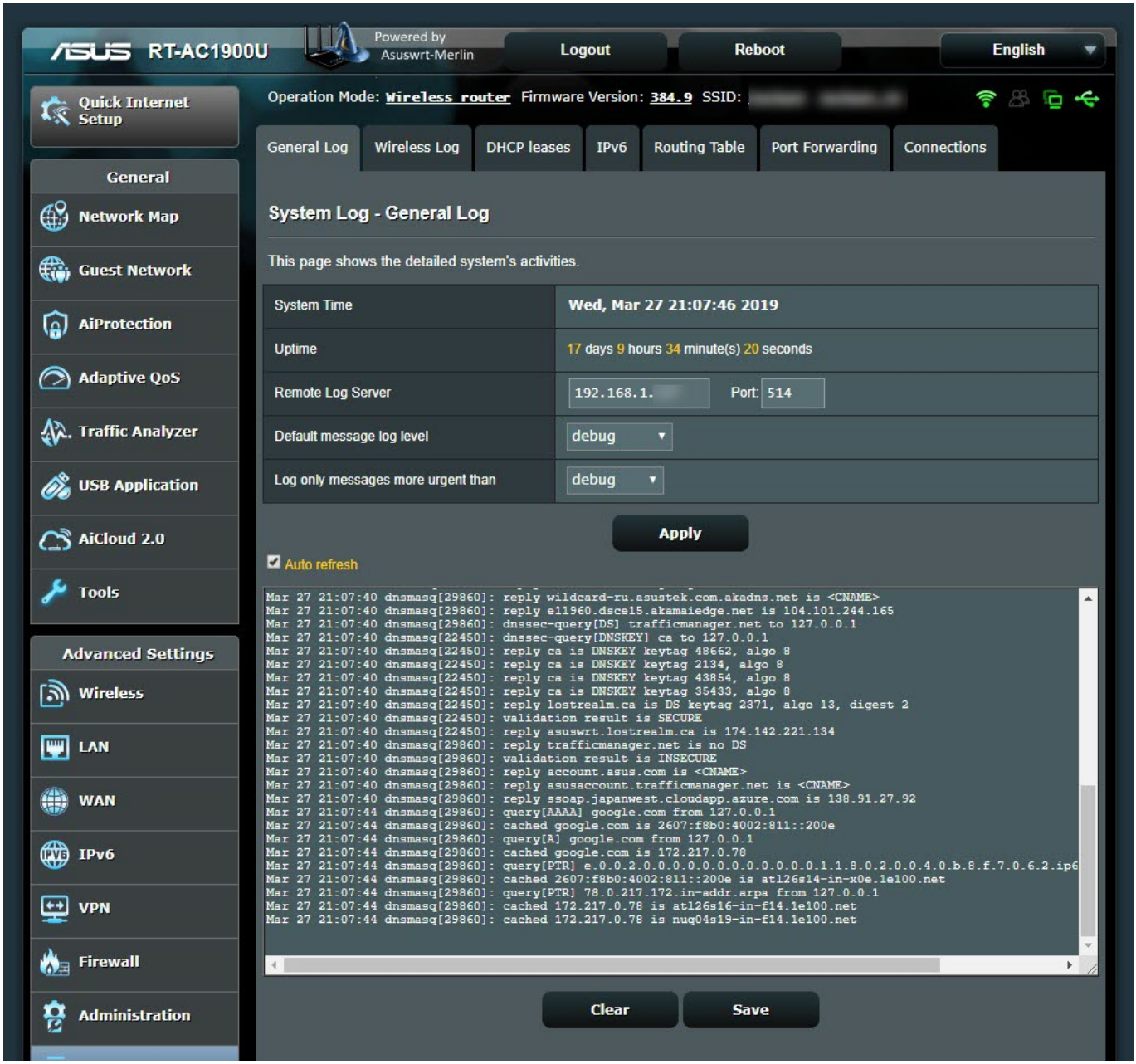
DNS isteklerini kayıt altına alabildiğim noktada Open Threat Exchange (OTX), Critical Stack gibi siber tehdit istihbaratı servislerinden faydalanarak bu DNS isteklerinde yer alan, alan adlarını ve ip adreslerini bu servislere sorarak ev ağımdaki zararlı sistemleri tespit edebilirdim. Vakit kaybetmeden bu fikrimi hayata geçirmek için ihtiyaç listesi üzerine düşünmeye başladım.

İlk olarak elimin altında bulunup bu gibi durumlarda her daim yardımıma koşan Mini-PC'imde çalışan Ubuntu işletim sistemi üzerine syslog-ng paketini kurmaya karar verdim. Paketi kurduktan sonra gelen dns isteklerini /var/log/dns-sys/gönderenin-ip-adresi klasörü altındaki tarih.log dosyasına kayıt edecek şekilde ayarladım ve /etc/syslog-ng/conf.d/dns-sys.conf dosyasına kayıt ettim.

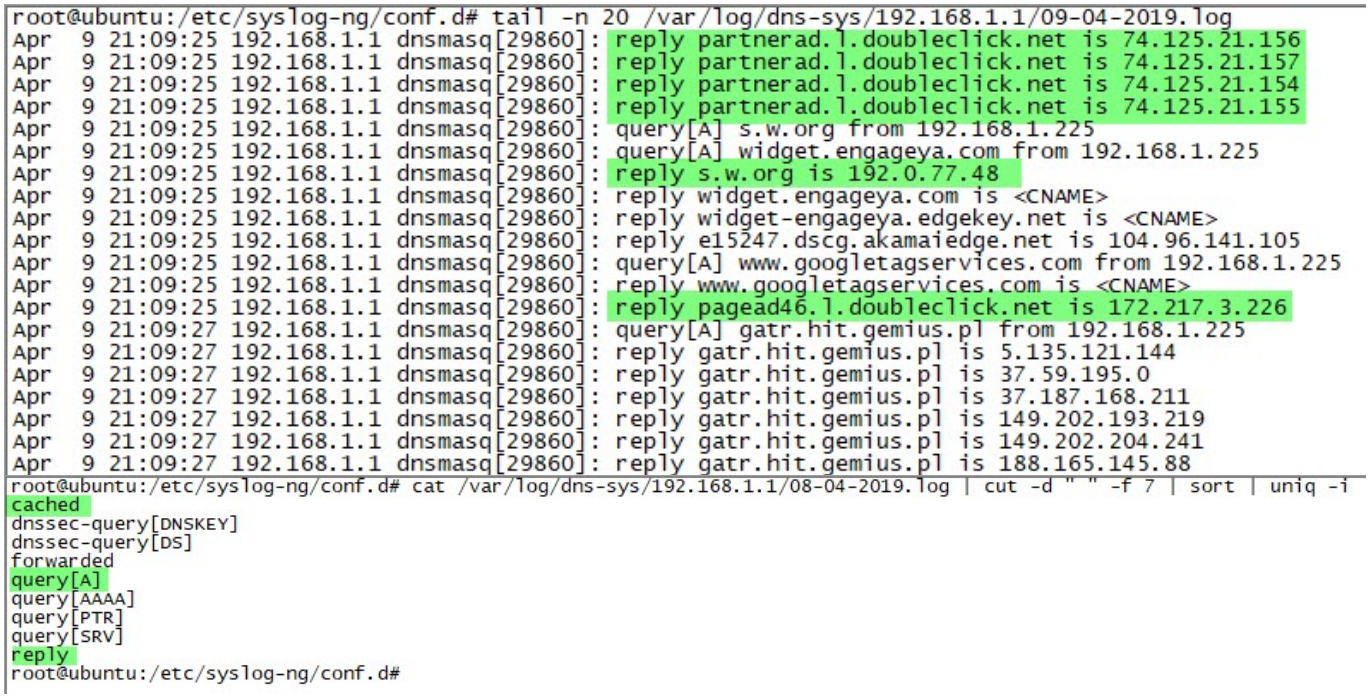
```
root@ubuntu:/etc/syslog-ng/conf.d# ls
dns-sys.conf
root@ubuntu:/etc/syslog-ng/conf.d# cat dns-sys.conf
################################################
options {
        create_dirs(yes);
        perm(0640);
        dir_perm(0750);
};


################################################
source s_net {
                tcp(ip(0.0.0.0) port(514));
                udp(ip(0.0.0.0) port(514));
};

################################################
destination d_host-specific {
        file("/var/log/dns-sys/$HOST/$DAY-$MONTH-$YEAR.log");
};

filter f_cached { match("cached"); };                    # Filter regex keyword cached
filter f_query  { match("query");  };                    # Filter regex keyword query
filter f_reply  { match("reply");  };                    # Filter regex keyword reply

log {
        source(s_net);
        filter(f_cached);
        destination(d_host-specific);
};

log {
        source(s_net);
        filter(f_query);
        destination(d_host-specific);
};

log {
        source(s_net);
        filter(f_reply);
        destination(d_host-specific);
};
```

Sonraki adımda dnscrypt-proxy aracının dns isteklerini yönlendiricinin syslog'una kayıt etmesi için /jffs/configs/dnsmasq.conf.add dosyasına log-queries satını ekledim. Ardından yönlendiricinin bu istekleri syslog sayfasında göstermesini sağlamak için Default message log level ve Log only messages more urgent than değerlerini debug olarak ayarladım ve bu mesajları Ubuntu üzerinde çalışan syslog-ng uygulamasına yönlendirmek için Remote Log Server değerini Ubuntu'nun ip adresi olarak tanımladım.

```
mert@RT-AC1900U-6610:/jffs/configs# cat dnsmasq.conf.add
no-resolv
log-queries
server=127.0.0.1#65053
mert@RT-AC1900U-6610:/jffs/configs#
```



Syslog-ng kayıtlarını teker teker incelemeye ve tehdit istihbaratı adına hangi tür kayıtlara odaklanmam gerektiğine bakmaya başladım. Kayıtlarda yer alan query[A], cached ve reply bilgilerden faydalanabileceğimi öğrendikten sonra bu kayıtları OTX ile entegre çalışabilen Security Onion'a gönderebileceğimi düşündüm. Security Onion'un 16.04.5.6 işletim sistemini kurup çalıştırdıktan sonra logstash servisinin (so-logstash) bir türlü çalışmadığını farkettim. Üzerinde debelenmeme rağmen başarılı olamadıktan sonra alternatif yollar üzerine araştırma yapmaya başladım.

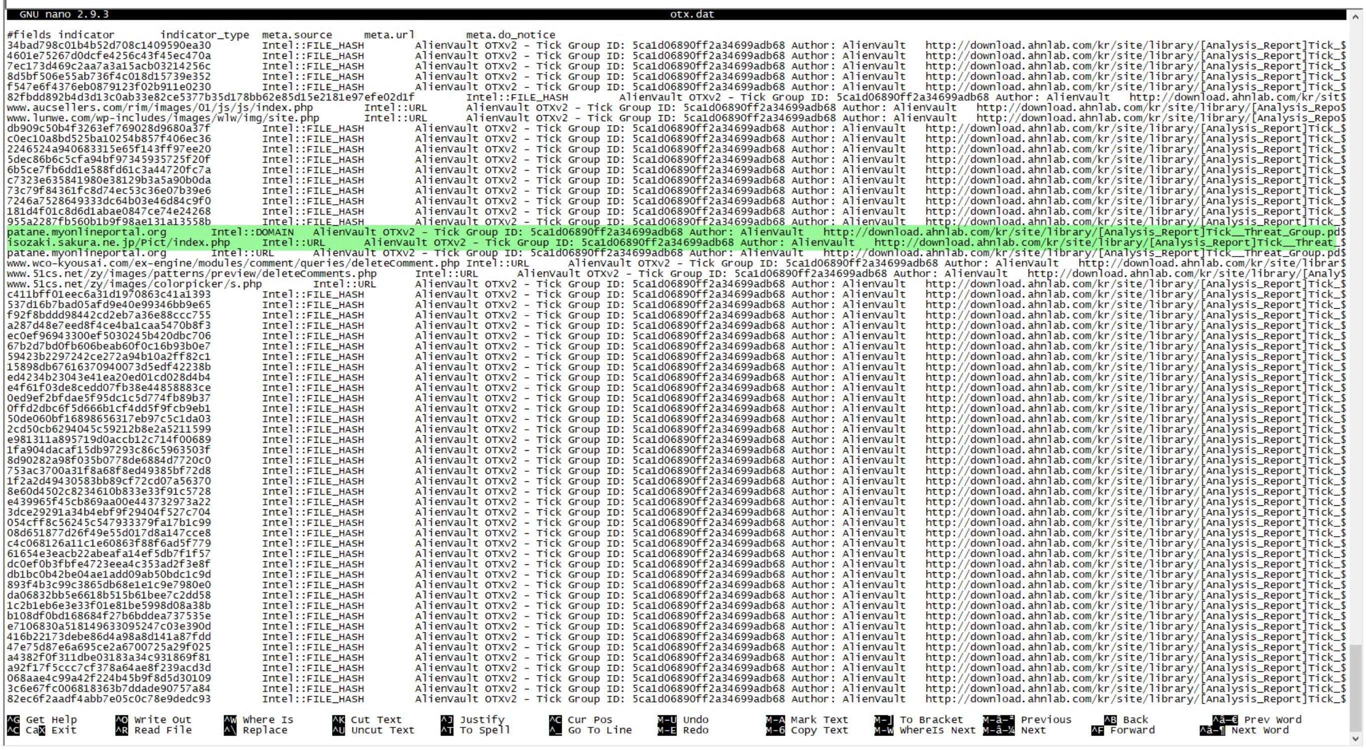


Twitter üzerinden ELK kurmam gerektiği ile ilgili bir mesaj paylaştığımda bulut ve hazır ELK sistemlerinden faydalanabileceğime dair mesajlar aldım. Ubuntu işletim sistemine ELK mı kursam yoksa bulut bir sistemden mi faydalansam derken Grok filter ve Translate filter eklentilerine sahip Logstash'in bu iş için biçilmiş kaftan olduğunu öğrendim.

Security Onion – OTX entegrasyonu için geliştirilmiş olan securityonion-otx betik dosyasını kendi ihtiyaçlarım doğrultusunda düzenlemeye başladım. bro-otx dosyası saat başı OTX'ten tehdit istihbaratı bilgisini /etc/logstash/ls-otx/otx.dat dosyasına kaydetmek için ayarladım. OTX.py dosyasını da her saatin 5. dakikasında otx.dat dosyasındaki zararlı URL ve DOMAIN kayıtlarından sadece alan adı bilgilerini alıp Translate filter tarafından

okunacak olan /etc/logstash/translate/OTX.yaml dosyası olarak kayıt etmesini sağladım.

```
root@ubuntu:/etc/cron.d# cat bro-otx
# /etc/cron.d/bro-otx
#
# crontab entry to manage Bro OTX pulse updates

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

0 * * * * root python /etc/logstash/ls-otx/bro-otx.py >> /var/log/bro-otx.log 2>&1
root@ubuntu:/etc/cron.d# cat ls-otx
# /etc/cron.d/bro-otx
#
# crontab entry to create Logstash dictionary from OTX file

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

*/5 */1 * * * root python /etc/logstash/ls-otx/OTX.py >> /var/log/ls-otx.log 2>&1
root@ubuntu:/etc/cron.d#
```

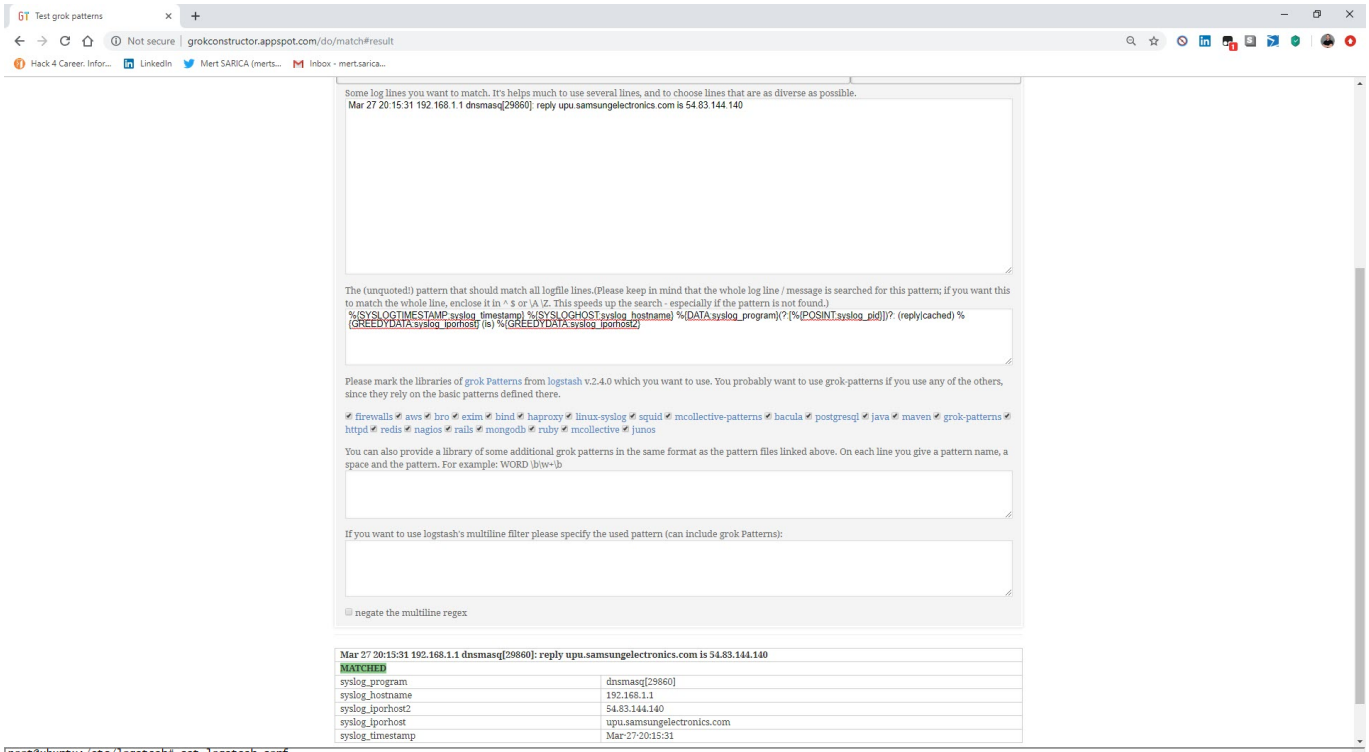


```
root@ubuntu:/etc/logstash/ls-otx# cat OTX.py
#!/usr/bin/env python
# -*- coding: utf-8 -*-
# OTX to Logstash Dictionary Script
# Author: Mert SARICA
# E-mail: mert [ . ] sarica [ @ ] gmail [ . ] com
# URL: https://www.mertsarica.com
#
# Credit: https://raw.githubusercontent.com/TravisFSmith/MyBroElk/master/maliciousIP.py

import re
debug = 0

def writeYAML():
        fname = "/etc/logstash/ls-otx/otx.dat"
        yamlFile = open('/etc/logstash/translate/OTX.yaml','w')
        with open(fname) as html:
                cti = []
                for line in html.readlines():
                        line = re.sub('\\r|\\n','',line)
                        if line.find("Intel::DOMAIN") >= 0:
                                try:
                                        line = line.split("\t")[0]
                                        if line not in cti:
                                                cti.append(line)
                                                if debug:
                                                        print line.split("\t")[0]
                                                yamlFile.write("\"" + line + "\": \"YES\"" + "\n")
                                except:
                                        continue

                        if line.find("Intel::URL") >= 0:
                                try:
                                        line = line.split("\t")[0]
                                        line = line.split("/")[0]
                                except:
                                        line = line.split("\t")[0]

                                try:
                                        line = line.split(":")[0]
                                        if line not in cti:
                                                cti.append(line)
                                                if debug:
                                                        print line
                                                yamlFile.write("\"" + line + "\": \"YES\"" + "\n")
                                except:
                                        if line not in cti:
                                                cti.append(line)
                                                if debug:
                                                        print line
                                                yamlFile.write("\"" + line + "\": \"YES\"" + "\n")

                yamlFile.close()

if __name__=="__main__":
        writeYAML()
root@ubuntu:/etc/logstash/ls-otx#
```

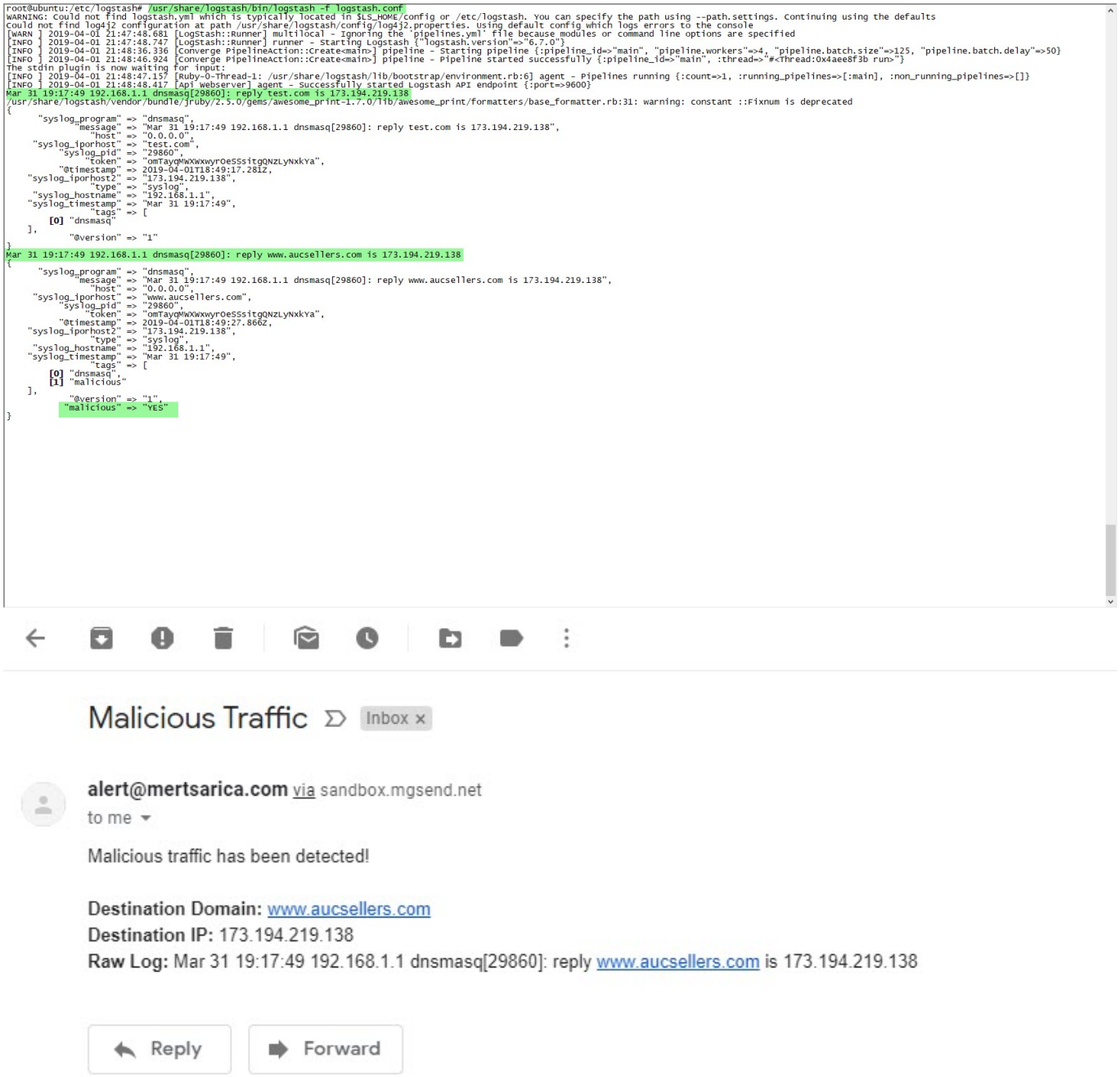
```
root@ubuntu:/etc/logstash/translate# ls
OTX.yaml
root@ubuntu:/etc/logstash/translate# head -n 10 OTX.yaml
"www.aucsellers.com": "YES"
"www.lunwe.com": "YES"
"patane.myonlineportal.org": "YES"
"isozaki.sakura.ne.jp": "YES"
"www.wco-kyousai.com": "YES"
"www.51cs.net": "YES"
"www6.intarnetservice.com": "YES"
"www.webmailerservices.com": "YES"
"go-trust.webmailerservices.com": "YES"
"www.adobeservice.net": "YES"
root@ubuntu:/etc/logstash/translate#
```

Logstash'un ayar dosyası (logstash.conf) üzerinde syslog-ng ile kayıt altına alınan DNS kayıtlarını Grok filtresi ile okuyan ve Translate filtresi ile burada yer alan ip adreslerinden veya alan adlarından herhangi birinin OTX.yaml dosyasında olması durumunda e-posta ile alarm gönderen tanımlamaları yaptım. Ardından Logstash'i yeniden başlatıp OTX.yaml dosyasında yer alan www[.]aucsellers[.]com adresine nslookup yaptığımda alarmın başarıyla

üremesini ve bana e-posta ile gönderilmesini sağlamış kısaca ev tipi tehdit istihbaratı servisini başarıyla hayata geçirmiş oldum. :)




```
root@ubuntu:/etc/logstash# cat logstash.conf
input {
  # stdin { type => syslog }
  file {
    path           => "/var/log/dns-sys/192.168.1.1/*.log"
    start_position => "beginning"
  }
}

filter {
  grok {
    match => { "message" => "%{SYSLOGTIMESTAMP:syslog_timestamp} %{SYSLOGHOST:syslog_hostname} %{DATA:syslog_program}(?:\[%{POSINT:syslog_pid}\])?: (reply|cached) %{GREEDYDATA:syslog_iporhost} (is) %{GREEDYDATA
:syslog_iporhost2}" }
    add_tag => "dnsmasq"
  }
  grok {
    match => { "message" => "%{SYSLOGTIMESTAMP:syslog_timestamp} %{SYSLOGHOST:syslog_hostname} %{DATA:syslog_program}(?:\[%{POSINT:syslog_pid}\])?: (query\[A\]) %{GREEDYDATA:syslog_iporhost} (from) %{GREEDYDATA
:syslog_queryfrom}" }
    add_tag => "dnsmasq"
  }
  translate {
    field => "syslog_iporhost"
    destination => "malicious"
    dictionary_path => '/etc/logstash/translate/OTX.yaml'
    add_tag => "malicious"
  }
  translate {
    field => "syslog_iporhost2"
    destination => "malicious"
    dictionary_path => '/etc/logstash/translate/OTX.yaml'
    add_tag => "malicious"
  }
  mutate {
    remove_tag => ["_grokparsefailure"]
  }
  if "dnsmasq" not in [tags] {
    drop { }
  }
}

output {
  stdout {
    codec => rubydebug
  }
  if [malicious] == "YES" and [syslog_iporhost2] {
   email {
        address => "127.0.0.1"
        from => "alert@mertsarica.com"
        htmlbody => "Malicious traffic has been detected!<br/><br/>
                     <b>Destination Domain: </b>%{syslog_iporhost}<br/>
                     <b>Destination IP: </b>%{syslog_iporhost2}<br/>
                     <b>Raw Log: </b>%{message}"
        port => 25
        subject => "Malicious Traffic"
        to => "mert.sarica@gmail.com"
        use_tls => false
   }
  } else if [malicious] == "YES" and [syslog_queryfrom] {
   email {
        address => "127.0.0.1"
        from => "alert@mertsarica.com"
        htmlbody => "Malicious traffic has been detected!<br/><br/>
                     <b>Source IP: </b>%{syslog_queryfrom}<br/>
                     <b>Destination IP or Domain: </b>%{syslog_iporhost}<br/>
                     <b>Raw Log: </b>%{message}"
        port => 25
        subject => "Malicious Traffic"
        to => "mert.sarica@gmail.com"
        use_tls => false
   }
```



Bir sonraki yazıda görüşmek dileğiyle herkese güvenli günler dilerim.