

Relazione: Interpretazione di Dati HTTP e DNS per Isolare Attori di Minacce

Introduzione

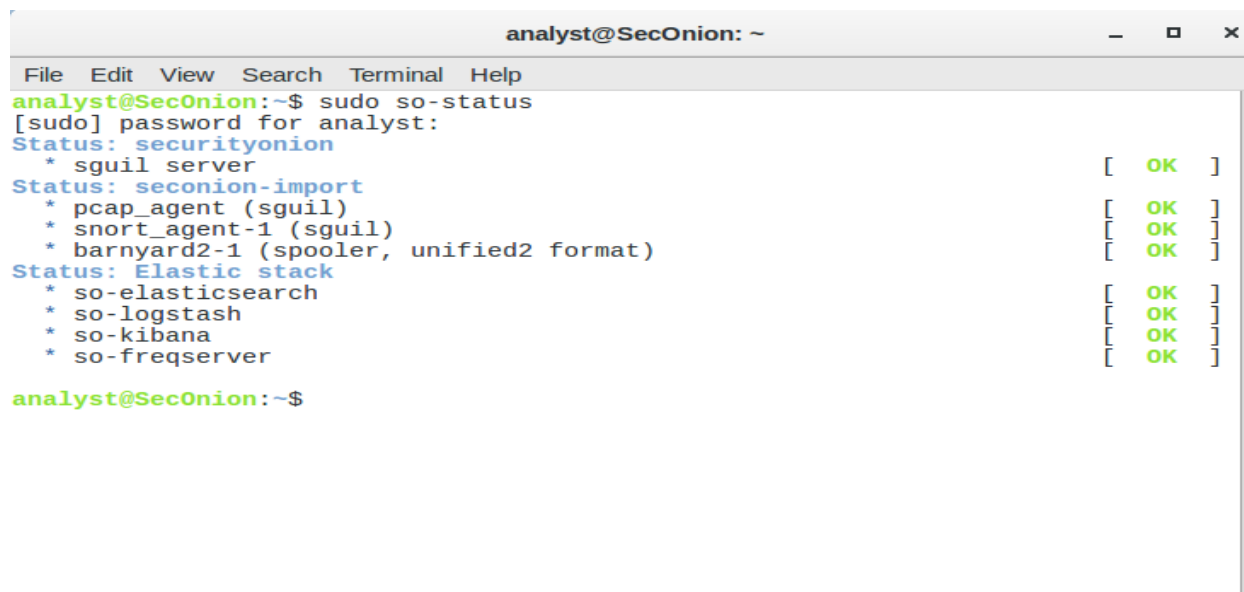
Questo laboratorio si è concentrato sull'analisi dei log relativi a exploit di vulnerabilità documentate nei protocolli HTTP e DNS . L'obiettivo principale era utilizzare Kibana, all'interno della macchina virtuale Security Onion, per indagare su un attacco di tipo SQL injection e su un caso di esfiltrazione di dati tramite query DNS . Il contesto simulato prevedeva che il personale di sicurezza informatica avesse rilevato un exploit con potenziale esposizione di dati sensibili .

Parte 1: Indagare su un Attacco SQL Injection

In questa prima parte, l'obiettivo era identificare la fonte e le informazioni compromesse durante un accesso non autorizzato a un server web tramite SQL injection . L'SQL injection è una tecnica di attacco comune in cui un malintenzionato inserisce comandi SQL dannosi tramite input non sanificati di un'applicazione web per manipolare il database sottostante.

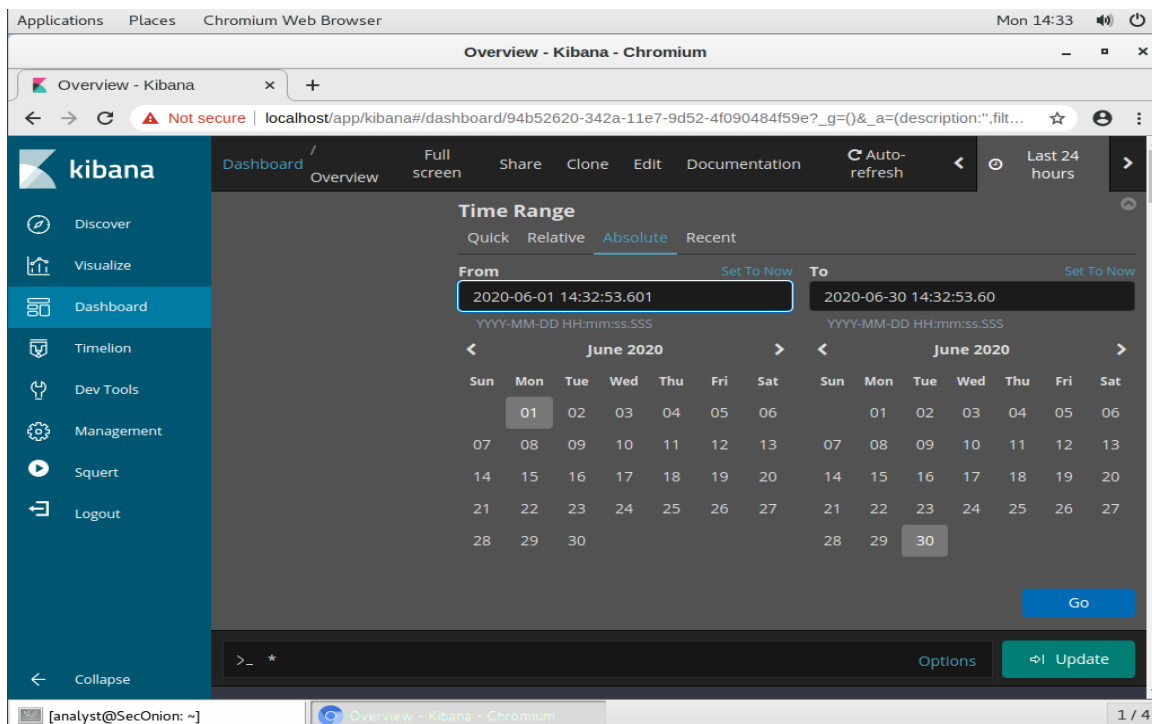
1.1 Configurazione Iniziale e Accesso a Kibana

Dopo aver avviato la VM Security Onion e aver effettuato l'accesso come utente analyst (password cyberops) , abbiamo verificato lo stato dei servizi con il comando `sudo so-status` per assicurarci che tutto fosse operativo.

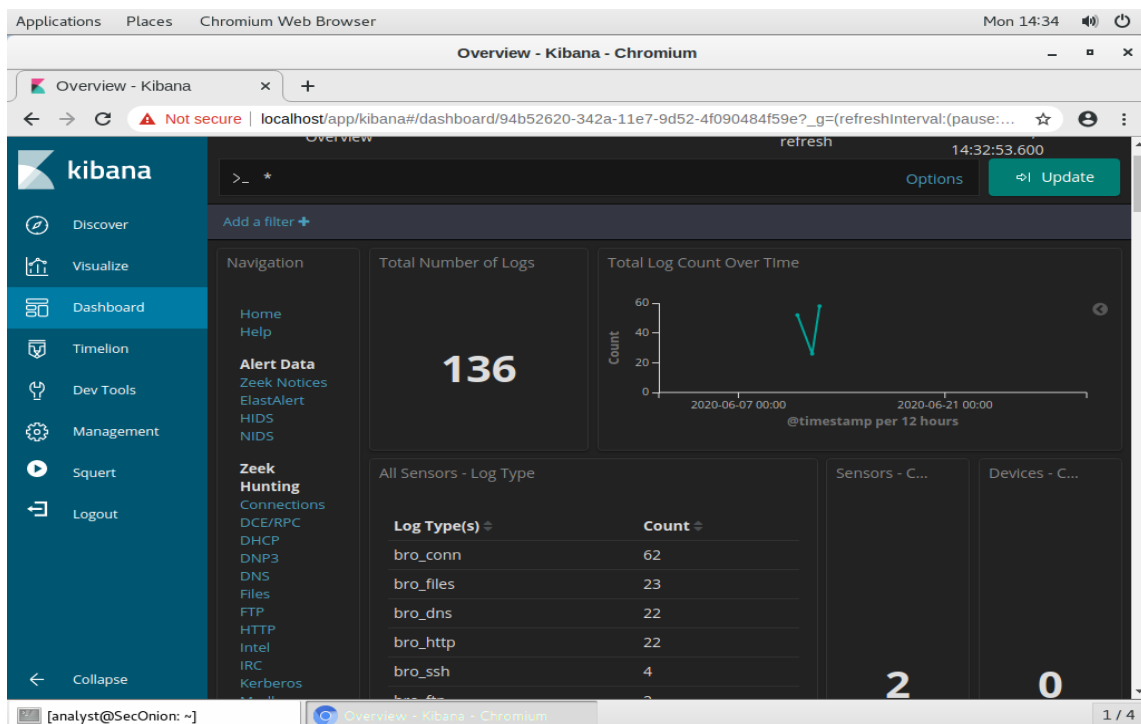


```
analyst@SecOnion: ~  
File Edit View Search Terminal Help  
analyst@SecOnion:~$ sudo so-status  
[sudo] password for analyst:  
Status: securityonion  
* sgul server [ OK ]  
Status: seconion-import  
* pcap_agent (sgul) [ OK ]  
* snort_agent-1 (sgul) [ OK ]  
* barnyard2-1 (spooler, unified2 format) [ OK ]  
Status: Elastic stack  
* so-elasticsearch [ OK ]  
* so-logstash [ OK ]  
* so-kibana [ OK ]  
* so-freqserver [ OK ]  
analyst@SecOnion:~$
```

Successivamente, abbiamo aperto Kibana dal desktop e abbiamo effettuato l'accesso . Poiché l'attacco era avvenuto nel giugno 2020, abbiamo modificato l'intervallo di tempo predefinito ("Ultime 24 ore") . Abbiamo selezionato un intervallo "Assoluto", impostando le date dal 1 giugno 2020 al 30 giugno 2020, e cliccato su "Vai" .

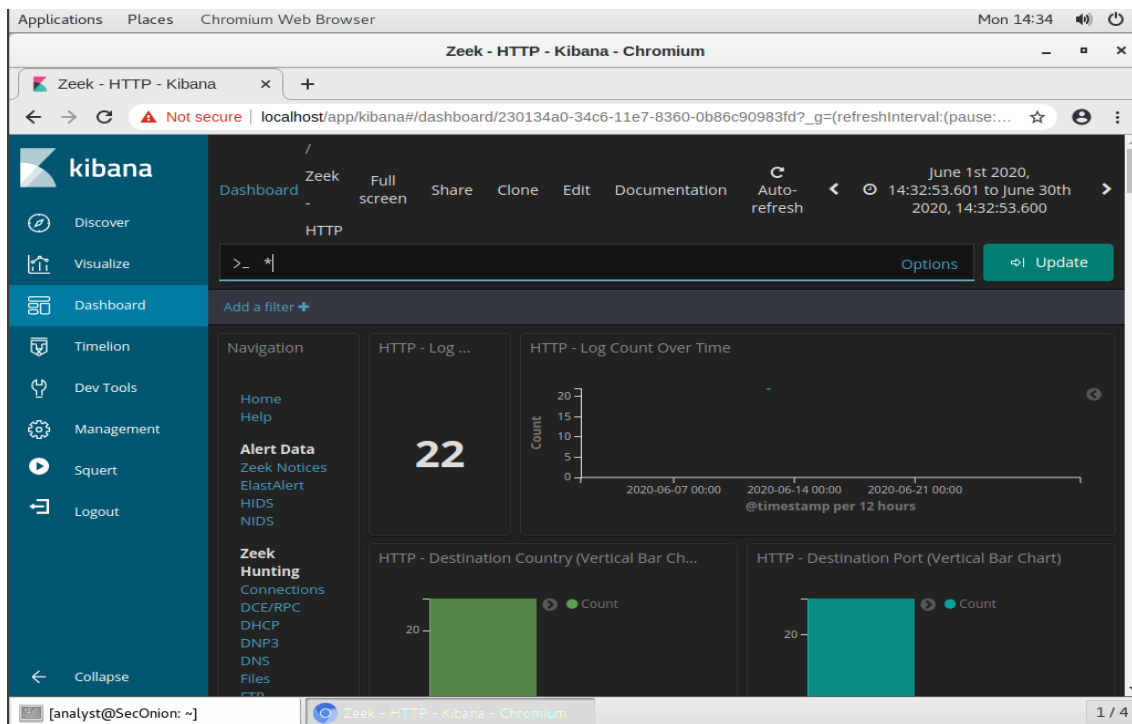


Questo ha aggiornato la dashboard, mostrando il numero totale di log per il periodo selezionato.



1.2 Filtraggio e Analisi del Traffico HTTP

Per concentrarci sull'attacco al server web, abbiamo filtrato i log per visualizzare solo il traffico HTTP. Abbiamo cliccato su "HTTP" nella sezione "Zeek Hunting" della dashboard.

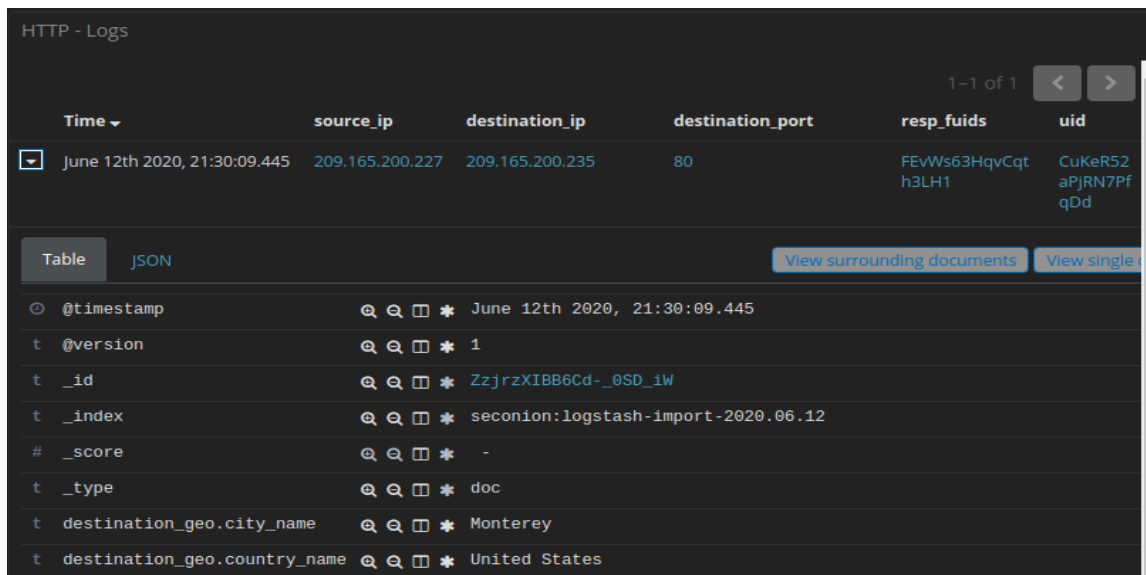


Scorrendo verso il basso fino alla tabella "HTTP - Logs", abbiamo identificato le informazioni chiave della connessione :

HTTP - Logs						
1-1 of 1						
Time	source_ip	destination_ip	destination_port	resp_fuids	uid	
June 12th 2020, 21:30:09.445	209.165.200.227	209.165.200.235	80	FEVWs63HqvCqt h3LH1	CuKeR52 aPJRN7Pf qDd	
Table JSON View surrounding documents View single document						
@timestamp	June 12th 2020, 21:30:09.445					
@version	1					
_id	ZzjrZXIBB6Cd-_0SD_iW					
_index	seconion:logstash-import-2020.06.12					
_score	-					
_type	doc					
destination_geo.city_name	Monterey					
destination_geo.country_name	United States					

- **Indirizzo IP di origine:** 209.165.200.227 [source: 27]
- **Indirizzo IP di destinazione:** 209.165.200.235 [source: 28]
- **Porta di destinazione:** 80 [source: 29]

Abbiamo quindi espanso la prima voce di log cliccando sulla freccia laterale :



The screenshot shows the Kibana interface for HTTP logs. At the top, a table lists log entries. The first entry is selected, and its details are shown in a JSON view below. The JSON view includes fields like @timestamp, @version, _id, _index, _score, _type, and destination_geo.city_name.

Time	source_ip	destination_ip	destination_port	resp_fuids	uid
June 12th 2020, 21:30:09.445	209.165.200.227	209.165.200.235	80	FEvWs63HqvCqt h3LH1	CuKeR52 aPJRN7Pf qDd

Field	Value
@timestamp	June 12th 2020, 21:30:09.445
@version	1
_id	ZzjrZXIBB6Cd-_0SD_1W
_index	seconion:logstash-import-2020.06.12
_score	-
_type	doc
destination_geo.city_name	Monterey
destination_geo.country_name	United States

Il timestamp dell'evento era **12 giugno 2020, 21:30:09.445** . Il tipo di evento era `bro_http` (si noti che Kibana usa ancora il vecchio nome "Bro" per Zeek) . Il campo `message` conteneva dettagli cruciali sulla richiesta HTTP GET .

In particolare, l'URI conteneva la seguente stringa:

```
username='+union+select+ccid, ccnumber, ccv, expiration, null+from+credit_cards+--+&password=
```

Questa stringa è un chiaro indicatore di un tentativo di SQL injection .

L'uso delle parole chiave SQL `union` e `select` suggerisce un tentativo di estrarre dati specifici (`ccid`, `ccnumber`, `ccv`, `expiration`) dalla tabella `credit_cards` del database, bypassando probabilmente i normali meccanismi di autenticazione

1.3 Esame della Trascrizione con capME!

Per approfondire, abbiamo cliccato sul valore nel campo `_id` della voce di log.

destination_ip	209.165.200.235
destination_ips	209.165.200.235
destination_port	80
event_type	bro_http
host	d68c9360b6ae
ips	209.165.200.235, 209.165.200.227
message	<pre>{ "ts": "2020-06-12T21:30:09.445030Z", "uid": "CuKeR52aPjRN7PfQDd", "id_h": "209.165.200.227", "id.orig_p": 56194, "id.resp_h": "209.165.200.235", "esp_p": 80, "trans_depth": 1, "method": "GET", "host": "209.165.200.235", "mutillidae/index.php?page=user-info.php&username='+union+select+ccber,ccv,expiration,null+from+credit_cards+-+&password=&user-info-it-button=View+Account+Details", "referrer": "http://209.165.200.235", "version": "1.1", "user_agent": "Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0", "request_len": 0, "response_body_len": 23665, "status_code": 200, "status_msg": "OK", "s": ["HTTP::URI_SQLI"], "resp_fuids": ["FEVWs63HqvCqth3LH1"], "resp_mime": ["text/html"] }</pre>
method	GET
path	/nsm/import/bro/bro-W5LdfbF0/http.log

Questo ha aperto una nuova scheda del browser con l'interfaccia capME!, che visualizza una trascrizione della sessione di rete (PCAP) . Il testo blu rappresenta le richieste del client (origine) e il testo rosso le risposte del server (destinazione) . All'interno della trascrizione, abbiamo localizzato nuovamente la stringa di SQL injection.

Utilizzando la funzione di ricerca del browser (Ctrl+F) per la parola "username", abbiamo inizialmente trovato occorrenze normali relative all'interfaccia web. Tuttavia, scorrendo più in basso nella trascrizione , abbiamo notato una sezione anomala nella risposta del server (testo rosso):

kibana

Discover

Visualize

Dashboard

Timeline

Dev Tools

Management

Squert

Logout

Collapse

HTTP - Logs

Limited to 10 results. Refine your search. 1-10 of 22

Time

source_ip

destination_ip

destination_port

resp_fuids

uid

_id

June 12th 2020, 21:30:09.445

209.165.200.227

209.165.200.235

80

FEVWs63HqvCqth3LH1

CuKeR52aPjRN7PfQDd

ZzjrZxIBB6Cd-_0SD_iW

Table

JSON

View surrounding documents

View single document

@timestamp

@version

_id

_index

_score

_type

destination_geo.city_name

destination_geo.country_name

destination_geo.ip

destination_geo.location

destination_geo.region_code

June 12th 2020, 21:30:09.445

1

ZzjrZxIBB6Cd-_0SD_iW

seconion-logstash-import-2020.06.12

-

doc

Monterey

United States

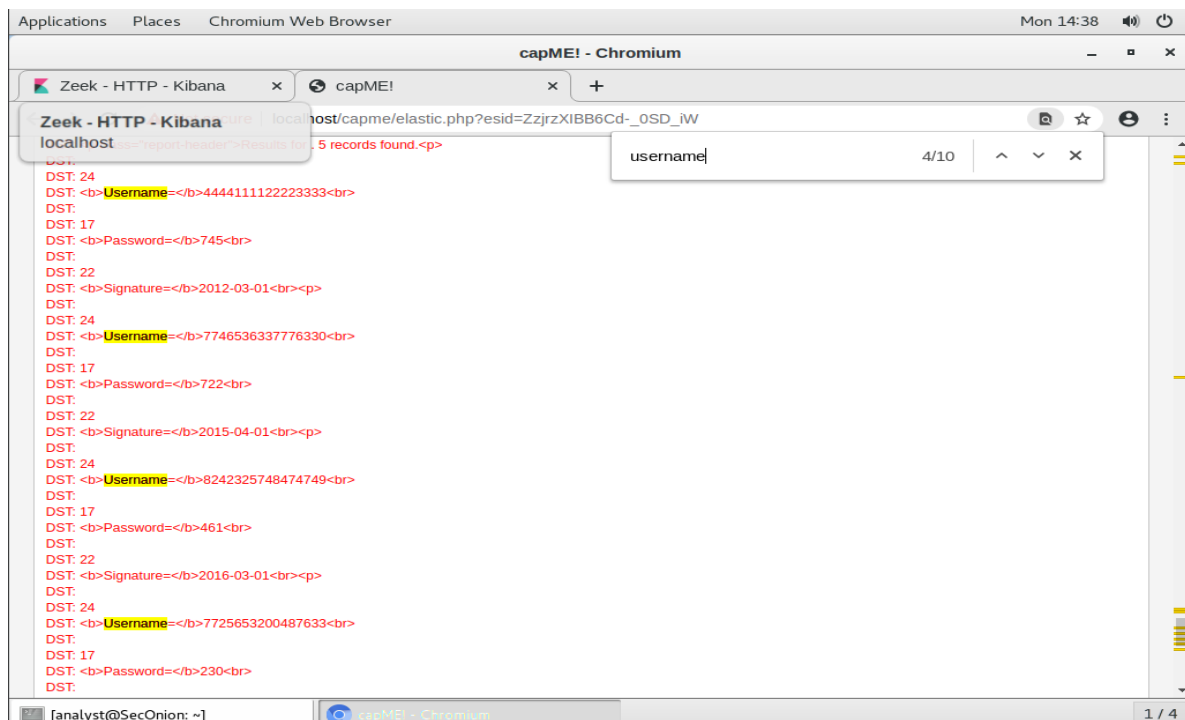
209.165.200.235

{ "lon": -121.8406, "lat": 36.3699 }

US-CA

Invece di una normale risposta della pagina, il server aveva restituito un elenco di dati che corrispondevano ai campi richiesti nell'iniezione SQL: numeri di carta di credito (etichettati come "Username"), codici CVV (etichettati come "Password") e date di scadenza (etichettate come "Signature") .

Questo confermava che l'attacco SQL injection aveva avuto successo e aveva portato all'esfiltrazione di dati sensibili delle carte di credito .



Ecco alcuni esempi dei dati esfiltrati visibili nello screenshot:

- Username: 4444111122223333, Password: 745, Signature: 2012-03-01
- Username: 7746536337776330, Password: 722, Signature: 2015-04-01
- Username: 8242325748474749, Password: 461, Signature: 2016-03-01

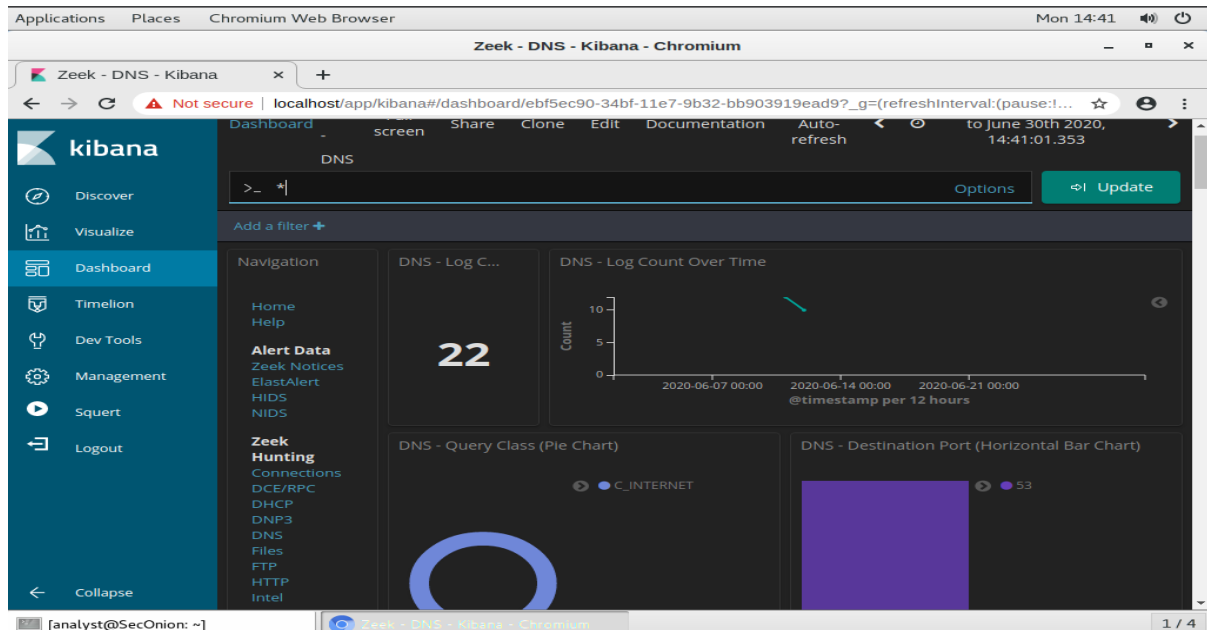
Username	Password	Signature
4444111122223333	745	2012-03-01
7746536337776330	722	2015-04-01
8242325748474749	461	2016-03-01
7725653200487633	230	2017-06-01
1234567812345678627	627	2018-11-01

Parte 2: Analizzare l'Esfiltrazione dei Dati DNS

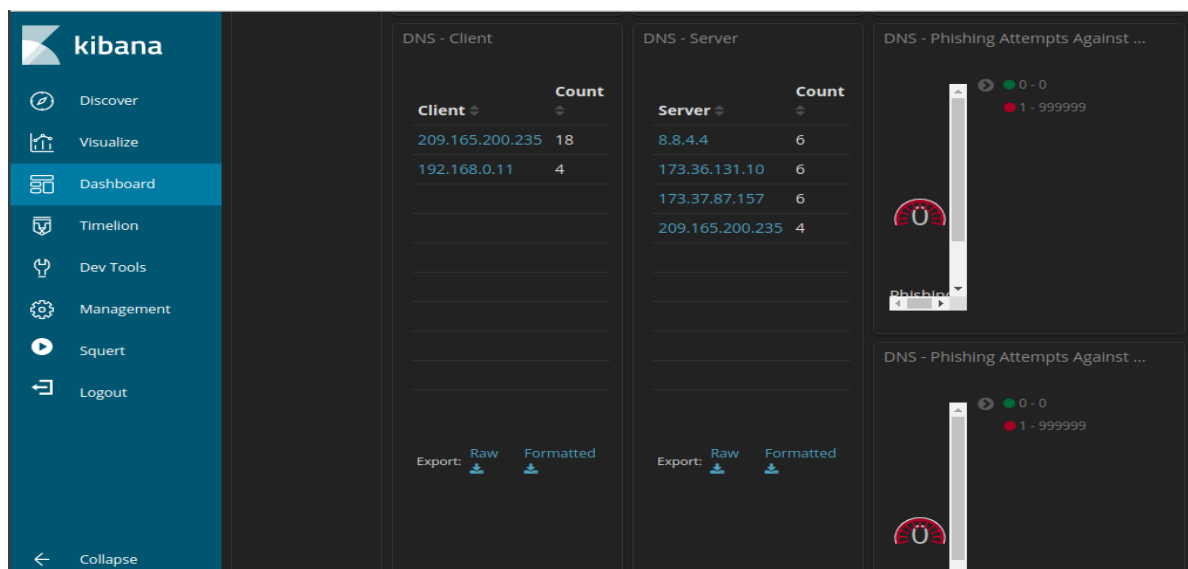
La seconda parte del laboratorio si è concentrata sull'indagine di query DNS sospettosamente lunghe, un potenziale indicatore di esfiltrazione di dati tramite il protocollo DNS.

2.1 Filtraggio e Analisi Iniziale del Traffico DNS

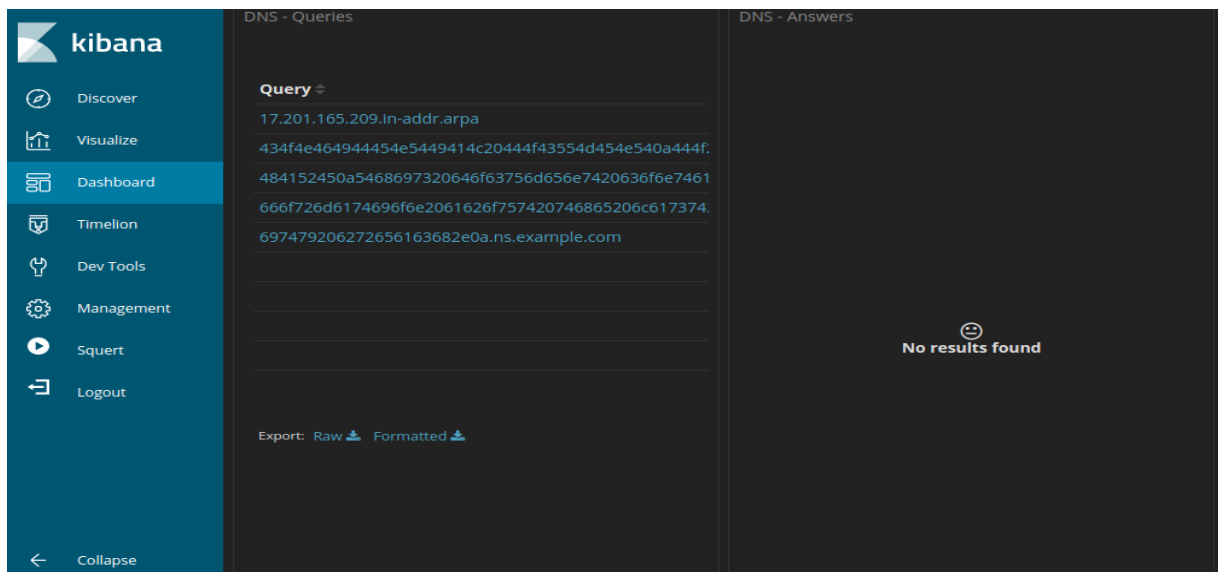
Siamo tornati alla dashboard principale di Kibana, abbiamo rimosso i filtri precedenti e ci siamo assicurati che l'intervallo di tempo fosse ancora impostato su giugno 2020 . Abbiamo quindi cliccato su "DNS" nella sezione "Zeek Hunting".



La dashboard DNS mostrava varie metriche, tra cui il conteggio dei log, i tipi di query più comuni (A, AAAA, PTR, ecc.), i codici di risposta, e i principali client e server DNS.

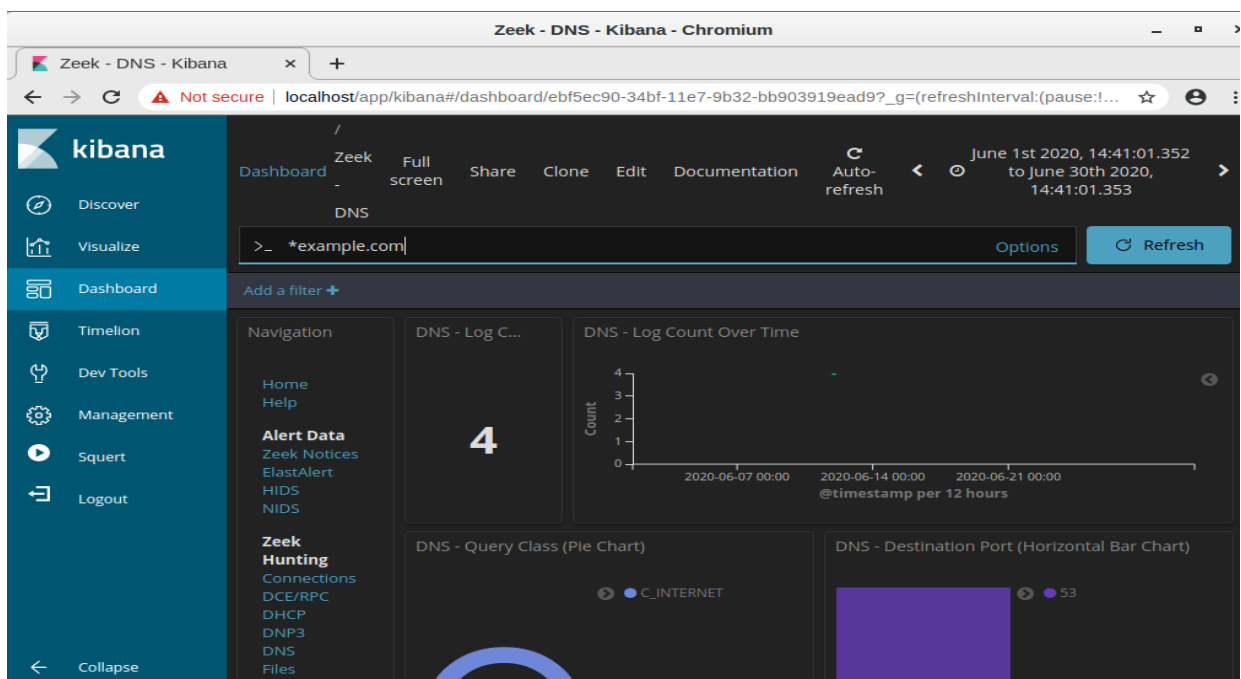


Scorrendo verso il basso fino alla sezione "DNS - Queries", abbiamo notato delle query molto insolite dirette a sottodomini di ns.example.com. Questi sottodomini erano stringhe estremamente lunghe di caratteri apparentemente esadecimali (0-9, a-f), non nomi di dominio leggibili.

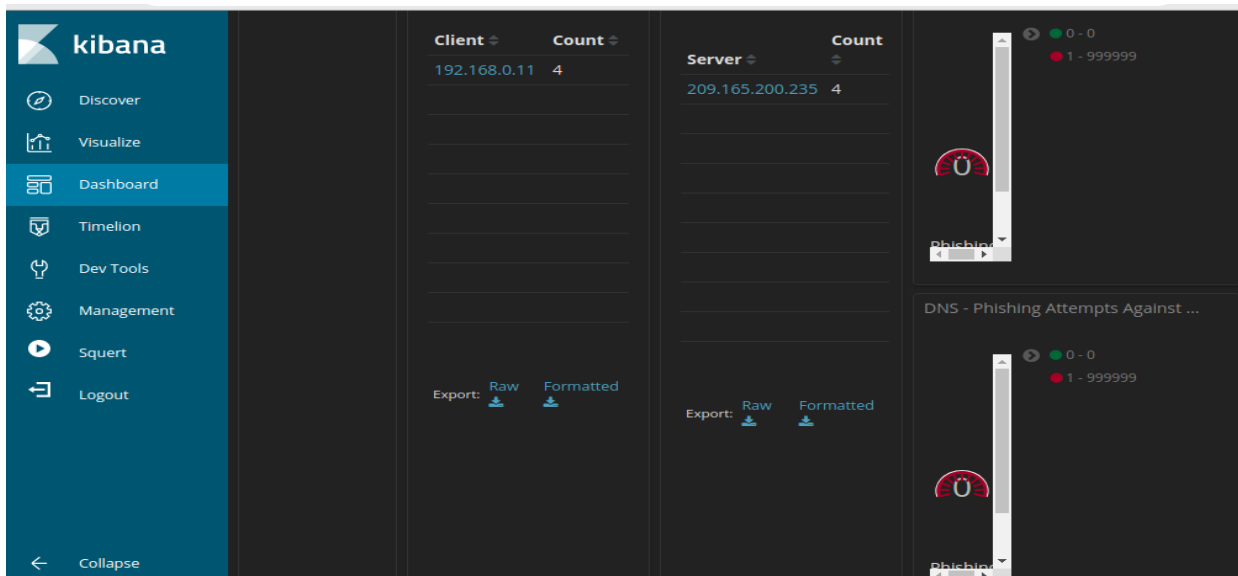


2.2 Isolamento e Analisi delle Query Sospette

Per isolare queste query, siamo tornati in cima alla pagina e abbiamo inserito *example.com nella barra di ricerca, quindi cliccato su "Aggiorna". Questo ha filtrato la vista, riducendo il numero di log visualizzati a quelli relativi a example.com.



Nella vista filtrata, abbiamo identificato il client e il server coinvolti in queste specifiche query DNS :



Client DNS: 192.168.0.11

- **Server DNS:** 209.165.200.235

2.3 Decodifica dei Dati Esfiltrati

Scorrendo nuovamente alla sezione "DNS - Queries", abbiamo visto le quattro query univoche con i lunghi sottodomini esadecimali.

Per capire cosa contenessero queste stringhe, abbiamo esportato l'elenco delle query cliccando su "Export: Raw" .

Questo ha scaricato un file CSV (DNS Queries.csv) nella cartella /home/analyst/Downloads .

Abbiamo aperto una finestra di terminale e navigato in /home/analyst/Downloads.

Abbiamo visualizzato il contenuto del file CSV :

```
analyst@SecOnion:~$ cd /home/analyst/Downloads/
analyst@SecOnion:~/Downloads$ ls
DNS - Queries.csv
analyst@SecOnion:~/Downloads$ cat DNS\ -\ Queries.csv
Query, Count
"434f4e464944454e5449414c20444f43554d454e540a444f204e4f542053.ns.example.com", 1
"484152450a5468697320646f63756d656e7420636f6e7461696e7320696e.ns.example.com", 1
"666f726d6174696f6e2061626f757420746865206c617374207365637572.ns.example.com", 1
"697479206272656163682e0a.ns.example.com".1
```

Usando un editor di testo (come gedit) , abbiamo modificato il file CSV rimuovendo tutto tranne le stringhe esadecimali (incluse le virgolette e la parte .ns.example.com), salvando poi il file pulito.

Infine, abbiamo utilizzato il comando xxd -r -p per decodificare le stringhe esadecimali dal file pulito e reindirizzare l'output nel file secret.txt. Abbiamo poi visualizzato il contenuto di secret.txt con cat :

```
analyst@SecOnion:~/Downloads$ xxd -r -p "DNS - Queries.csv" > secret.txt
analyst@SecOnion:~/Downloads$ cat secret.txt
CONFIDENTIAL DOCUMENT
DO NOT SHARE
This document contains information about the last security breach.
```

La decodifica ha rivelato che le lunghe stringhe esadecimali non erano affatto sottodomini, ma contenevano un messaggio di testo: "CONFIDENTIAL DOCUMENT DO NOT SHARE This document contains information about the last security breach.".

Questo risultato ha dimostrato che il protocollo DNS veniva utilizzato come canale nascosto per esfiltrare dati. Probabilmente un malware sull'host client (192.168.0.11) stava leggendo documenti sensibili, codificandone il contenuto in esadecimale e suddividendolo in blocchi inviati come query DNS per sottodomini inesistenti. Questo metodo è insidioso perché il traffico DNS è spesso meno monitorato rispetto ad altri protocolli come HTTP/S e tende ad essere permesso attraverso i firewall, rendendolo un vettore attraente per l'esfiltrazione di dati aggirando le difese perimetrali.

Conclusione

Attraverso l'analisi dei log HTTP e DNS in Kibana, questo laboratorio ha dimostrato come identificare e analizzare due distinti vettori di attacco. Abbiamo confermato un attacco SQL injection riuscito che ha portato all'esfiltrazione di dati di carte di credito tramite HTTP e abbiamo scoperto l'uso del DNS tunneling per esfiltrare un documento confidenziale codificato all'interno di query DNS. Entrambi gli scenari evidenziano l'importanza del monitoraggio approfondito dei log di rete e dell'analisi del traffico, anche per protocolli apparentemente benigni come il DNS, per rilevare attività malevole.