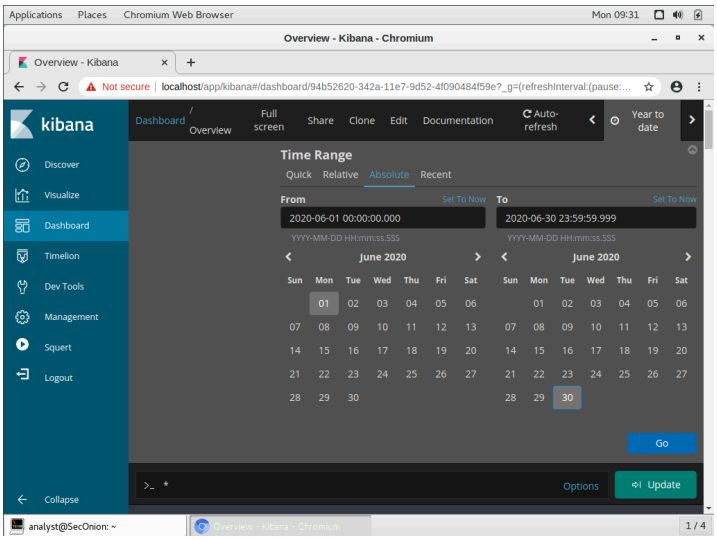
Bonus 1: Interpretare Dati HTTP e DNS per Isolare l'Attore della Minaccia

Introduzione

Kibana è uno strumento open-source utilizzato in ambito cybersecurity per la visualizzazione e l'analisi dei dati raccolti. Fa parte dello stack ELK (Elasticsearch, Logstash, Kibana) e consente di esplorare grandi volumi di log in tempo reale.



Gli analisti di sicurezza lo impiegano per monitorare eventi, individuare comportamenti anomali e creare dashboard interattive. Kibana supporta la creazione di grafici, tabelle e mappe utili per correlare eventi sospetti.
Grazie alla sua interfaccia intuitiva, è possibile filtrare i dati con query avanzate e segnare incidenti rilevanti.

In contesti di Security Information and Event Management (SIEM), Kibana è usato per rilevare minacce. Elastic Security, integrato in Kibana, fornisce funzionalità specifiche per la difesa degli endpoint. La sua scalabilità lo rende adatto a piccole reti e infrastrutture complesse. In sintesi, Kibana è uno strumento essenziale per il monitoraggio proattivo della sicurezza informatica.

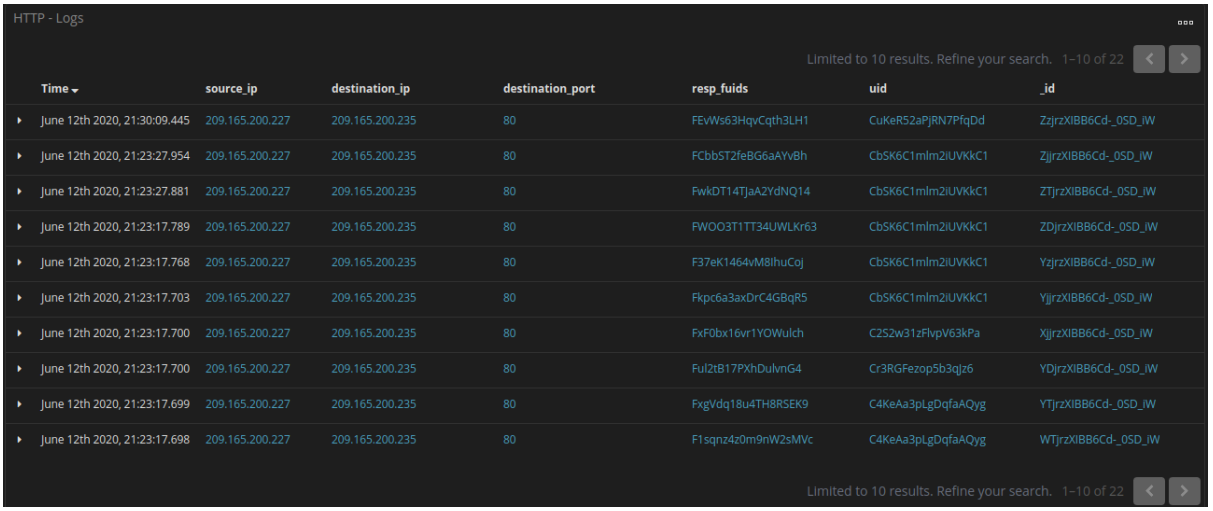
Parte 1: Investigare un Attacco di SQL Injection

PASSO 1: CAMBIARE L'INTERVALLO DEL TEMPO

Eseguendo vari passaggi guidati dell'esercizio (a,b,c,d,e) arriviamo al passo 2, e alla risposta delle domande.

PASSO 2: FILTRARE IL TRAFFICO HTTP

Analizzando i log troviamo i 2 indirizzi ip cercati e la porta di destinazione.


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
June 12th 2020, 21:23:27.954	209.165.200.227	209.165.200.235	80	FCbbST2feBG6aAYvBh	CbSK6C1mIm2iUVKkC1	ZjjrzXIBB6Cd-_0SD_iW
June 12th 2020, 21:23:27.881	209.165.200.227	209.165.200.235	80	FwkDT14TJaA2YdNQ14	CbSK6C1mIm2iUVKkC1	ZTjrzXIBB6Cd-_0SD_iW
June 12th 2020, 21:23:17.789	209.165.200.227	209.165.200.235	80	FWOO3T1TT34UWLKr63	CbSK6C1mIm2iUVKkC1	ZDjrzXIBB6Cd-_0SD_iW
June 12th 2020, 21:23:17.768	209.165.200.227	209.165.200.235	80	F37eK1464vM8IhuCoj	CbSK6C1mIm2iUVKkC1	YzjrzXIBB6Cd-_0SD_iW
June 12th 2020, 21:23:17.703	209.165.200.227	209.165.200.235	80	Fkpc6a3axDrC4GBqR5	CbSK6C1mIm2iUVKkC1	YjjrzXIBB6Cd-_0SD_iW
June 12th 2020, 21:23:17.700	209.165.200.227	209.165.200.235	80	FxF0bx16vr1YOWuIch	C2S2w31zFlvpV63kPa	XjjrzXIBB6Cd-_0SD_iW
June 12th 2020, 21:23:17.700	209.165.200.227	209.165.200.235	80	Ful2tB17PXhDulvnG4	Cr3RGFezop5b3qJz6	YDjrzXIBB6Cd-_0SD_iW
June 12th 2020, 21:23:17.699	209.165.200.227	209.165.200.235	80	FxgVdq18u4TH8RSEK9	C4KeAa3pLgDqfaAQyg	YTjrzXIBB6Cd-_0SD_iW
June 12th 2020, 21:23:17.698	209.165.200.227	209.165.200.235	80	F1sqnz4z0m9nW2sMVc	C4KeAa3pLgDqfaAQyg	WTjrzXIBB6Cd-_0SD_iW

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

Domanda 1: **Qual è l'indirizzo IP sorgente?** **R:** **Sorgente ip:** 209.165.200.227

Domanda 2: **Qual è l'indirizzo IP destinazione?** **R:** **Destinazione ip:** 209.165.200.235

Domanda 3: **Qual è il numero di porta destinazione?** **R:** **Porta di destinazione:** 80

Domanda 4: **Qual è il timestamp del primo risultato?** **R:** June 12th 2020, 21:30:09.445

Domanda 5: **Qual è il tipo di evento? Cosa è incluso nel campo message?**

Tipo di evento:
R: L'evento rappresenta **una richiesta HTTP GET contenente un attacco SQL Injection**. È classificato come potenzialmente dannoso, come indicato anche dal tag `"HTTP::URI_SQLI"`.

Domanda 6: **Questi sono dettagli sulla richiesta HTTP GET fatta dal client al server. Concentrati specialmente sul campo uri nel testo del messaggio. Qual è il significato di queste informazioni?** Contenuto del campo message:

R: Il campo `message` (in questo contesto rappresentato dalla riga intera del log in formato JSON) contiene dettagli su **una richiesta HTTP GET** inviata dal client `209.165.200.227` al server `209.165.200.235` sulla porta 80.

PASSO 3: RIVEDERE IL RISULTATO

Focus sul campo uri:

plaintext

```
"/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"
```

Questo campo mostra un attacco **SQL Injection** nel parametro `username`.
L'attaccante tenta di iniettare la seguente query SQL:

sql

```
' UNION SELECT ccid, ccnumber, ccv, expiration, null FROM credit_cards --
```

Obiettivo: **esfiltrare dati dalla tabella `credit_cards`** (ID carta, numero, codice di sicurezza e data di scadenza), sfruttando una vulnerabilità in `user-info.php`.

Significato delle informazioni:

- **È un tentativo di attacco SQL Injection via GET**, usato per estrarre dati sensibili dal database.
- La presenza del tag `HTTP::URI_SQLI` indica che è stato rilevato automaticamente come potenziale attacco.
- Può essere utilizzato dai difensori per:
    - Bloccare IP sospetti.
    - Monitorare attività malevole.
    - Correggere la vulnerabilità nel codice del sito (es. assenza di sanitizzazione dei parametri).

In sintesi, si tratta di un **alert di sicurezza** rilevante, utile per l'analisi delle minacce e la risposta agli incidenti.

Domanda: **Cosa vedi più avanti nella trascrizione riguardo ai nomi utente? Fornisci alcuni esempi di nome utente, password e firma che sono stati esfiltrati.**

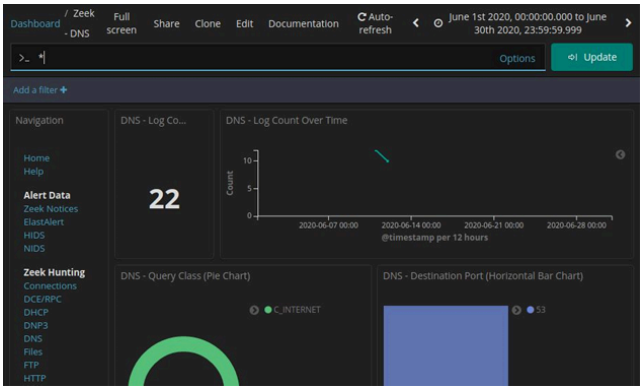
R: Troviamo questo, un tentativo di password generator che trova anonymous come username. Nei commenti si trova una riga che dice che la password potrebbe essere vuota oppure samurai. Inoltre un'altro esempio di nome utente è anonymous.

```
DST: 130
DST: php?page=dns-lookup.php">DNS Lookup</a></li>
DST: .....</ul>
DST: .....</li>
DST: .....<li>
DST: .....<a href="">JavaScript Injection</a>
DST: .....<ul>
DST: .....<li><a href="./index.php">Those "Back" Buttons</a></li>
DST: .....<li>
DST: .....<a href="./index.php?page=password-generator.php&username=
DST:
DST: 9
DST: anonymous
DST:
DST: 1a4
DST: ">
DST: .....Password Generator
DST: .....</a>
DST: .....</li>
DST: .....</ul>
DST: .....</li>
DST: .....<li>
DST: .....<a href="">HTTP Parameter Pollution</a>
DST: .....<ul>
DST: .....<li><a href="./index.php?page=user-poll.php">Poll Question</a></li>
DST: .....</ul>
DST: .....</li>
DST: .....<li>
DST: .....<a href="">Cascading Style Injection</a>
DST: .....<ul>
DST: .....<li><a href="./index.php?page=set-backgr
DST: abel" style="text-align: center;">
DST: ...Site hacked...err...quality-tested with Samurai WTF, Backtrack, Firefox, Burp-Suite, Netcat, and
DST: ...<a href="https://addons.mozilla.org/en-US/firefox/collections/jdruin/pro-web-developer-qa-pack/" style="text-decoration: none;">
DST: ...these Mozilla Add-ons
DST: ...</a>
```

Da una ricerca con parola chiave Samurai si ritrova questo messaggio. Sono anche menzionati Burp-Suite, Netcat, Backtrack.

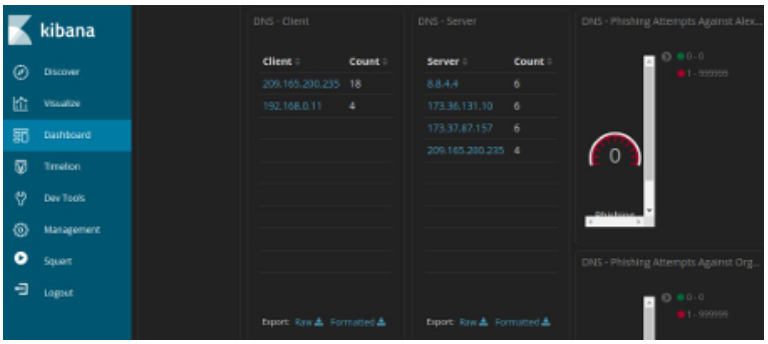
Parte 2: Analizzare l'Esfiltrazione DNS

PASSO 1: FILTRARE PER TRAFFICO DNS



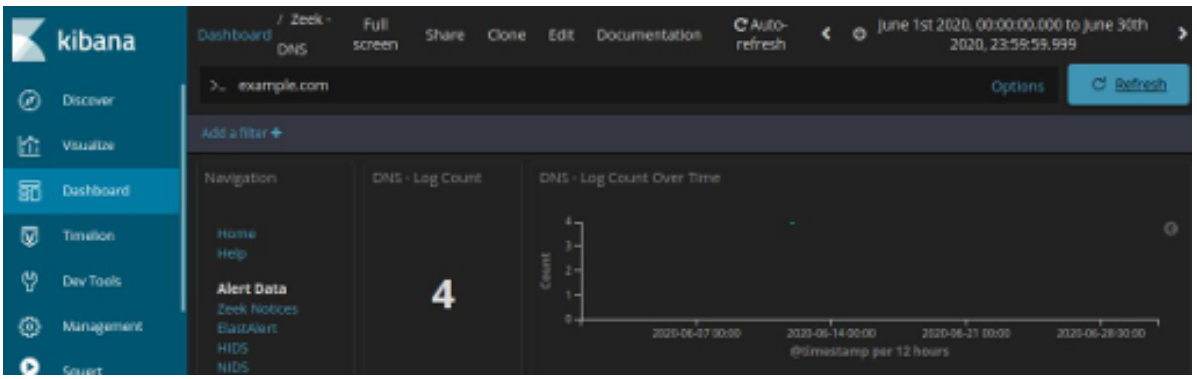
Dalla parte superiore del Dashboard Kibana, cancelliamo eventuali filtri e termini di ricerca e facciamo clic su Home sotto la sezione Navigation del Dashboard. Clicchiamo poi su **DNS** notando metriche e grafico.

PASSO 2: RIVEDERE LE VOCI RELATIVE AI DNS.



Scorriamo verso il basso vedendo i principali tipi di query DNS. Trovando anche un elenco dei principali client DNS e server DNS basati sul conteggio delle loro richieste e risposte.

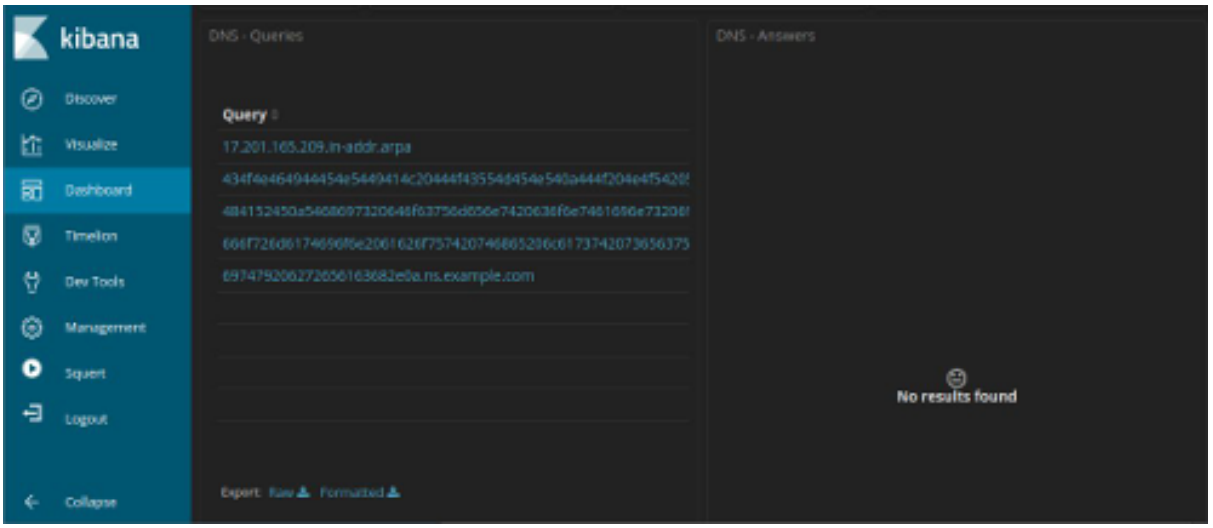
Una volta eseguiti tutti i passaggi descritti dall'esercizio ci troviamo in questa schermata di Kibana, ove in alcune query abbiamo sottodomini insolitamente lunghi collegati a ns.example.com . Usiamo come filtro e clickiamo su Update.



Abbiamo individuato le informazioni su DNS - Client DNS - Server. **Registriamo gli Ip del client e del server DNS.**

PASSO 3: DETERMINARE I DATI ESFILTRATI

Continuando a scorrere verso il basso possiamo vedere quattro voci di log uniche per le query DNS a example.com. Notiamo anche come le query siano a sottodomini sospettosamente lunghi collegati a ns.example.com.
Facciamo click su Export: Raw per scaricare le query su un file esterno.



Andiamo su → /home/analyst/Downloads ed apriamo il file usando un editor di testo come gedit. Modifichiamo il file rimuovendo anche le virgolette e dovremmo avere qualcosa del genere:

```
434f4e464944454e5449414c20444f43554d454e540a444f204e4f542053
484152450a5468697320646f63756d656e7420636f6e7461696e7320696e
666f726d6174696f6e2061626f757420746865206c6173742073656375726
69747920627265616368682e0a
```

Apriamo adesso un terminale usando il comando ***xxd*** e decodifichiamo il testo nel file CSV e salviamolo in un file chiamato ''Secret.txt''. Con ***Cat*** andiamo a visualizzare il suo contenuto.

```
analyst@SecOnion:~/Downloads$ xxd -r -p "DNS - Queries.csv" > secret.txt
analyst@SecOnion:~/Downloads$ cat secret.txt
```

Domanda 1: **I sottodomini erano realmente sottodomini? Se no, qual è il testo?**

R: No, non erano veri sottodomini. Erano dati codificati in esadecimale mascherati come sottodomini DNS per eludere i controlli di sicurezza.

Domanda 2: **Cosa implica questo risultato?**

R: Questo indica un attacco di esfiltrazione dati via DNS dove:

- I dati sensibili sono stati codificati in formato esadecimale
- Ogni "sottodominio" conteneva una porzione dei dati codificati
- Le query DNS sono state usate come canale nascosto per trasferire informazioni

Domanda 2.1: **Quale è il significato più ampio?**
R: Questo rappresenta una tecnica di evasion avanzata perché:

- Il traffico DNS è raramente ispezionato in dettaglio
- Le query DNS sono considerate "normali" e passano attraverso la maggior parte dei firewall
- È difficile da rilevare senza analisi forensi specifiche
- Aggirare i controlli DLP (Data Loss Prevention) tradizionali

Domanda 3: **Cosa potrebbe aver creato queste query?**
R: Possibili responsabili:

- Malware avanzato (APT - Advanced Persistent Threat)
- Insider threat con strumenti personalizzati
- Attaccanti sofisticati che usano tecniche di DNS tunneling
- Script personalizzati per l'esfiltrazione steganografica

Domanda 4: **Perché DNS come mezzo di esfiltrazione?**
R: Il DNS è stato scelto perché:

Vantaggi tattici:

- **Ubiquità:** DNS è presente ovunque e necessario per il funzionamento di rete
- **Bassa sospettosità:** Le query DNS sono considerate traffico legittimo
- **Bypass dei controlli:** Raramente ispezionato dai sistemi DLP
- **Alta disponibilità:** Funziona anche in reti fortemente filtrate

Caratteristiche tecniche:

- **Capacità di payload:** Ogni query può trasportare ~255 caratteri
- **Difficile da bloccare:** Bloccare DNS compromette la rete
- **Steganografia naturale:** I dati si nascondono nel traffico normale
- **Resilienza:** Funziona anche con DNS ricorsivi e cache

Contromisure Consigliate

1. Monitoraggio DNS avanzato
2. Analisi delle query anomale (lunghezza, entropia, pattern)
3. DNS filtering con whitelist/blacklist
4. Behavioral analysis del traffico DNS
5. Implementazione di DNS over HTTPS/TLS con controlli aggiuntivi

Questo è un ottimo esempio di come gli attaccanti sfruttino protocolli "fidati" per scopi malevoli.