Detecting Typosquatting with Apache Metron

## Cybersquatting

In 1990s, a tremendous growth in internet commerce triggered a spike in new web domain registrations. A new phenomenon took place. Individuals known as “cybersquatters” began to register domain names resembling companies and brands names, which did not have online presence. The goal was to sell these domains back to companies and trademark owners at much higher prices.

One of the examples of cybersquatting is related to the TRUMP trademark owned by Donald Trump. In 2007, J. Taikwok Yung registered four web sites parodying Mr. Trump and TV shows with his involvement. The domain names—trumpbudhabi.com, trumpindia.com, trumpmumbai.com, and trumpbeijing.com—were registered when the Trump organization declared intentions to build condominiums and hotels in Bangalore and Mumbai, India. Attorneys representing Trump filed a suit against Yung in 2013 seeking $100,000 as a compensation for damages of federal cybersquatting for each of the domains based on the Anticybersquatting Consumer Protection Act. The court ruled that Mr. Yung must pay $32,000 damages to Mr. Trump.

## Typosquatting

Typosquatting is a use of domain names spelled closely to well-known ones in a malicious ways, such as for phishing or spear phishing attacks. A study conducted by Sophos, anti-malware and encryption products developer, analyzed a number of domain names generated by one-character mistakes for Facebook.com, Google.com, Twitter.com, Microsoft.com, Apple.com and Sophos.com. This resulted in 2,249 unique site names, such as pple.com, facemook.com, twitterz.com, etc. The study revealed that not all of the generated one-character-wrong domain names were registered. Several web sites from the produced list belonged to their rightful owners, e.g. racebook.com was a racing site and goole.com described a large port in England.

The study revealed that approximately 3% of studied URLs were related to hacking, phishing, online fraud or spamming; 2% fell into the adult and dating category, 15% of links contained or were related to advertisement sites and popups, and 12% belonged to cybersquatting or domain parking category.

Another study produced by Sophos, described a case of a person who lost 16.50 of Bitcoin, which was worth $10,000 at then current prices. In a URL, instead of typing *blockchain* the fraud victim entered *block****o****chain*, and consequently a browser was redirected to a phishing website. Only four differences could be visually detected between the original and phishing sites: (1) Favicon was missing from the phishing site, (2) Differences in domain names and protocols in URLs, (3) Differences in menu items, and (4) The phishing site prompted a user to enter a password in the very beginning while the original site did not.

|  |
| --- |
| Original Website |
|  |
| Phishing Website |
|  |

## Problem Statement

Generate summary data for domains from the Alexa Top Sites list in a Bloom Filter and use it to detect potential typosquatting instances in proxy data. Apply Apache Metron to combat typosquatting using its flat file loader and typosquatting generation capabilityes. Generate a list of candidate typosquatted domain names, import it into an HBase table, and verify these candidate domain names with Metron’s ENRICHMENT\_EXISTS function.

### Bloom Filter

Bloom Filter was introduced by Burton H. Bloom in his 1970 paper “Space/time trade-offs in hash coding with allowable errors.” The Bloom is commonly used to efficiently test whether an item belongs to a set with two possible outcomes: possibly in a set or definitely not in a set. Apache Metron allows to create, merge or modify bloom filters with the following functions:

* BLOOM\_INIT(size, fpp) creates a Bloom filter to handle size number of elements with fpp probability of false positives (0 < fpp < 1).
* BLOOM\_ADD(filter, object) adds an item to an existing Bloom filter.
* BLOOM\_MERGE( filters ) merges Bloom filters passed in a list data structure.

## Proxy

When employed, a proxy device intercepts and sends requests from client computers on a network to web servers. Then it receives responses, such as web page data, from websites, and sends them back to the individual clients on the network.

## Squid Proxy

Squid is a caching proxy for the Web supporting HTTP, HTTPS, FTP, and more. It reduces bandwidth and improves response times by caching and reusing frequently-requested web pages. Squid has extensive access controls and makes a great server accelerator. It runs on most available operating systems, including Windows and is licensed under the GNU GPL.

## Generating Typosquatting Domain with Metron

Apache Metron contains a capability for typosquatting domain generation using strategies, such as listed below:

* Bitsquatting—when a malicious domain name is one bit different from the original one. Example:

|  |
| --- |
| Binary representation of cnn.com: |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | 01100011 | 0110111**0** | 01101110 | 0101110 | 01100011 | 01101111 | 01101101 | | C | n | n | . | c | o | m | |
| Domain name with one bit difference from cnn.com: |
| |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | | 01100011 | 0110111**1** | 01101110 | 0101110 | 01100011 | 01101111 | 01101101 | | C | o | n | . | c | o | m | |

* Homoglyphs. Substituting characters for ASCII or Unicode analogues which are visually similar, e.g. latimes.com vs. latlmes.com where character i is substituted with a vertical bar.
* Subdomains. Making part of the domain a subdomain, e.g. am.azon.com where the top level domain name is azon.com and am.azon.com is a subdomain.
* Hyphenation. Inserting or removing hyphens.
* Insertion of characters, e.g. google.com vs. gooogle.com.
* Omission of characters, e.g. google.com vs. gogle.com.
* Replacement, e.g. google.com vs. gopgle.com.
* Transposition, e.g. google.com vs. gogole.com.

In this assignment, we need to generate summary sketches from flat data and load it into HBase.

## Apache HBase

Apache HBase is a NoSQL database, which operates within the Hadoop ecosystem. HBase is open-source non-relational distributed database, which originates from the Google Bigtable process. One of the main features of HBase is that it works in column-oriented fashion, grouping table columns into column families. Each column family is stores as a separate data file. More details about HBase are available at <https://mapr.com/blog/in-depth-look-hbase-architecture/>

## Elasticsearch

Elasticsearch is an open-source distributed search and analytics engine, which uses REST architecture. Elasticsearch is commonly used for is commonly used for log analytics, full-text search, and operational intelligence use cases. Elasticsearch can be connected with Kibana, a visualization tool, to provide near-real time analytics using large volumes of data. More details about Elasticsearch are available at <https://aws.amazon.com/elasticsearch-service/what-is-elasticsearch/>

## Generating Summaries with Metron

Apache Metron has two utilities, flatfile\_summarizer.sh and flatfile\_loader.sh, which will be used for generating summaries. Useful flatfile\_summarizer.sh parameters:

* state\_init for initializing a state object.
* state\_update for updating a state object.
* state\_merge – for merging a list of states.

Both flatfile\_loader.sh and flatfile\_summarizer.sh allow adjusting parallel processing settings by specifying the number of threads (-p option) and batch size (-b option), specify an output destination (-o option) and the output mode (-om option). The list of output modes are:

* LOCAL. Output to local disk (default).
* CONSOLE. Write the object summarized to std out (terminal window). This is useful for visual analyzis of summary statistics about the data being imported, e.g. view the number of typosquatted domains generated for the Alexa Top Sites list.
* HDFS. Write to a folder in the Hadoop Distributed File System.

## Reading Summaries with Metron

Metron’s built-in language, Stellar, allows to read data stored in HDFS using function OBJECT\_GET(hdfs\_path), which will also deserialize the data into an object for later use.

Example. The flatfile\_summarizer.sh utility was used to write an object to /apps/metron/objects/alexa\_10k\_filter.ser file in HDFS. It is possible to read, deserialize this object and use the Bloom filter to determine if the domain goggle is a typosquatted domain with the following line of Stellar language:

BLOOM\_EXISTS( OBJECT\_GET('/apps/metron/objects/alexa\_10k\_filter.ser'), 'goggle')

Assignment

## Step 1. Obtain Putty—an SSH client for Windows.

This lab focuses on Windows users. Linux and Max users may complete the assignment as well.

**Windows users**. Download the Putty SSH client from <https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html> . Download either 32bit or 64 bit version of putty.exe from the Alternative Binary Files section.

**Linux users** may use a command-line SSH implementation. A guide is available at <https://www.digitalocean.com/community/tutorials/how-to-configure-ssh-key-based-authentication-on-a-linux-server>

**Mac users** may use Putty for Mac OS or other SSH client. Guides are available at <https://www.ssh.com/ssh/putty/mac/> and <https://support.rackspace.com/how-to/logging-in-with-an-ssh-private-key-on-linuxmac/>

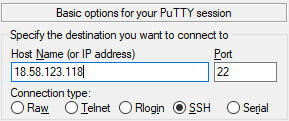
## Step 2. Obtain a private key.

Download a private RSA key from WebCampus. You can find **metron-cluster.ppk** in the Course Documents section. Save the key at a location on your computer. You can create a new folder, for example, C:\tmp\keys\.

Note: Normally, private keys are not shared. This is done to simplify the assignment and reduce your workload.

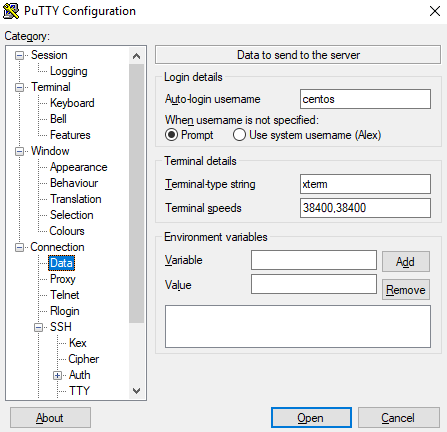
## Step 3. Create a new SSH connection in Putty.

Start Putty. First, enter the IPv4 address of your single-node Hadoop cluster. This IP address was sent in a separate email. Make sure SSH is selected.

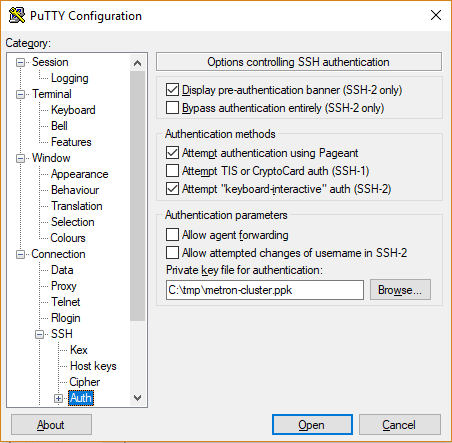


In Putty, use left-side tree-like navigation panel.

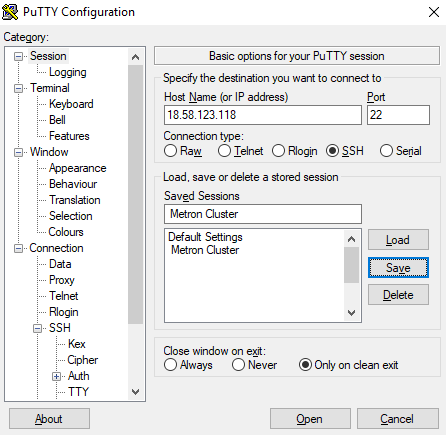
Under Connection, click Data and enter centos in the Auto-login username.



Under Connection, expand SSH and click on the Auth item. Then in the right panel, click the Browse button and locate the previously downloaded RSA key.



Do NOT click the Open button yet. Scroll the left-hand-side navigation bar to the top and click the Session item. Next, in the Session box type Metron Cluster and save the connection by clicking the Save button.

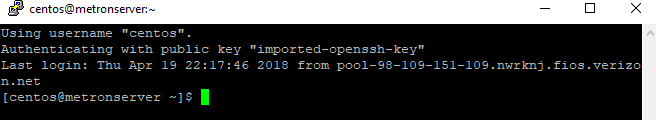


Make a screenshot and add a description.

The connection should be saved successfully.

## Step 4. Establish a connection.

Start Putty. Load the previously stored Metron Cluster connection. For this, click on Metron Connection and click the Load button. Click the Open button. The connection should be established. A similar window should be displayed.



Make a screenshot and add a description.

## Step 5. Set environment variables.

The following environment variables will be used in this assignment: METRON\_HOME—The home directory of Apache Metron, ZOOKEEPER—The list of Zookeeper nodes port numbers specified, BROKERLIST—The comma-separated list of Kafka brokers with port numbers specified and ES\_HOST—An Elasticsearch node and its port number. Since we use a single-node Apache Hadoop cluster, everything is installed on the single node named metronserver.localdomain.

METRON\_HOME variable should point to Metron’s home directory.

ZOOKEEPER variable contains host(s) and port(s) of cluster nodes to establish a connection with Zookeper in a comma-separated format, e.g. node1:2181,node2:2181.  Since we use a single-node cluster, there will be a single host.

BROKERLIST – a variable containing a comma separated list of nodes and ports, where Kafka brokers are running, e.g. node1:6667,node2:6667. Since we use a single-node cluster, there will be a single host.

ES\_HOST – a variable containing a host and port where Elasticsearch master is up and running, e.g. node1:9200.

We will set these variables in the environment file, which is used in CentOS 7 Linux operating system to set variable values when a computer boots. Thus, if a connection is interrupted for any reason, it will not be necessary to set these values manually anymore.

Start the vi text editor as a super user. To do so, type the following command and press Enter:

sudo vi /etc/environment

Now, you need to press the i button on your keyboard to switch the vi editor to the Insert mode.

Either type the four lines below, or copy and paste then into the vi editor. To paste the date into vi, simply right click somewhere in the vi window.

METRON\_HOME="/usr/metron/0.4.3"

ZOOKEEPER="metronserver.localdomain:2181"

ES\_HOST="metronserver.localdomain:9200"

BROKERLIST="metronserver.localdomain:6667"

First, press the Esc (Escape) button on your keyboard to exit the Insert mode. Then save the file using the following key combination

:wq

And press Enter.

To confirm that variables’ values have been set successfully, close the Putty window, then start Putty again and connect to your single-node cluster. Type the following commands (and hit Enter after each line) to see the values stored in these environment variables.

echo $METRON\_HOME

echo $ZOOKEEPER

echo $ES\_HOST

echo $BROKERLIST

Make a screenshot and add a description.

## Step 6. Pull Zookeper configuration.

Zookeper is a coordinator within an Apache Hadoop cluster. ZooKeeper is a centralized service for maintaining configuration, naming, providing distributed synchronization, and providing group services in distributed applications. Zookeper service itself is distributed and highly reliable.

Pull the Zookeeper configuration information into a local folder using the following command:

sudo $METRON\_HOME/bin/zk\_load\_configs.sh --mode PULL -z $ZOOKEEPER -o $METRON\_HOME/config/zookeeper/ -f

Make a screenshot and add a description.

## Step 7. Install Squid proxy.

Install Squid proxy by executing the command:

sudo yum install -y squid

Make a screenshot and add a description.

Strart Squid proxy with this command:

sudo service squid start

Make a screenshot and add a description.

## Step 8. Retrieve Alexa Top Sites data.

Navigate to your user’s home directory:

cd ~

Download Alexa Top Sites Data:

wget http://s3.amazonaws.com/alexa-static/top-1m.csv.zip

Unzip the downloaded zip archive:

unzip top-1m.csv.zip

Copy data for top 10,000 web sites into a new file, top-10k.csv

head -n 10000 top-1m.csv > top-10k.csv

Make a screenshot and add a description.

## Step 9. Use the Bloom filter to estimate the number of typosquatted domains.

Configuration of the Bloom filter requires us to estimates two numbers: the number of items in the Bloom filter and the probability of false positives. These numbers will be used to determine the size of the Bloom filter.

First, we will run flatfile\_summarizer.sh  in the CONSOLE output mode to display and count the number of typosquatted domain names across the entire document. This will provide us with the numbers for the next step.

Create a file ~/extractor\_count.json, which will hold the extractor configuration:

vi ~/extractor\_count.json

Press the i button on your keyboard to switch the vi editor to the Insert mode.

Copy and paste the following content:

{

"config" : {

"columns" : {

"rank" : 0,

"domain" : 1

},

"value\_transform" : {

"domain" : "DOMAIN\_REMOVE\_TLD(domain)"

},

"value\_filter" : "LENGTH(domain) > 0",

"state\_init" : "0L",

"state\_update" : {

"state" : "state + LENGTH( DOMAIN\_TYPOSQUAT( domain ))"

},

"state\_merge" : "REDUCE(states, (s, x) -> s + x, 0)",

"separator" : ","

},

"extractor" : "CSV"

}

Press the Esc (Escape) button on your keyboard to exit the Insert mode. Then save the file using the following key combination:

:wq

Press Enter.

The extractor configuration uses the following properties:

* columns: the number of columns in the CSV data format. There are 2 columns, rank at the first position and domain at the second position.
* separator: commas are used to separate columns.
* value\_transform: transform values in the domain column by removing top level domain data such as .com, .net, .org, etc.
* value\_filter: discard empty values.
* state\_init: initialize the state to 0. The 0L value means that 0 of the Long Integer type should be used.
* state\_update: update the state with the number of typosquatted domains per domain
* state\_merge: partial sums from several threads will be merged into the total.

Run the flatfile\_summarizer.sh with the configuration file we created above:

sudo $METRON\_HOME/bin/flatfile\_summarizer.sh -i ~/top-10k.csv -e ~/extractor\_count.json -p 5 -om CONSOLE

Make a screenshot and add a description.

The second from the bottom line of the output should be “Processed 9999 - \”

And the bottom line should display 3496552, which is the total possible number of elements in the Bloom filter. We will use this number in the actual Bloom filter in the next step.

## Step 10. Generate the Bloom Filter

Now we can generate the Bloom filter to do the actual work. The extractor configuration file will be adapted from the previous step using the value we obtained (3496552). Then, we will use the extractor to generate the Bloom filter rather than the sum of the typosquatted domains.

The following settings are different from the previous step:

* state\_init: This setting dictates to create a bloom filter initialized with two values
  + The size of 3496552 we calculated in the previous step
  + The false positive probability (0.001, which is 0.1%)
* state\_update: Update the bloom filter (the state variable) with each typosquatted domain
* state\_merge: Merge the bloom filters generated per thread into a final, single bloom filter.

Create a file ~/extractor\_filter.json, which will hold new extractor configuration:

vi ~/extractor\_filter.json

Press the i button on your keyboard to switch the vi editor to the Insert mode.

Copy and paste the following content:

{

"config" : {

"columns" : {

"rank" : 0,

"domain" : 1

},

"value\_transform" : {

"domain" : "DOMAIN\_REMOVE\_TLD(domain)"

},

"value\_filter" : "LENGTH(domain) > 0",

"state\_init" : "BLOOM\_INIT(3496552, 0.001)",

"state\_update" : {

"state" : "REDUCE( DOMAIN\_TYPOSQUAT( domain ), (s, x) -> BLOOM\_ADD(s, x), state)"

},

"state\_merge" : "BLOOM\_MERGE(states)",

"separator" : ","

},

"extractor" : "CSV"

}

Press the Esc (Escape) button on your keyboard to exit the Insert mode. Then save the file using the following key combination:

:wq

Press Enter.

Now we can use the Bloom filter in HDFS to store data in the  /tmp/reference/alexa10k\_filter.ser file. To do so execute this command:

sudo $METRON\_HOME/bin/flatfile\_summarizer.sh -i ~/top-10k.csv -o /tmp/reference/alexa10k\_filter.ser -e ~/extractor\_filter.json -p 5 -om HDFS

To ensure the Bloom filter functions as expected, execute commands to see if google, gogle, github, and gituub are included into the set of typosquatted domains. Note: google and github are the correct domain names and should not be present in the set of typosquatted domains.

Start the Stellar language command line interface, which helps in debugging, troubleshooting and learning Stellar. To do so, execute this command:

sudo $METRON\_HOME/bin/stellar -z $ZOOKEEPER

Now, submit queries for each of the four domain names:

BLOOM\_EXISTS(OBJECT\_GET('/tmp/reference/alexa10k\_filter.ser'), 'gogle')

BLOOM\_EXISTS(OBJECT\_GET('/tmp/reference/alexa10k\_filter.ser'), 'google')

BLOOM\_EXISTS(OBJECT\_GET('/tmp/reference/alexa10k\_filter.ser'), 'github')

BLOOM\_EXISTS(OBJECT\_GET('/tmp/reference/alexa10k\_filter.ser'), 'gituub')

Make a screenshot and add a description.

There should be a bigger delay for the first query and shorter for the subsequent calls since the results for the subsequent calls will be pulled from the cache created by the first query.

Exit Stellar command line by typing:

quit

## Step 11. Squid data parser

Create the squid topic in Metron with this command:

/usr/hdp/current/kafka-broker/bin/kafka-topics.sh --zookeeper $ZOOKEEPER --create --topic squid --partitions 1 --replication-factor 1

Start the squid parser:

sudo $METRON\_HOME/bin/start\_parser\_topology.sh -z $ZOOKEEPER -s squid

Make a screenshot and add a description.

## Step 12. Set up Enrichment, Threat Intel and Threat Triage

The squid parser is running now. Next, we should create an enrichment to add a field **is\_potential\_typosquat** which will help to determine if a domain is potentially typosquatted. We will set up an alert for typosquatted domains and triage those alerts.

In the enrichment configuration file below, the following changes are made:

* A new field is\_potential\_typosquat is created to indicate whether the domain is typosquatted according to the Bloom filter of the 10,000 Alexa Top Sites list.
* The is\_alert field is updated according to the is\_potential\_typosquat field.
* A new threat triage rule is added to provide an analyst with a context and a score.

Crete a squid enrichment configuration file:

sudo vi $METRON\_HOME/config/zookeeper/enrichments/squid.json

Press the i button on your keyboard to switch the vi editor to the Insert mode. Copy and paste the following content:

{

"enrichment": {

"fieldMap": {

"stellar" : {

"config" : [

"domain\_without\_tld := DOMAIN\_REMOVE\_TLD(domain\_without\_subdomains)",

"is\_potential\_typosquat := BLOOM\_EXISTS(OBJECT\_GET('/tmp/reference/alexa10k\_filter.ser'), domain\_without\_tld)",

"domain\_without\_tld := null"

]

}

}

,"fieldToTypeMap": { }

},

"threatIntel": {

"fieldMap": {

"stellar" : {

"config" : [

"is\_alert := (exists(is\_alert) && is\_alert) || is\_potential\_typosquat"

]

}

},

"fieldToTypeMap": { },

"triageConfig" : {

"riskLevelRules" : [

{

"name" : "Alexa 10k Typosquat Bloom",

"comment" : "Inspect a bloom filter with potentially typosquatted domains from the top Alexa 10k",

"rule" : "is\_potential\_typosquat != null && is\_potential\_typosquat",

"score" : 10,

"reason" : "FORMAT('%s is a potential typosquatted domain from the top 10k domains from alexa', domain\_without\_subdomains)"

}

],

"aggregator" : "MAX"

}

}

}

Press the Esc (Escape) button on your keyboard to exit the Insert mode. Then save the file using the following key combination:

:wq

Press Enter.

Push the configuration:

sudo $METRON\_HOME/bin/zk\_load\_configs.sh -m PUSH -i $METRON\_HOME/config/zookeeper -z $ZOOKEEPER

## Step 13. Configure Elasticsearch index.

Elasticsearch creates a new index automatically when it received new data. We will adjust the mappings for the indx to add alert as a nested field and ensure that the rest of the fields will be assigned correct data types. To do so, we will specify a template by executing the following command (copy and paste into the command line and then press Enter):

curl -XPOST "http://$ES\_HOST/\_template/squid\_index" -d '{

"template": "squid\_index\*",

"mappings": {

"squid\_doc": {

"dynamic\_templates": [

{

"timestamps": {

"match": "\*:ts",

"match\_mapping\_type": "\*",

"mapping": {

"type": "date",

"format": "epoch\_millis"

}

}

},

{

"threat\_triage\_score": {

"mapping": {

"type": "float"

},

"match": "threat:triage:\*score",

"match\_mapping\_type": "\*"

}

},

{

"threat\_triage\_reason": {

"mapping": {

"type": "text",

"fielddata": "true"

},

"match": "threat:triage:rules:\*:reason",

"match\_mapping\_type": "\*"

}

}

],

"properties" : {

"action" : {

"type" : "text","fielddata" : true

},

"bytes" : {

"type" : "long"

},

"code" : {

"type" : "long"

},

"domain\_without\_subdomains" : {

"type" : "text","fielddata" : true

},

"elapsed" : {

"type" : "long"

},

"full\_hostname" : {

"type" : "text","fielddata" : true

},

"guid" : {

"type" : "keyword"

},

"ip\_dst\_addr" : {

"type" : "ip"

},

"ip\_src\_addr" : {

"type" : "ip"

},

"is\_alert" : {

"type" : "text","fielddata" : true

},

"is\_potential\_typosquat" : {

"type" : "boolean"

},

"method" : {

"type" : "text","fielddata" : true

},

"original\_text" : {

"type" : "text","fielddata" : true

},

"source:type" : {

"type" : "keyword"

},

"timestamp" : {

"type" : "date",

"format": "epoch\_millis"

},

"url" : {

"type" : "text","fielddata" : true

},

"alert" : {

"type" : "nested"

}

}

}

}

}'

Make a screenshot and add a description.

## Step 14. Generate Sample Data

We will now use squidclient to visit a regular domain and typosquatted domains, and send the data to kafka. To do so, run this command:

squidclient <http://www.github.com>

Make a screenshot and add a description.

squidclient <http://gituub.com/apache/metron>

Make a screenshot and add a description.

Feed the generated squid access.log into Kafka:

sudo cat /var/log/squid/access.log | /usr/hdp/current/kafka-broker/bin/kafka-console-producer.sh --broker-list $BROKERLIST --topic squid

Make a screenshot and add a description.

## Step 15. Investigate alerts with Metron Alerts UI.

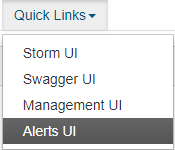
The data, which we fed into Kafka, should have reached the Elasticsearch squid index. Let's investigate triggered alerts using Metron Alerts UI. Navigate to Ambari, enter username and password. In the left hand side menu, click on Metron.



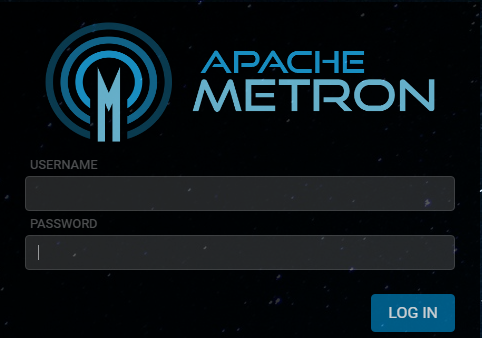
On the top of the screen, in its middle section, click the Quick Links menu.



The menu will display a drop-down list. Click Alerts UI.



A new window will open with a prompt to enter username and password.

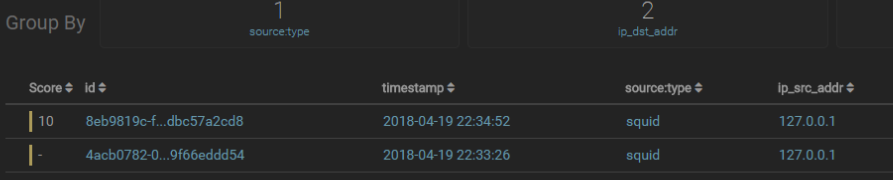


Use the following credentials:

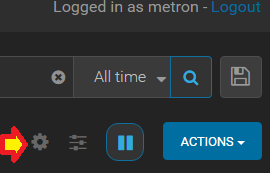
Username: metron

Password: Smoothmetron2

In the Metron UI, you should see the two records from squid. One with score 10 and another one without a score.

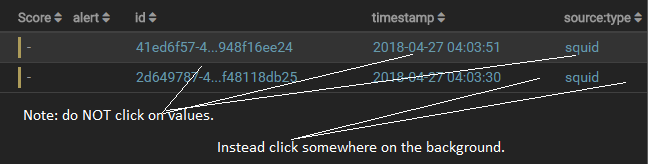


Add the Score column to the table. To do so, click the settings button in the top right corner:

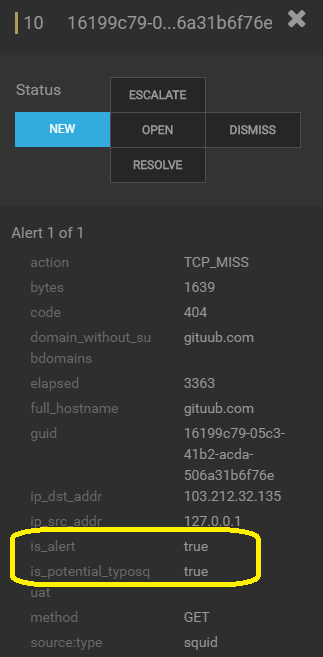


In the settings screen, scroll down until is\_alert appears. Check the box next to it and click the Save button.

To see record details and to change its status, click somewhere within the record – do not click on any value, but click on the background within the record:



Then, you will see more details and will have the ability to change the status to Escalate, Dismiss, Resolve, or Open.



Make a screenshot and add a description.

The task is complete.

## References

<https://www.cnn.com/2014/03/01/studentnews/trump-cybersquatting-lawsuit/index.html>

<https://www.gpo.gov/fdsys/pkg/CRPT-106srpt140/html/CRPT-106srpt140.htm>

<https://www.upcounsel.com/cybersquatting-examples>

<https://nakedsecurity.sophos.com/typosquatting/>

<https://www.wired.com/images_blogs/threatlevel/2011/09/Doppelganger.Domains.pdf>

<https://securitycommunity.tcs.com/infosecsoapbox/articles/2018/04/23/typosquatting-detection-using-advanced-analytics-0>

<https://www.usatoday.com/story/tech/news/2016/12/01/hackers-use-typo-squatting-lure-unwary-url-hijacking/94683460/>

<https://nakedsecurity.sophos.com/2014/03/24/bitcoin-user-loses-10k-to-typosquatters/>

H. Bloom “Space/time trade-offs in hash coding with allowable errors.”

J. Spaulding, S. Upadhyaya and A. Mohaisen. (2016). The Landscape of Domain Name Typosquatting: Techniques and Countermeasures. In prof. of 2016 11th International Conference on Availability, Reliability and Security, pp. 284-289.

<https://github.com/apache/metron/tree/master/use-cases/typosquat_detection>

Alexa Top 500 Sites <https://www.alexa.com/topsites>

Alexa Top 10,000 Sites <https://aws.amazon.com/alexa-top-sites/>

Proxy and DNS Log Files Concepts <https://fedvte.usalearning.gov/courses/NMOT/course/videos/pdf/NMOT_D01_S05_T02_STEP.pdf>

Bitsquatting. <http://dinaburg.org/bitsquatting.html>

Bitsquatting. DNS hacking without exploitation. <http://media.blackhat.com/bh-us-11/Dinaburg/BH_US_11_Dinaburg_Bitsquatting_WP.pdf>

<http://www.squid-cache.org>