1. Add the sample web log data to Kibana.
2. Answer the following questions:
   * In the last 7 days, how many unique visitors were located in India?
     + 248
   * In the last 24 hours, of the visitors from China, how many were using Mac OSX?
     + 9
   * In the last 2 days, what percentage of visitors received 404 errors? How about 503 errors?
     + 1.266% - 404
     + 93.67% - 503
   * In the last 7 days, what country produced the majority of the traffic on the website?
     + China
   * Of the traffic that's coming from that country, what time of day had the highest amount of activity?
     + Noon to 1:00PM
   * List all the types of downloaded files that have been identified for the last 7 days, along with a short description of each file type (use Google if you aren't sure about a particular file type).
     + .css- Cascading style sheets file. Used to describe how HTML elements are displayed on the screen.
     + .deb- Debian software package
     + .gz- Gnu zip (gzip) file. Archive file compressed by gzip.
     + .rpm- RedHat Package manager
     + .zip- compressed (or archive) folder
3. Now that you have a feel for the data, Let's dive a bit deeper. Look at the chart that shows Unique Visitors Vs. Average Bytes.
   * Locate the time frame in the last 7 days with the most amount of bytes (activity).
     + 21-09-23 @ 15:00:00
   * In your own words, is there anything that seems potentially strange about this activity?
     + Traffic (bytes) is almost double of the next highest event, but this was performed by only 1 unique visitor.
4. Filter the data by this event.
   * What is the timestamp for this event?
     + 21-9-23 @ 15:58:15
   * What kind of file was downloaded?
     + .gz
   * From what country did this activity originate?
     + China
   * What HTTP response codes were encountered by this visitor?
     + Response code of 200 (OK, request has succeeded)
5. Switch to the Kibana Discover page to see more details about this activity.
   * What is the source IP address of this activity?
     + 1.145.31.121
   * What are the geo coordinates of this activity?
     + geo.coordinates:{ "lat": 28.28980556, "lon": -81.43708333 }
   * What OS was the source machine running?
     + Win 8
   * What is the full URL that was accessed?
     + https://artifacts.elastic.co/downloads/kibana/kibana-6.3.2-linux-
   * From what website did the visitor's traffic originate?
     + http://www.elastic-elastic-elastic.com/success/aleksandr-serebrov
6. Finish your investigation with a short overview of your insights.
   * What do you think the user was doing?
     + Trying to download Kibana install package archive
   * Was the file they downloaded malicious? If not, what is the file used for?
     + Initial thoughts are that the file is indeed malicious. If it is not, it is an installer for an out of date Kibana package archived into a tarball using gzip.
   * Is there anything that seems suspicious about this activity?
     + Redirect from an unknown and untrusted website. File could contain malicious content
     + File size is way too small for the install package of Kibana (current version is 271 MB)
     + Downloading an out of date version of Kibana (6.3.2) which would have several vulnerabilities as shown on NIST: NVD (<https://nvd.nist.gov/vuln/search/results?form_type=Basic&results_type=overview&query=kibana&search_type=all&isCpeNameSearch=false> )
   * Is any of the traffic you inspected potentially outside of compliance guidelines?
     + The referer url does not exist, and is available per a whois lookup. This means that it could be a fake or spoofed site that an attacker could use to have users download malicious files unknowingly. This pertains to the integrity of the data as we cannot confirm the file downloaded was not tampered with.

Kibana Continued:

Open the Metrics page for the web machine you attacked and answer the following questions:

Which of the VM metrics were affected the most from this traffic?

Outbound traffic was the most affected. Memory was also greatly affected.

Bonus: Notice that your wget loop creates a lot of duplicate files on your jump box.

Write a command to delete all of these files at once.

rm ./\*.html\*

Find a way to run the wget command without generating these extra files.

Look up the flag options for wget and find the flag that lets you choose a location to save the file it downloads.

-O

Save that file to the Linux directory known as the "void" or the directory that doesn't save anything

while [ true ]; do wget -O /dev/null 10.0.0.9; done

Bonus: Write a nested loop that sends your wget command to all three of your web VMs over and over.

while [ true ];

do

wget -O /dev/null 10.0.0.9

wget -O /dev/null 10.0.0.10

done