Week: #3 Understand working of HTTP Headers

	Understand working of HTTP headers: Conditional Get: If-Modified-Since HTTP Cookies: Cookie and Set-Cookie Authentication: Auth-Basic Design a web page that has one embedded page (e.g. image) and sets a cookie and enables authentication. You are required to configure the web server (e.g. apache) with authentication mechanism. Show the behavior of conditional get when embedded objects is modified and when it is not (you can just change the create date of the embedded object). Decode the Basic-Auth header using Base64 mechanism as per the password setup. **Observation**: Show the behavior of browser when is cookie is set and when cookie is removed.

Week: 5

Understanding Working of HTTP Headers

Question: Understand working of HTTP headers

Conditional Get: If-Modified-Since

HTTP Cookies: Cookie and Set-Cookie

Authentication: Auth-Basic

Design a web page that has one embedded page (e.g. image) and sets a cookie and enables authentication. You are required to configure the web server (e.g. apache) with authentication mechanism. Show the behavior of conditional get when embedded objects are modified and when it is not (you can just change the create date of the embedded object). Decode the Basic-Auth header using Base64 mechanism as per the password setup.

Observation: Show the behavior of browser when is cookie is set and when cookie is removed.

Solution: Analyzing Basic Authentication and Cookies

The three parts of experiment are:

1. Password Authentication
2. Cookie Setting
3. Conditional get

Steps of Execution (for Password Authentication)

1. Executing the below commands on the terminal.

--> To update and integrate the existing softwares

sudo apt-get update

--> To install the apache utility

sudo apt-get install apache2 apache2-utils

```
osboxes@osboxes: ~
osboxes@osboxes:~$ sudo apt-get install apache2-utils
[sudo] password for osboxes:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages will be upgraded:
  apache2-utils
1 upgraded, 0 newly installed, 0 to remove and 247 not upgraded.
Need to get 81.7 kB of archives.
After this operation, 0 B of additional disk space will be used.
Get:1 http://us.archive.ubuntu.com/ubuntu xenial-updates/main amd64 apache2-utils amd64 2.4.18-2ub
untu3.10 [81.7 kB]
Fetched 81.7 kB in 1s (45.5 kB/s)
(Reading database ... 217540 files and directories currently installed.)
Preparing to unpack .../apache2-utils_2.4.18-2ubuntu3.10_amd64.deb ...
Unpacking apache2-utils (2.4.18-2ubuntu3.10) over (2.4.18-2ubuntu3.9) ...
Processing triggers for man-db (2.7.5-1) ...
Setting up apache2-utils (2.4.18-2ubuntu3.10) ...
osboxes@osboxes:~$
```

--> Provide username and password to set authentication

sudo htpasswd -c /etc/apache2/.htpasswd ANY_USERNAME

```
osboxes@osboxes:~$ sudo htpasswd -c /etc/apache2/.htpasswd netwo
New password:
Re-type new password:
Adding password for user netwo
osboxes@osboxes:~$ sudo cat /etc/apache2/.htpasswd
netwo:$apr1$6YdDa0Ti$ELrUaOlQ/jun9TTU1PYKu/
osboxes@osboxes:~$
```

Here “netwo” is the username. Also, password is entered twice.

--> View the authentication

sudo cat /etc/apache2/.htpasswd

2. To setup the authentication phase, execute the following commands. Configuring Access control within the Virtual Host Definition.

--> Opening the file for setting authentication

sudo nano /etc/apache2/sites-available/000-default.conf

```
<VirtualHost*:80>
        ServerAdmin webmaster@localhost
        DocumentRoot  /var/www/html
        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined
        <Directory "/var/www/html">
                AuthType Basic
                AuthName "RESTRICTED"
                AuthUserFile /etc/apache2/.htpasswd
                Require valid-user
        </Directory>
</VirtualHost>
```

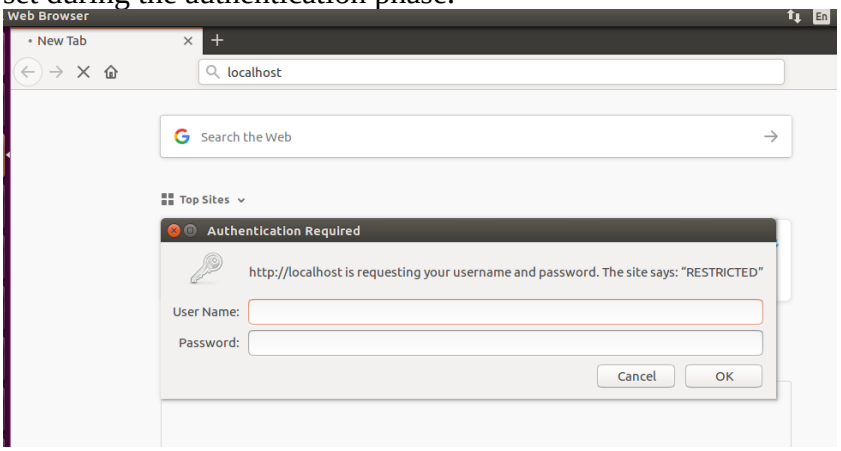
```
 GNU nano 2.5.3        File: /etc/apache2/sites-available/000-default.conf

<VirtualHost *:80>


        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/html

        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined

                <Directory "/var/www/html">
                        AuthType Basic
                        AuthName "RESTRICTED"
                        AuthUserFile /etc/apache2/.htpasswd
                        Require valid-user >
        </Directory>

</VirtualHost>
```

3. Password policy implementation is done by restarting the server as:

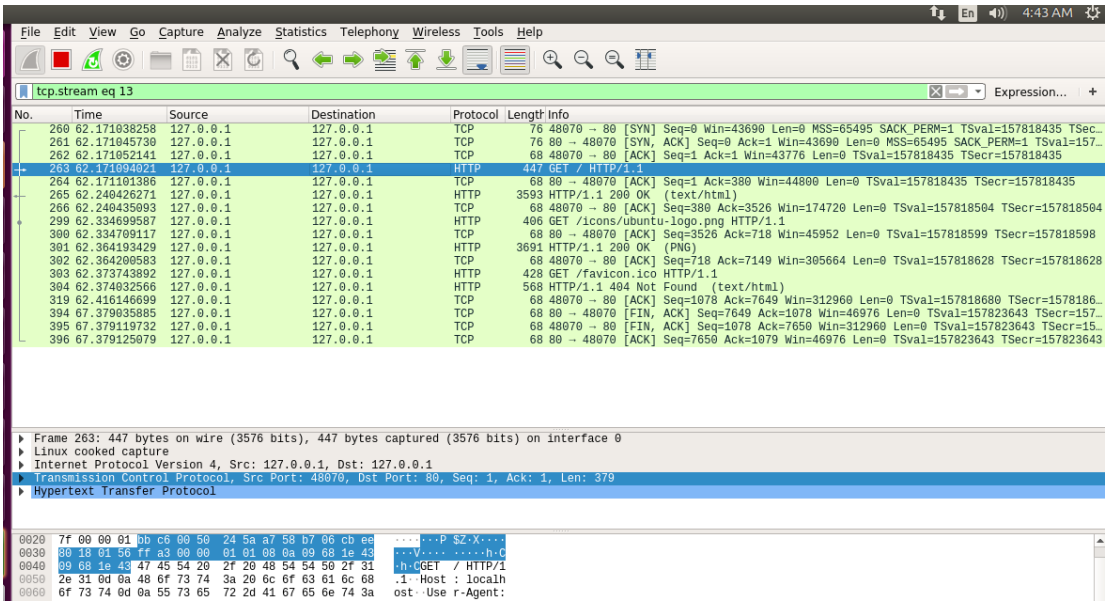
sudo service apache2 restart

```
osboxes@osboxes:~$ sudo service apache2 restart
osboxes@osboxes:~$
```

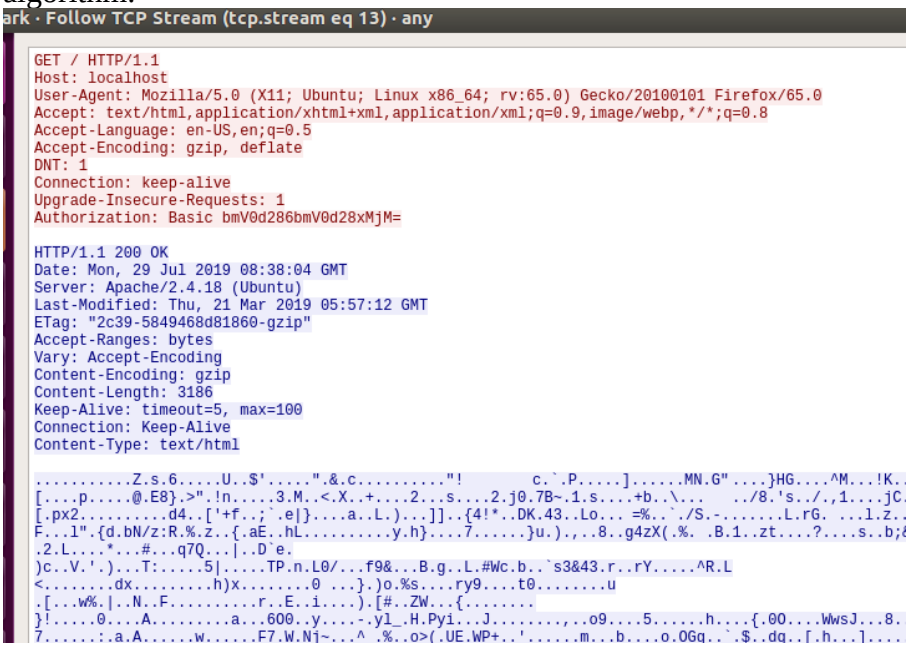
4. The localhost is then accessed using the Firefox browser requiring a username and a password set during the authentication phase.



5. Wireshark is used to capture the packets sent upon the network.



6. Using the “follow TCP stream” on the HTTP message segment the password was retrieved which was encrypted by the base64 algorithm and decryption could be done with same algorithm.



Steps of Execution (Cookie Setting)

1. A PHP file to set the cookie is created which also contains an image in it (placed under the HTML directory) to be accessed once the cookie is set. The following code helped to set the cookie:

<html>
<?php

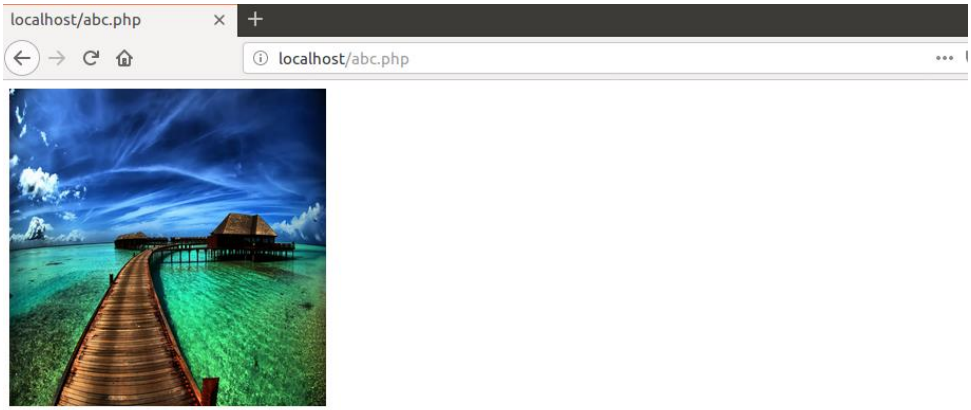
```
        setcookie("namecookie","netqwerty",time()+123);
        setcookie("nickname","work");
?>
<img src= “highres.png” width= “300” height= “300” title= “password” />
</html>
```

```
  GNU nano 2.5.3                File: abc.php

<html>
<?php
setcookie("namecookie","netqwerty",time()+123);
setcookie("nickname","work");
?>
<img src="highres.jpg" width="300" height="300" title="password"/>
</html>
```

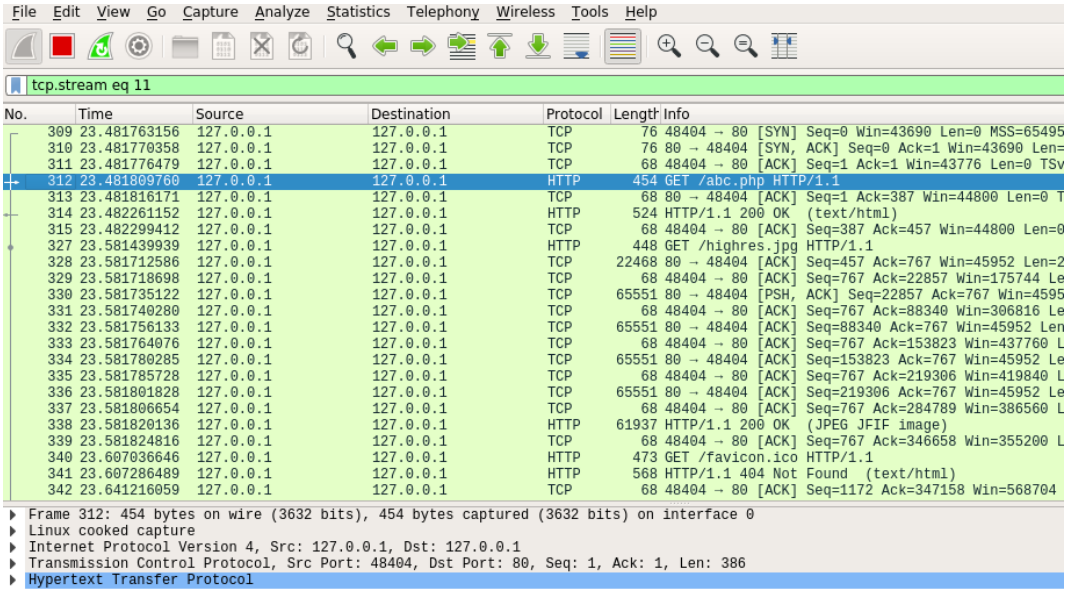
Note: Here you can add any image if required

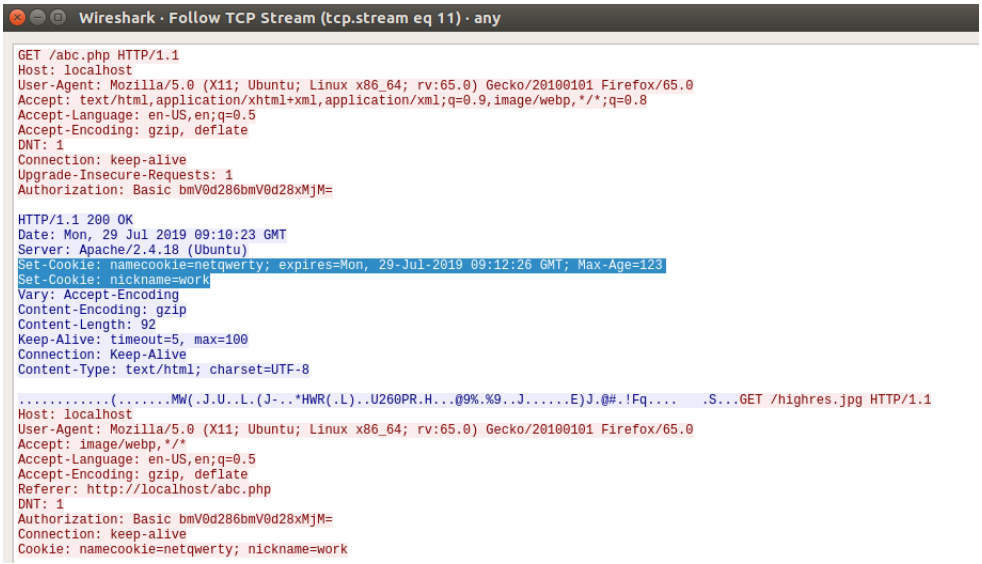
Note: You can capture Cookies mostly during the first time of web access. Hence keep wireshark capture ready before executing the task for the first time.

2. The combined file saved with a .php extension is placed under **/var/www/html** for accessing.



3. The packets are captured using Wireshark and using the “follow TCP stream” which checks for the set-cookie field whether the cookie is set or not set.





The cookie is set as shown in the above screenshot.

Observation: Understand and work out base 64 algorithm and write in your observation. Observe various parameters associated with Cookie in the wireshark capture.

Conditional Get: If-Modified-Since

Before performing the steps below, make sure your browser's cache is empty. (To do this under Firefox, select Tools -> Clear Recent History and check the Cache box). Now do the following:

- Start up your web browser, and make sure your browser's cache is cleared, as discussed above.
- Start up the Wireshark packet sniffer.
- Enter the following URL into your browser http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file2.html
- Your browser should display a very simple five-line HTML file.
- Quickly enter the same URL into your browser again (or simply select the refresh button on your browser)
- Stop Wireshark packet capture, and enter "http" in the display-filter-specification window, so that only captured HTTP messages will be displayed later in the packet-listing window.

Observations:

- ✓ Inspect the contents of the first HTTP GET request from your browser to the server. Do you see an "IF-MODIFIED-SINCE" line in the HTTP GET?
- ✓ Inspect the contents of the server response. Did the server explicitly return the contents of the file? How can you tell?
- ✓ Now inspect the contents of the second HTTP GET request from your browser to the server. Do you see an "IF-MODIFIED-SINCE:" line in the HTTP GET? If so, what information follows the "IF-MODIFIED-SINCE:" header?
- ✓ What is the HTTP status code and phrase returned from the server in response to this second HTTP GET? Did the server explicitly return the contents of the file? Explain.

Repeat the above task with some images on the server.

Attach screenshots wherever necessary.