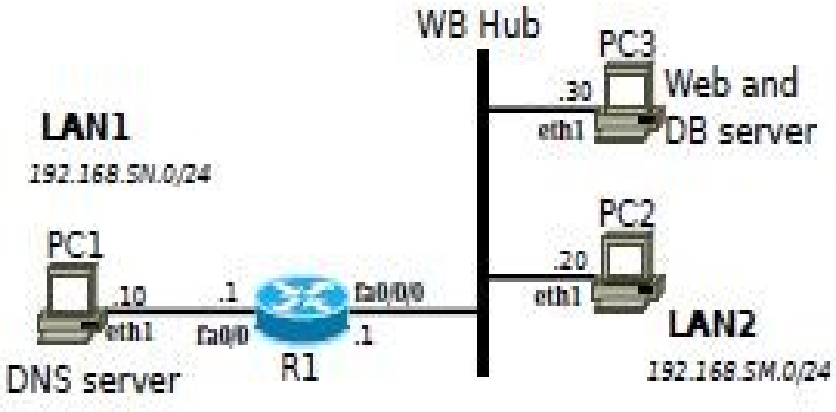
Lab 12: Application Layer

Mark Rutkowski
Arun Kumar Rajendra Kumar

Here, we discuss about setup of DNS, HTTP and HTTPS server. DNS serves as a name to the IP address thereby eliminating the need of remembering IP address. Generally, DNS depends on TCP or UDP and they are at application layer. Only when there is a data truncation, DNS prefers TCP. We set up the following network to start with the experiment.



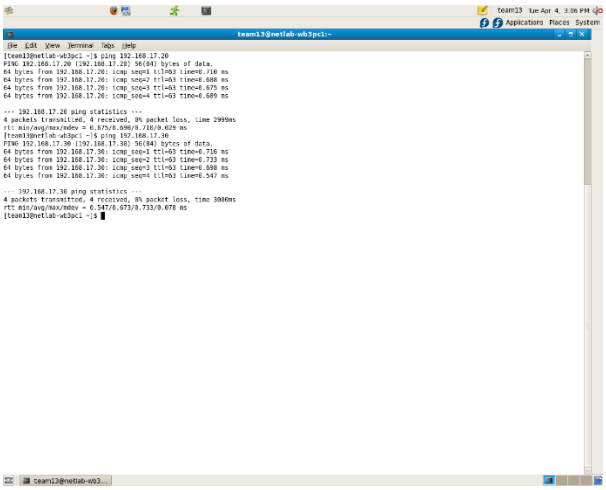
Webmin service is started on PC1 using **sudo/etc/init.d/webmin start** command & http://127.0.0.1:10000 is accessed using the web browser. Changes are made in Address and Topology section, Zone defaults in the DNS main page. Then a master zone is created for forward lookup. Following it, master zone for reverse lookup is created for each LAN. Address record is added to each PC and then DNS server is restarted. Dig is run on PC2 for forward and reverse queries. It is checked by pinging www.wbSN.netlab.local. In the second part, an Apache web server is configured from the root directory i.e. /var/www/html. A html file called as index.html is created and then httpd is configured by making necessary changes in Listen & Server name. Again the web browser is started to access the web page set up on PC3. In the third section, we use telnet command which allows the client host to access web server. When the command for it executed, the server replies with HTTP header and HTML content. For another html file, this part is repeated.

Virtual host refers to running more than one web server on a single machine. It can be either IP-based or name-based. And here we learn how to configure virtual hosts. The old html file created is copied into the new file & some directives are added inside the configuration file. Once, this step is done, the httpd daemon is restarted & web browser is checked for its status. An additional step is performed by running the website with same IP address but different port number.

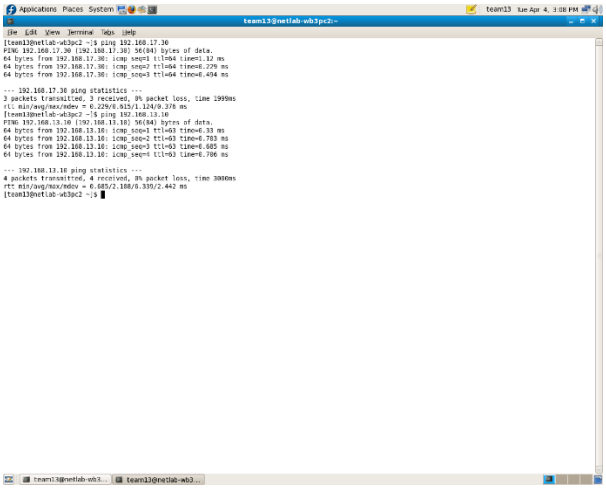
Access control allows access to a given resource only by control policies and user authentication. That feature is tested here. Initially, a html file called index.html is created and the http main configuration file is edited. Changes are made in such a way that only devices in LAN2 can access the page. Then, authentication scheme of Apache is tested by providing the necessary command. Again the web server is restarted. Now the web page is made accessible only when a valid user name and password is provided. It works on IP-based access control and various parameters like the TCP requests, GET messages are studied.

We might be using https connection in our daily lives even without our knowledge. This section describes its usage. HTTPS makes sure that there is data encryption and server authentication in a TCP connection using protocols like SSL or TLS. To play with this, we first make necessary changes in the configuration file by editing the listen directive followed by restarting the web server.

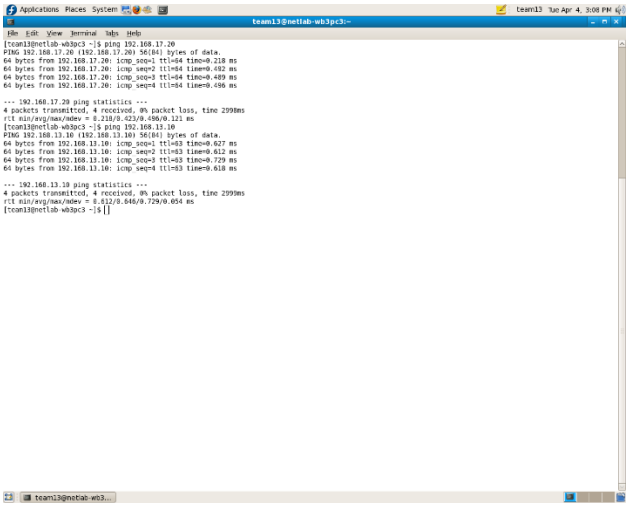
Section 4.1: Topology Setup



This is PC1 pinging PC2 and PC3.

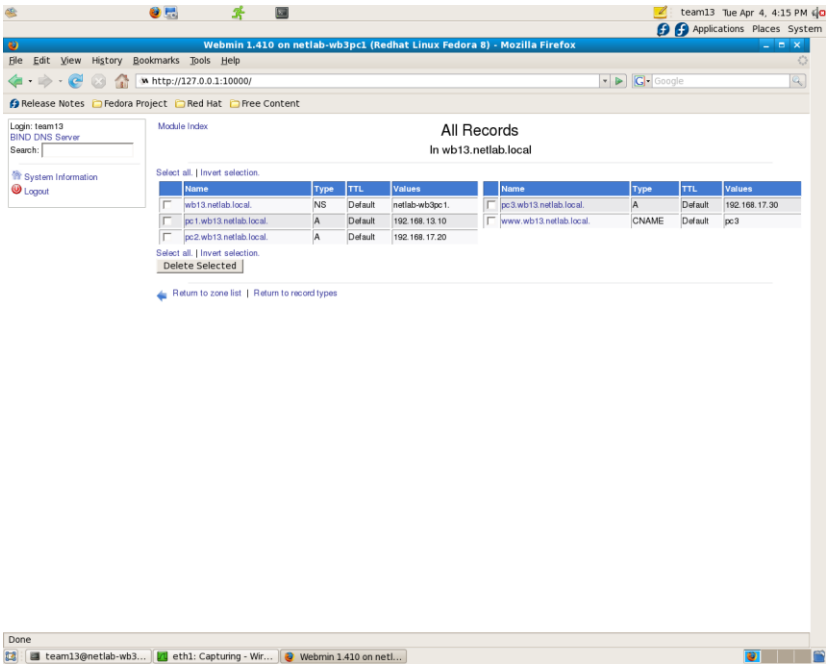


This is PC2 pinging PC1 and PC3.

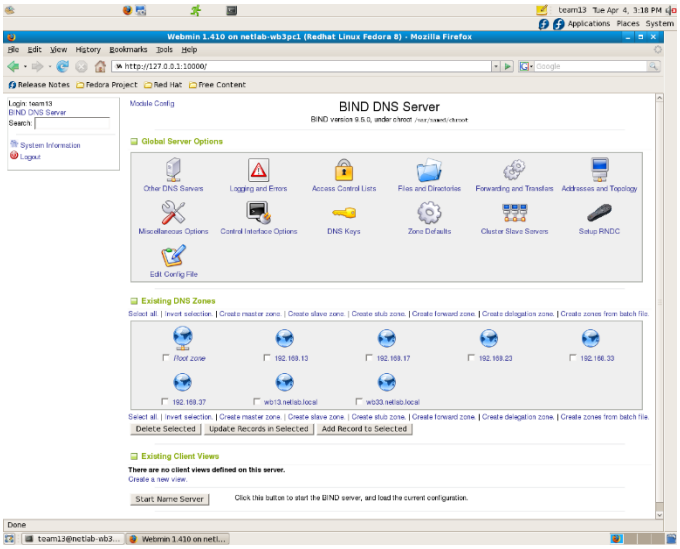


This is PC3 pinging PC1 and PC2.

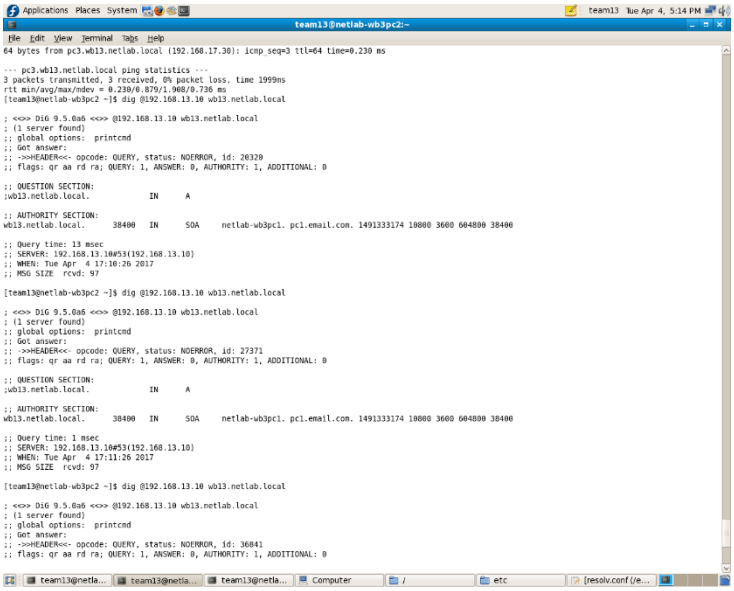
Section 4.2: DNS server configuration



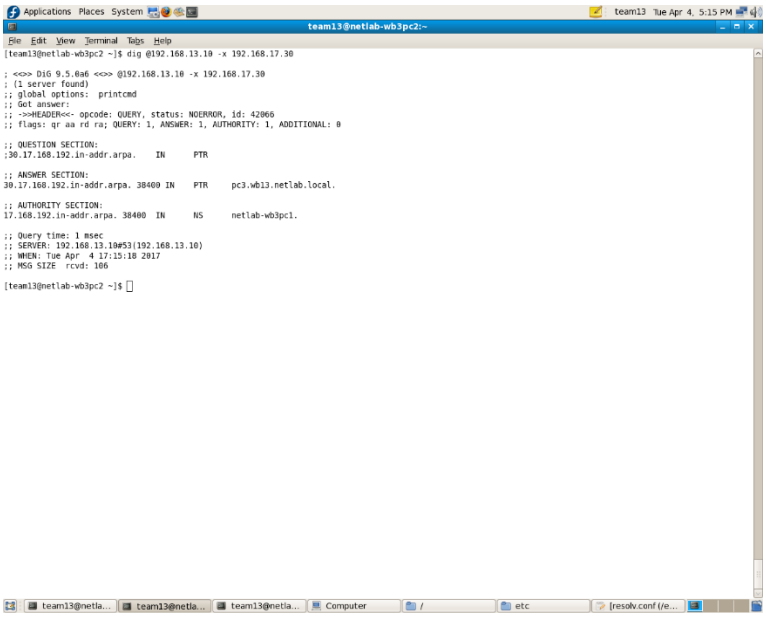
These are the records we added to the forward lookup master zone.



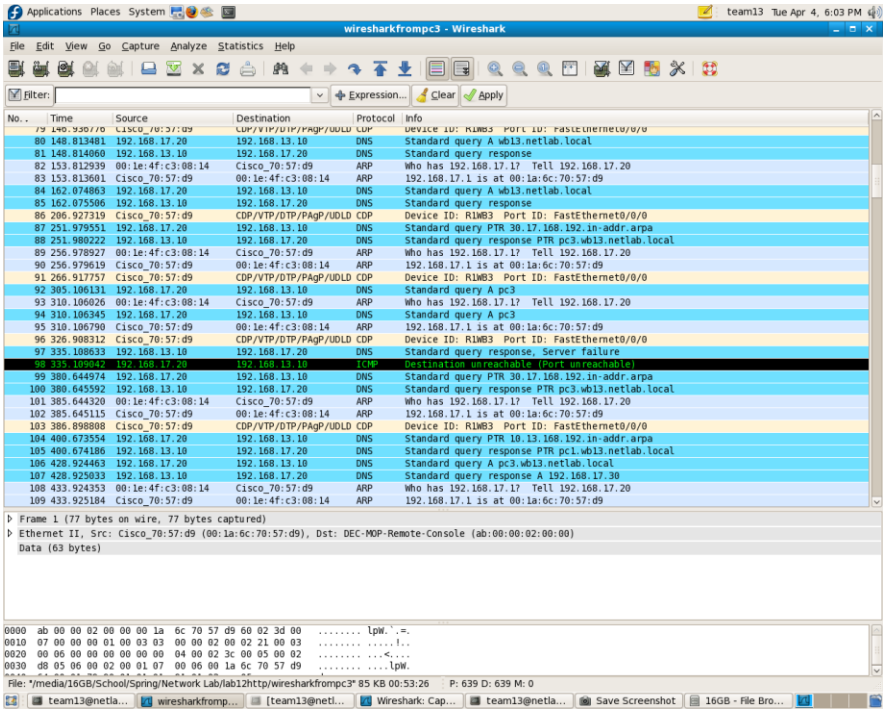
This shows the reverse lookup master zones for 192.168.13 and 192.168.17. The other DNS zones that we did not create in this picture were deleted later.



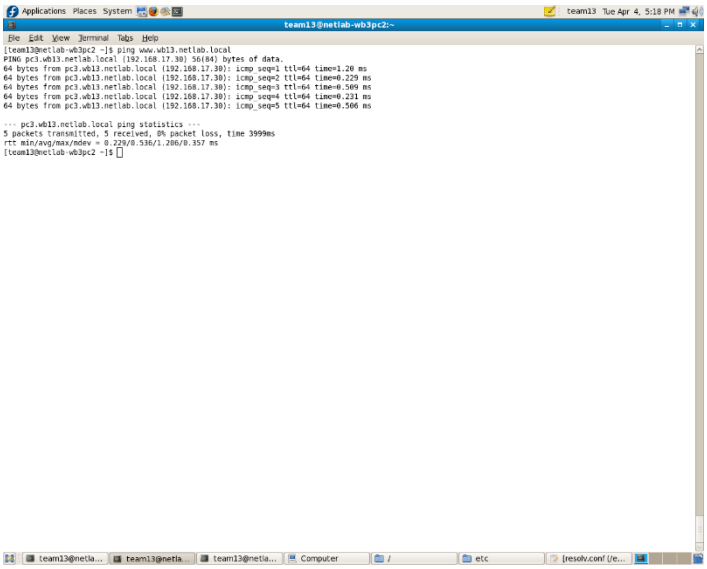
This is the dig result from executing the forward lookup query of the domain wb13.netlab.local.



This shows the dig result from executing the reverse lookup query using PC3's IP address as 192.168.17.30.

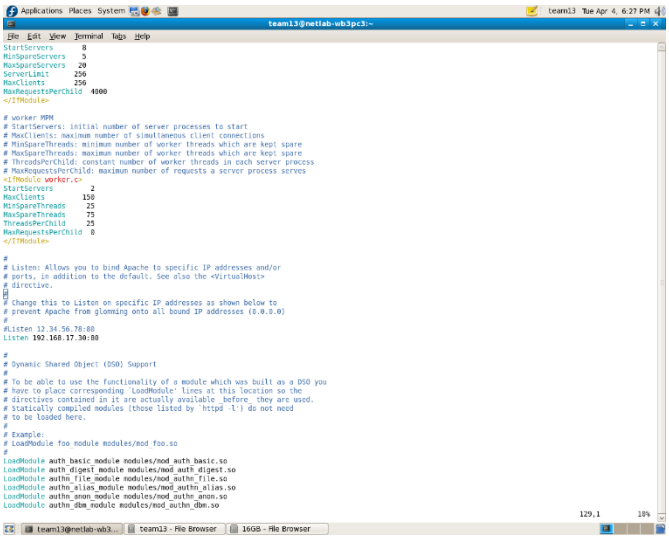


This shows the Wireshark output when we did forward and reverse lookup queries for the DNS server.



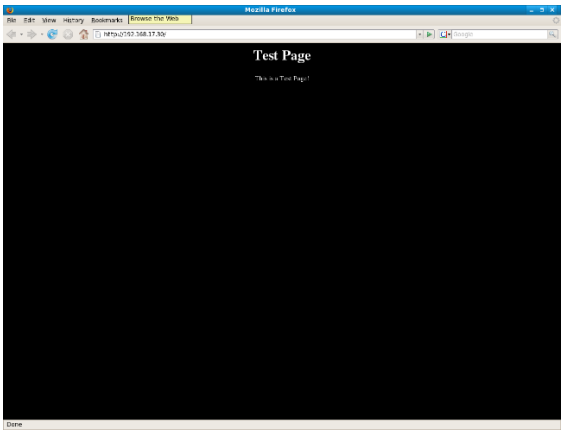
This shows pinging the www.wb13.netlab.local zone from PC2.

Section 4.3: Basic HTTP server configuration

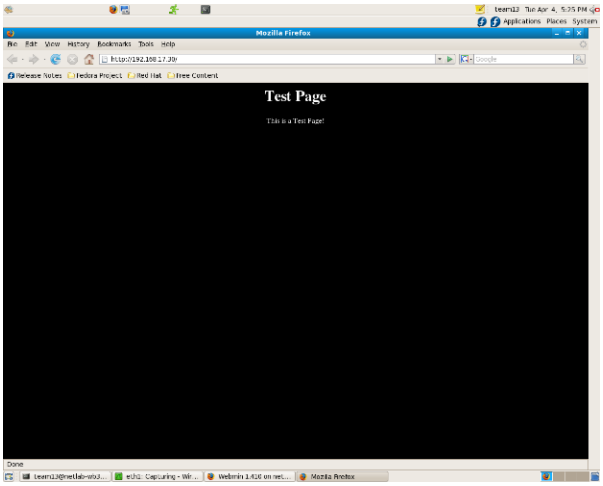


This shows changing the Listen directive in the /etc/htpd/conf/http.conf.

This shows changing the ServerName directive in the /etc/htpd/conf/http.conf.

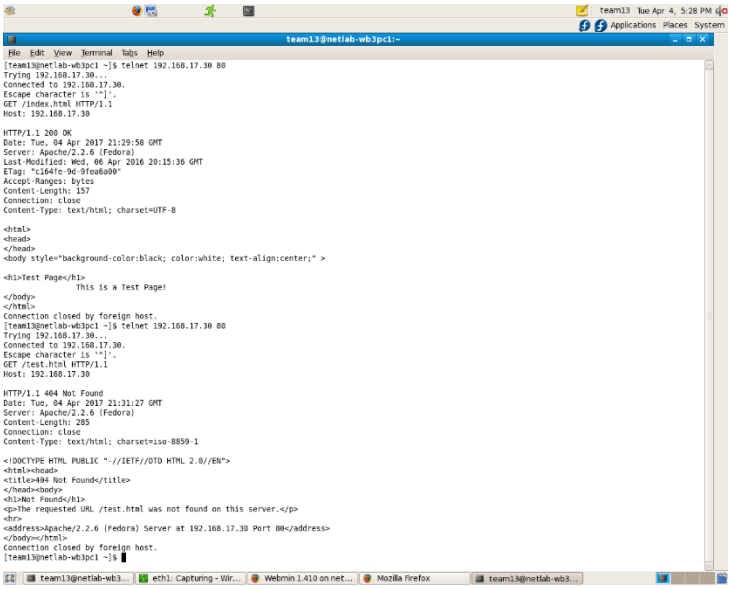


This shows PC2 accessing the web page hosted on the PC3's web server.



This shows PC1 accessing the web page hosted on the PC3's web server.

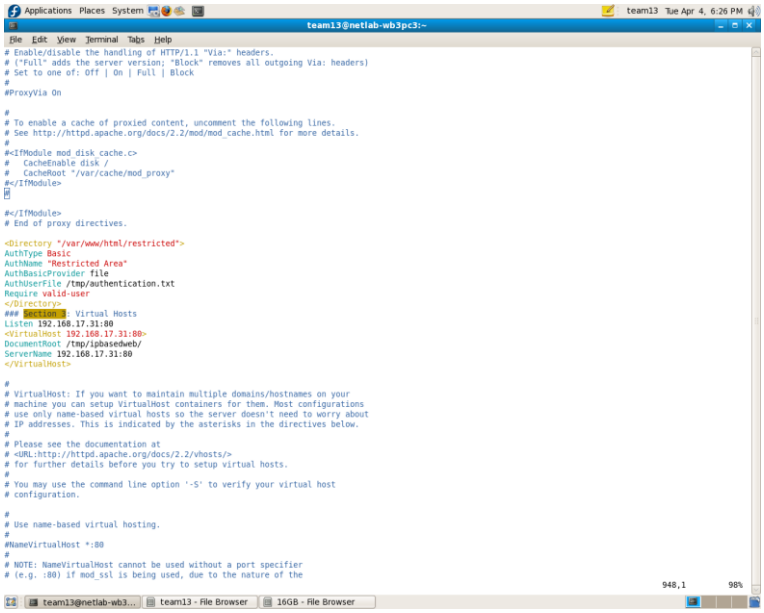
Section 4.4: Browse without browser



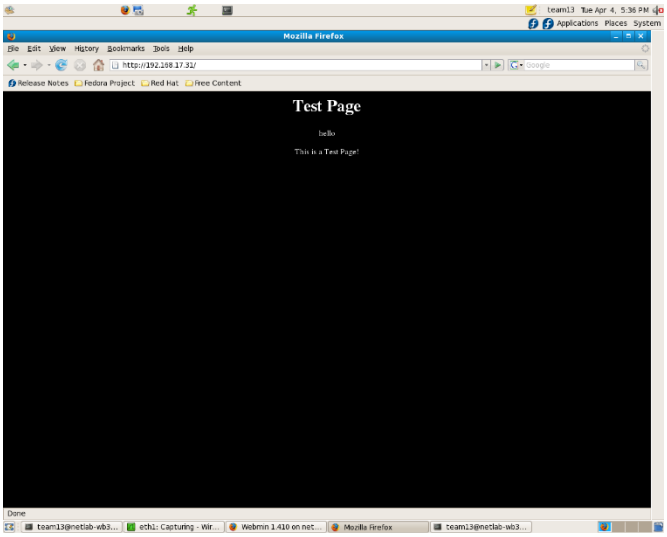
This shows the telnet result from PC1. The top result was successful because it used the file index.html in the GET command, which was located in the /var/www/html directory. The second result failed because it used the file test.html and there was no listing of that in the /var/www/html directory. It throws a 404 Not Found error when it can not find the file and correct HTTP header.

Section 4.5: Virtual host configuration

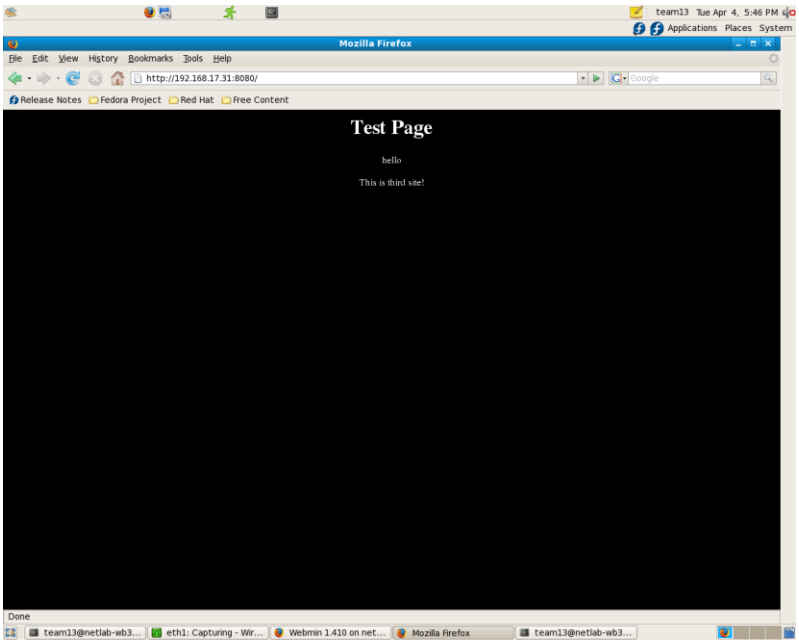
After adding another IP address to PC3 and modifying the httpd configuration file after the Section 3: Virtual Hosts section, we could test whether we can access that web page running on the web server.



This shows the modifications of the Section 3 Virtual Hosts section in the httpd configuration file.



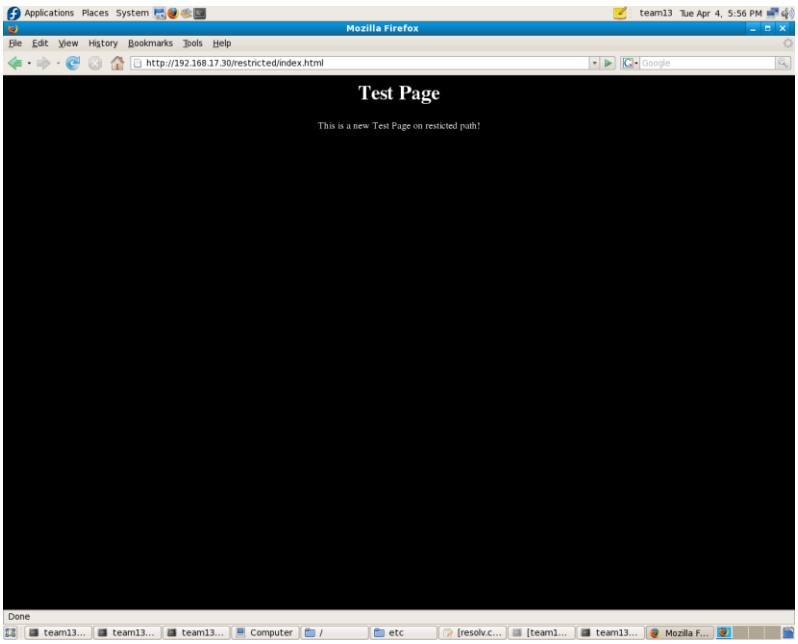
This is PC1 accessing the new website at 192.178.17.31.



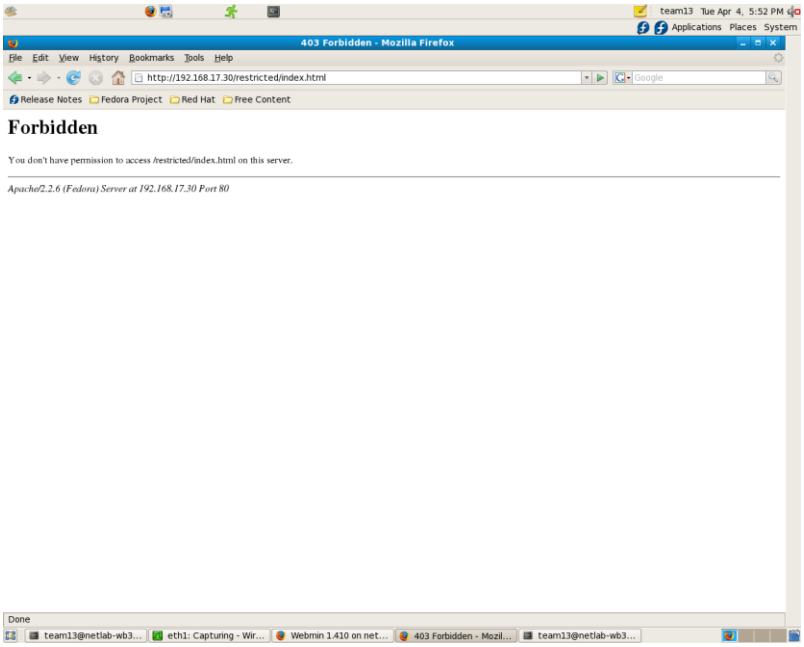
This is PC1 accessing the third website using the same IP address as before 192.168.17.31, but now using port 8080 instead of port 80.

Section 4.6: Access control and basic authentication

After creating the /var/www/html/restricted directory and modifying the httpd configuration file after the Section 3: Virtual Hosts section to allow any devices in LAN2 to access the website but deny all other devices in other LANs, we could test if the rules we set worked.



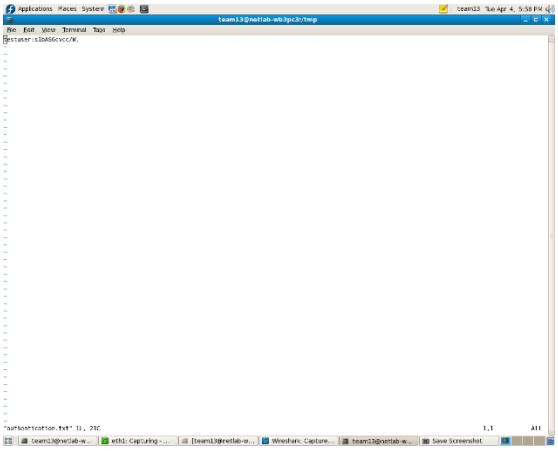
This is PC2 accessing the website at 192.168.17.30/restricted/index.html and it is allowed because it is in LAN2.



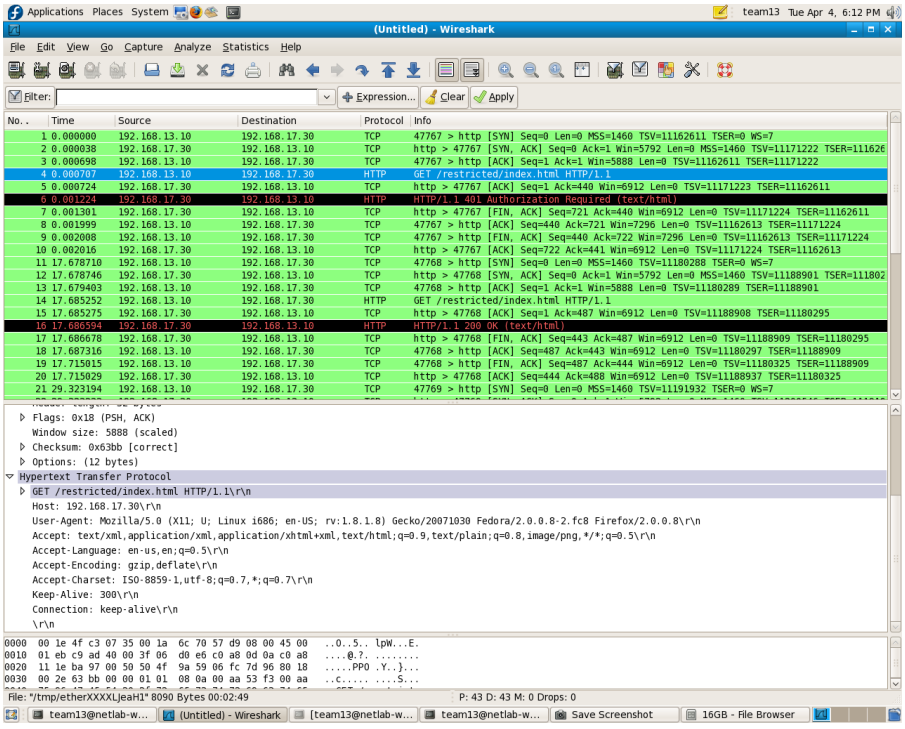
This is PC1 accessing the website at 192.168.17.30/restricted/index.html and it is not allowed because it is in LAN1 and not LAN2.

Instead of forbidding and allowing certain users, we can allow users to authenticate themselves in order to prove whether they can access the website or not. Using htpasswd to create a user and

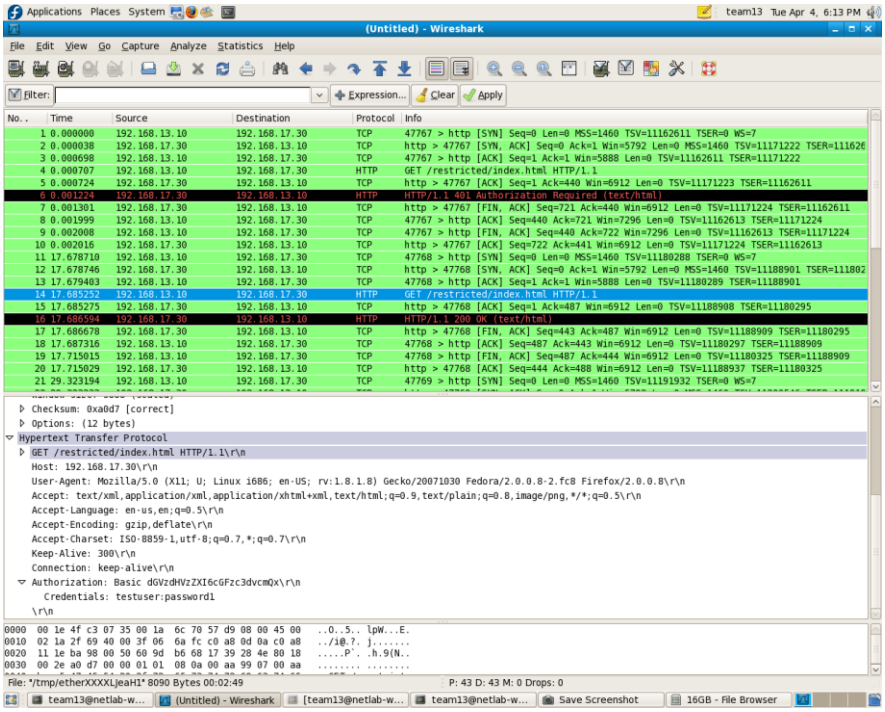
password file in order to access the website and telling the httpd configuration file where to look in order to authenticate the user input, we could verify the test results.



This shows the encrypted password of the user testuser we created.



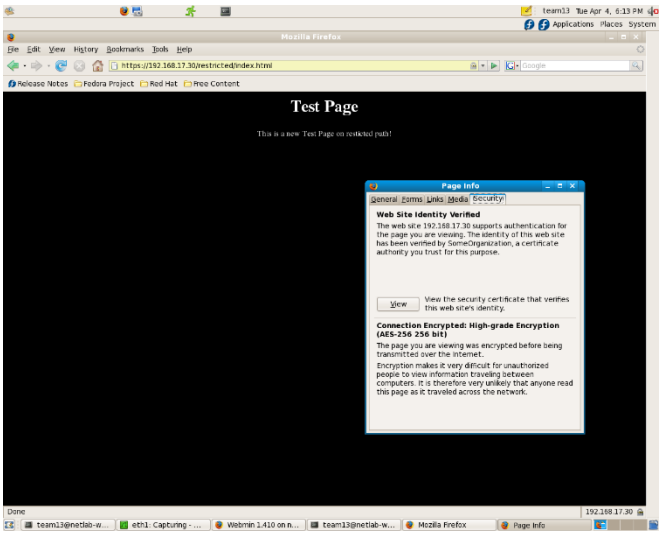
This shows the Wireshark output when PC1 attempted to access the website after entering in the proper credentials. This is before we entered in any credentials and it does not show any authentication in the GET message.



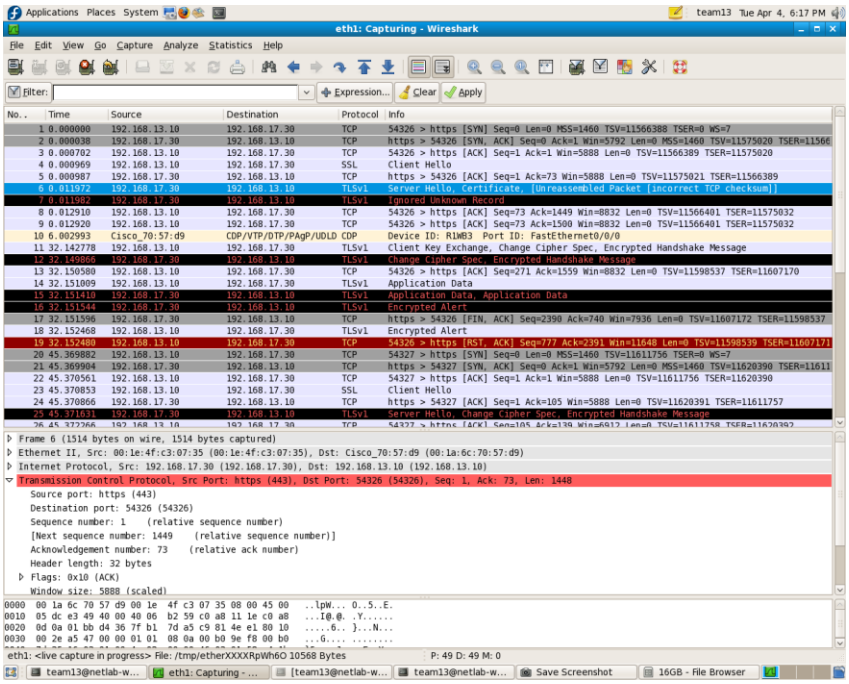
It shows that authentication is required and once the correct username and password were entered, it would be allowed access even though it is in LAN1. It also shows the credentials under the authorization in plaintext with the username: testuser and the password: password1.

Section 4.7: Using https

We changed the configuration file in /etc/httpd/conf.d/ssl.conf to include the IP address of PC3 192.168.17.30 under the Listen directive and to use port 443, which is the default port for secure http access.



This shows PC1 accessing the website using https after entering in the credentials. It shows that it is using AES 256 bit encryption when accessing the website.



This shows the Wireshark output when accessing the website from PC1 using https. It shows the protocols TLS and SSL. It shows the TLS handshake when the client first sends the Hello message. The server will respond with a Hello message acknowledging that it received the client's message. The certificate and key exchange will be established between PC1 and the web server. Then the server will send a Hello message back saying it is done with setting up the certificate and encryption. After this, the data can be transferred and PC1 can access the website from PC3's web server.

R1WB3#show run
Building configuration...

Current configuration : 1149 bytes
!
version 12.4
service timestamps debug datetime msec
service timestamps log datetime msec
no service password-encryption
!
hostname R1WB3
!
boot-start-marker
boot system flash:c2800nm-adventerprisek9-mz.124-24.T8.bin
boot-end-marker
!
logging message-counter syslog
!

```
no aaa new-model
!
!
!
dot11 syslog
ip source-route
!
!
ip cef
!
!
no ip domain lookup
no ipv6 cef
!
multilink bundle-name authenticated
!
!

!
voice-card 0
!

!
!
archive
 log config
  hidekeys
!
!
interface Loopback0
 no ip address
!
interface FastEthernet0/0
 ip address 192.168.13.1 255.255.255.0
 ip virtual-reassembly
 duplex auto
 speed auto
!
interface FastEthernet0/1
 no ip address
 ip virtual-reassembly
 duplex auto
 speed auto
!
interface FastEthernet0/0/0
 ip address 192.168.17.1 255.255.255.0
```

```
 ip virtual-reassembly
 duplex auto
 speed auto
!
no ip forward-protocol nd
no ip http server
no ip http secure-server
!
!
!
ip access-list extended NP
!

!
control-plane
!
!
!
line con 0
 exec-timeout 0 0
 logging synchronous
line aux 0
line vty 0 4
 login
!
scheduler allocate 20000 1000
end

R1WB3#
```

This is the show run output for R1.