Trey Atwood

September 9th, 2025

Web Server Hosting and Encryption

Exploring web servers, and what different traffic looks like to users. Whether or not traffic is encrypted and what to look for to see if it is or isn't.

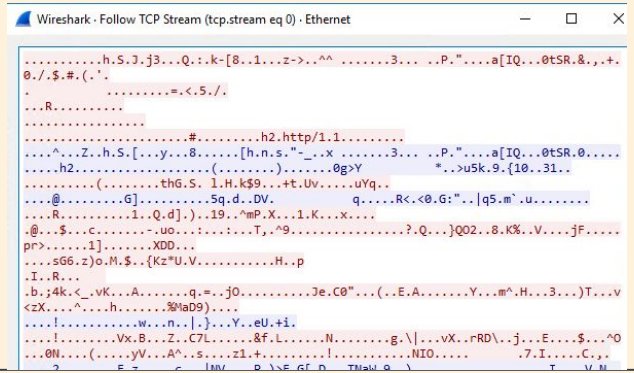
What is the difference between encrypted and unencrypted traffic?

With encrypted traffic the data that is transmitted across the network even if its intercepted, its scrambled in random bytes. So if someone were to intercept the packet on the network they would have no idea what it means, all they would see is scrambled nonsense. Now with unencrypted traffic if the packet were to be intercepted on the network they would be able to intercept the data and see it in plaintext. The unencrypted traffic could contain passwords, usernames, or email addresses in plaintext for whoever intercepted it to read. Unencrypted traffic is very dangerous as the information that is held in the packet can be read in plaintext.

Unencrypted Traffic

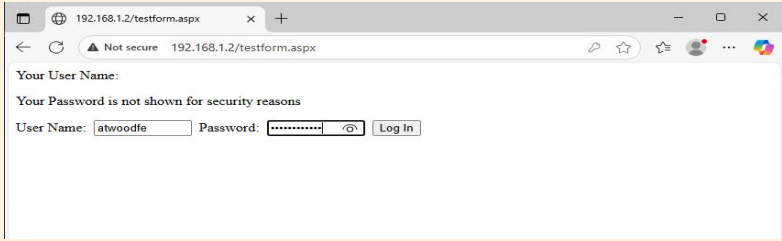
```
POST /testform.aspx HTTP/1.1
Host: 192.168.1.2
Connection: keep-alive
Content-Length: 45
Cache-Control: max-age=0
Origin: http://192.168.1.2
Content-Type: application/x-www-form-urlencoded
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/140.0.0.0 Safari/537.36 Edg/140.0.0.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/we
bp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: http://192.168.1.2/testform.aspx
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9

UserName=atwoodfe&UserPassword=123password%21
```

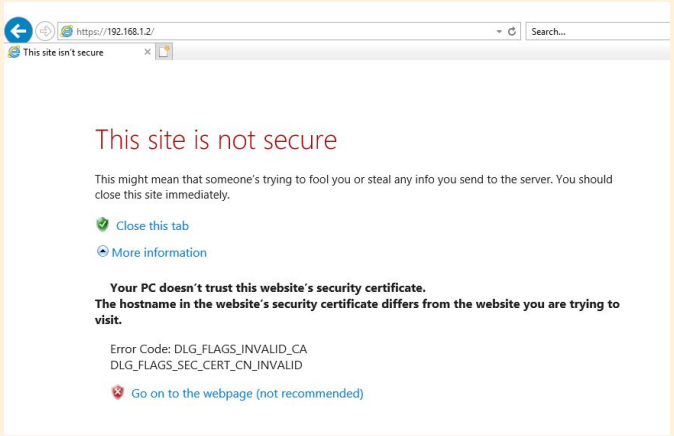
Encrypted Traffic



What does the difference look like to the user?

The difference to the user isn't large, if the user is to access a website that isn't encrypted they're going to see in the top left on the url bar it'll say not secure. Nothing more nothing less, that's the only warning the user is going to get that the website might not be safe to enter information on. If the user attempts to access the site but its encrypted they might be prompted with a screen saying it's not safe to continue, and if they are to continue to the site it's at their own risk and they'll see on the url bar saying it's not secure.



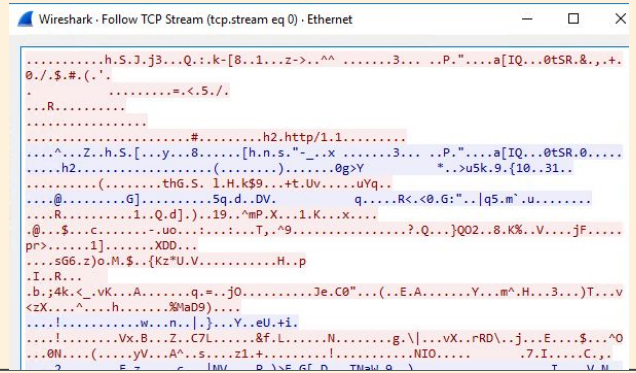


What does it look like from the network traffic analyst's perspective?

To an analyst they're going to be using Wireshark so the aren't going to see the same things as the user. In Wireshark they're going to see two different protocols, HTTP and HTTPS/TLS. The HTTP traffic they're going to see is going to allow them to reconstruct the entire conversation that was had between the two computers, whole files, credentials and URLs that were accessed. For the HTTPS/TLS traffic they're only going to be able to see encrypted traffic, random bytes of characters that aren't going to have any useful information in plaintext like the HTTP traffic would.

HTTP Traffic

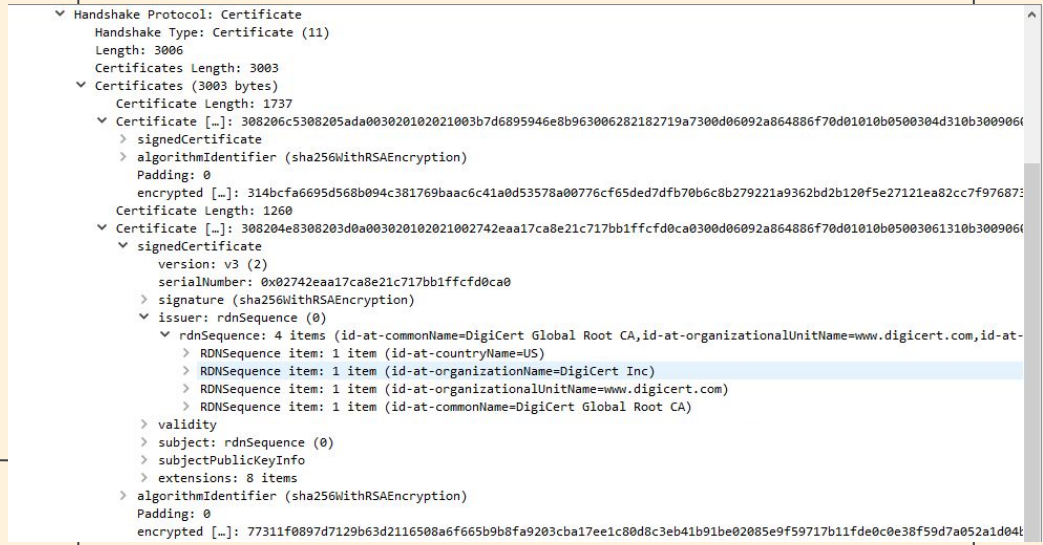
```
POST /testform.aspx HTTP/1.1
Host: 192.168.1.2
Connection: keep-alive
Content-Length: 45
Cache-Control: max-age=0
Origin: http://192.168.1.2
Content-Type: application/x-www-form-urlencoded
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/140.0.0.0 Safari/537.36 Edg/140.0.0.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/we
bp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Referer: http://192.168.1.2/testform.aspx
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9

UserName=atwoodfe&UserPassword=123password%21
```



HTTPS/TLS Traffic

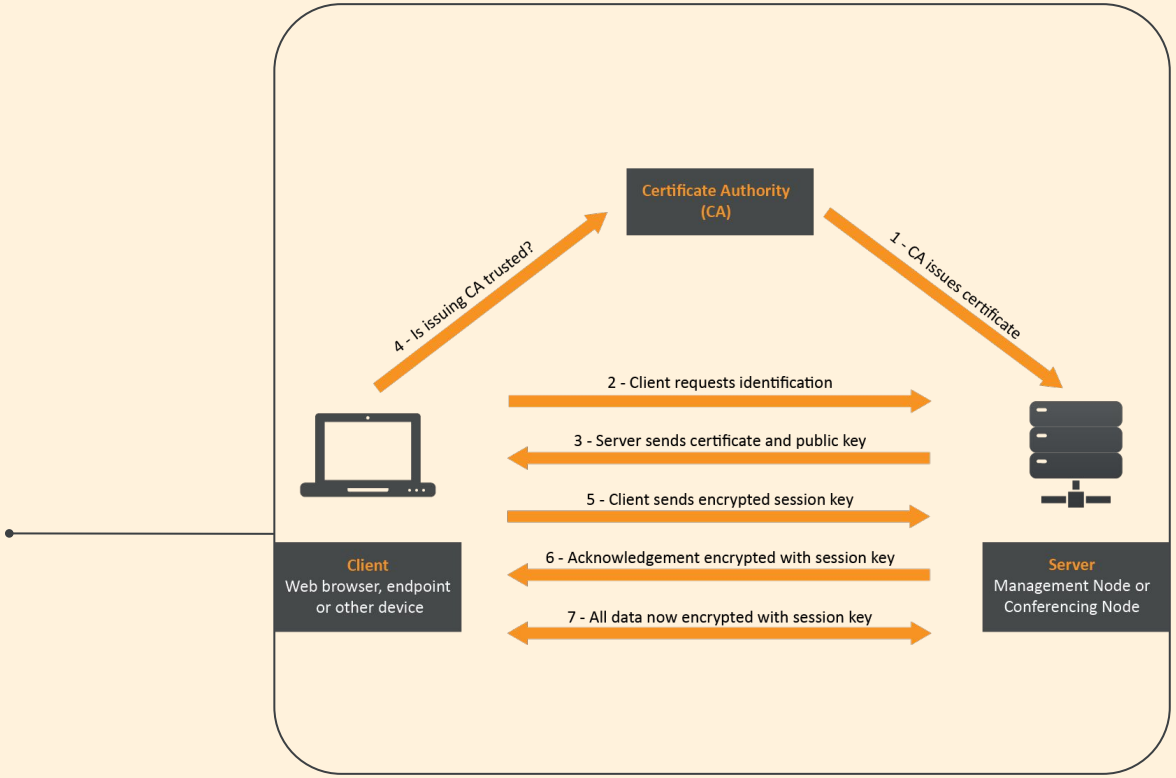
What is contained on a TLS certificate

There is a lot of information inside a TLS certificate, for example to the right is an example of a default windows certificate. The things you can find inside a certificate are; the organization name of the issuer, the public key, serial number, and the country name of the issuing organization. From the image on the right the organization is DigiCert, the public key is 308204…, the serial number is 0x02742…, and the country name is the US. There is a lot of information inside the certificate, these are just a couple example of what there is inside a default windows certificate.



What is the process for getting a TLS certificate?

The process of getting a TLS certificate signed by a Certificate Authority are as follows, generate a public and a private key, create a certificate signing request, with the public key, submit that CSR to a trusted CA. Then the certificate authority will validate the ownership of the domain that you're trying to get the certificate placed on. After its verified that you own the domain, the CA will sign the certificate with its private key, then they'll issue the certificate. Lastly the newly signed certificate will be assigned on the server. Once the certificate is vetted by the CA the website will be open for everyone to safely visit it.

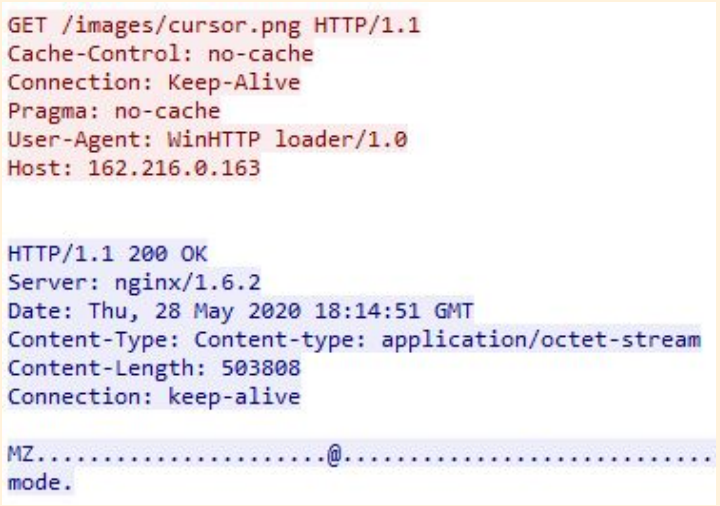


How is a self-signed certificate different from a certificate authority signed certificate?

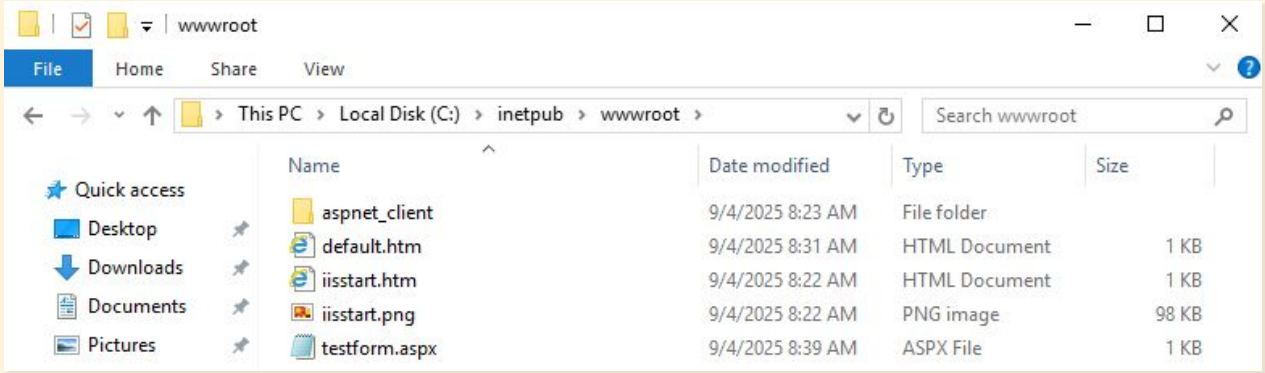
A self signed certificate is different from a CA signed certificate because the self signed certificate is in the name, you sign it yourself. The main difference between the two is that if you’re self signing a certificate for your own web browser, it might be secure and is a real and legitimate certificate there isn’t a third party vetting it like there would be if it was certified by a CA. That’s the only difference, a CA certificate is vetted by a third-party that is globally recognized as a CA, so therefore it is vetted and trusted. DigiCert and GlobalSign are two big companies who certify certificates for web browsers, and they will automatically be trusted as they are globally recognized for what they do.

Demonstrate and comment on the use of encrypted transmission to conceal malware infections and attacks

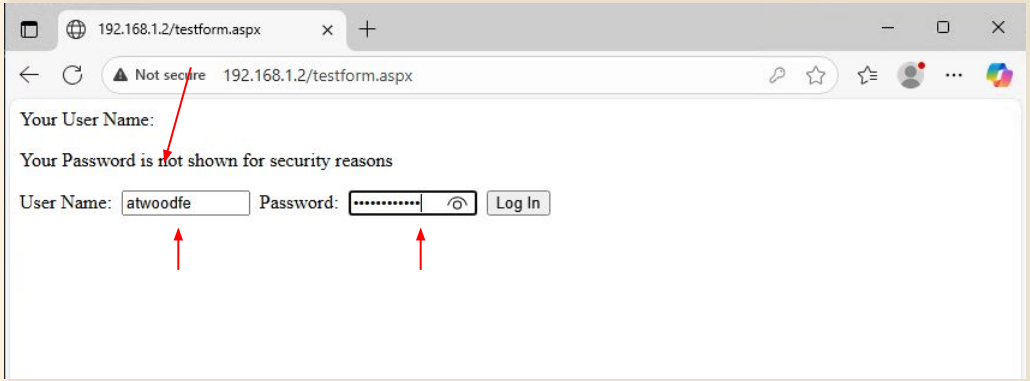
As we learned from working on our last project, hackers can conceal an .exe file as a .png for example. Misleading the user into thinking a file is something else when in reality it is malware and is dangerous. When inspecting the packets in Wireshark the files are tagged with “MZ”, which is the windows symbol for a .exe file. For example, in a secured network that’s protected by a firewall malicious software can be hidden in a zipped file, which hides it from the firewall when it scans the packets going to the network, leading to a scenario like above where someone downloads a .png, thinking it’s safe when it’s really malware.



This is an example from our last project, the MZ tag at the bottom means this is a .exe file that is being hidden within the cursor.png file at the top. This is something that can be missed by a firewall when downloaded as a zip file



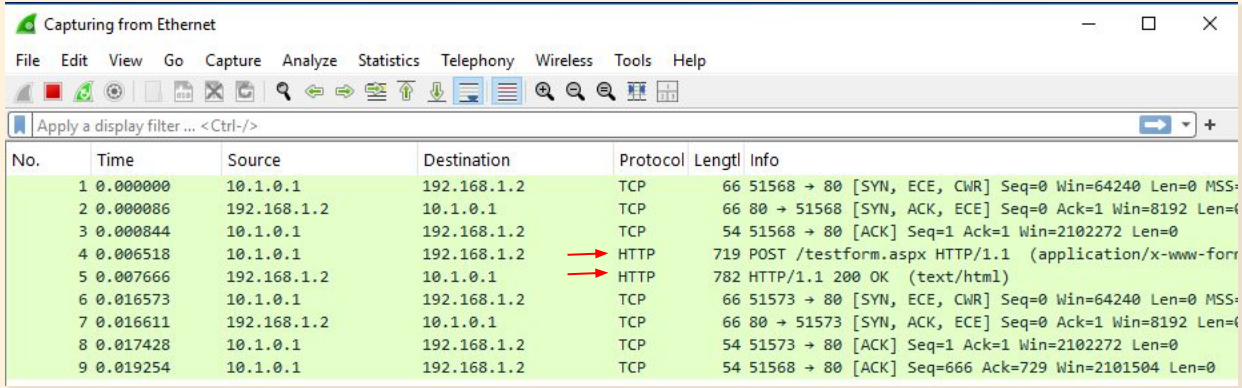
We start by creating our web server, this is what that looks like in the file explorer. The testform.aspx file is the scripts that we downloaded to prompt for a username and password. In order to get access to the server we need to connect to the ip “192.168.1.2/testform.aspx”

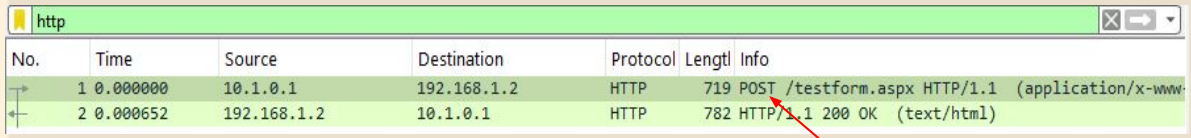


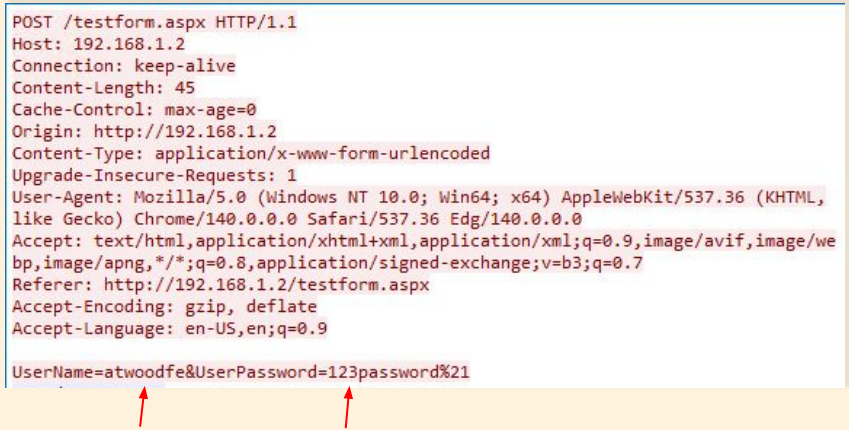
Here we’re prompted to enter a username and a password. To the user, we can’t see the password and we are told that the password is hidden for security reasons. We will see later that that is not true.

Being on a website that isn’t secure, we aren’t given a very bold warning about the site, other than a “not secure” warning in the top left of the webpage.

Here we enter our username “atwoodfe” and password that’s hidden for now







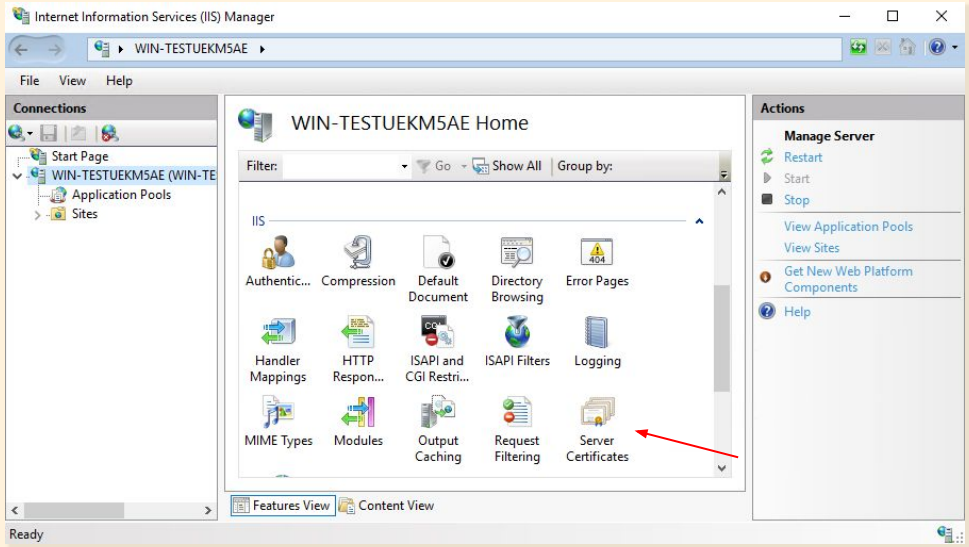
With wireshark open, we were watching the network traffic when suddenly we were given packets from a http query. We see two http requests here.

We know that the http packet that is tagged with the “POST” tag will have the information that was entered into the website lets see what information we can find from that packet.

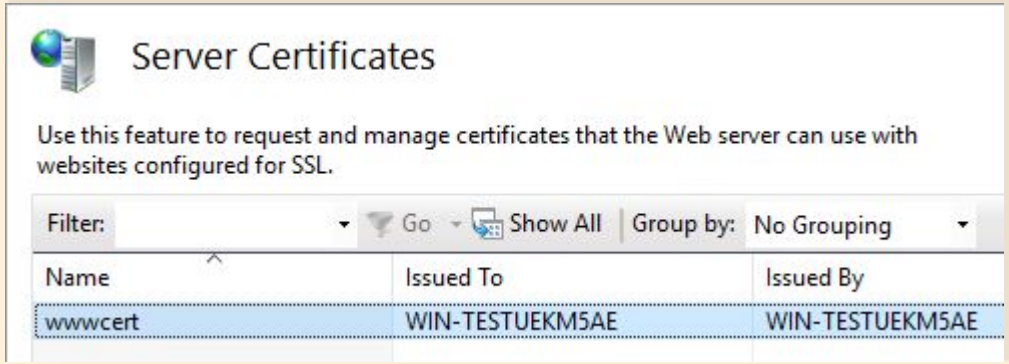
After following the TCP Stream from that packet this is what we see.

Now at the bottom in clean plaintext, we can see what was entered as our username and password.

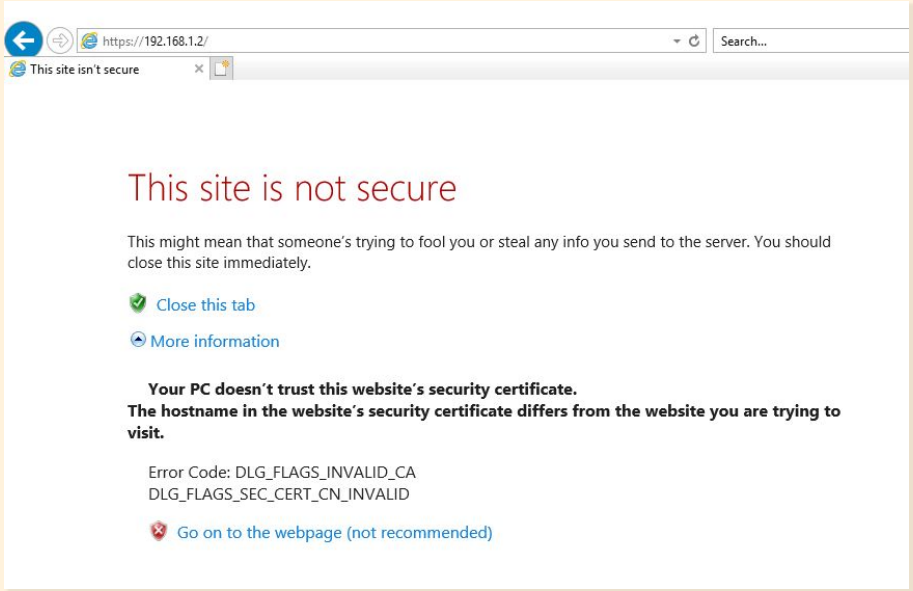
Our username “atwoodfe” and password “123password” are revealed in plaintext for us to see.



We now need to add our own self signed certificate to our web server. To do this we open the IIS manager in Windows Server Manager, and navigate to the Server Certificates icon.



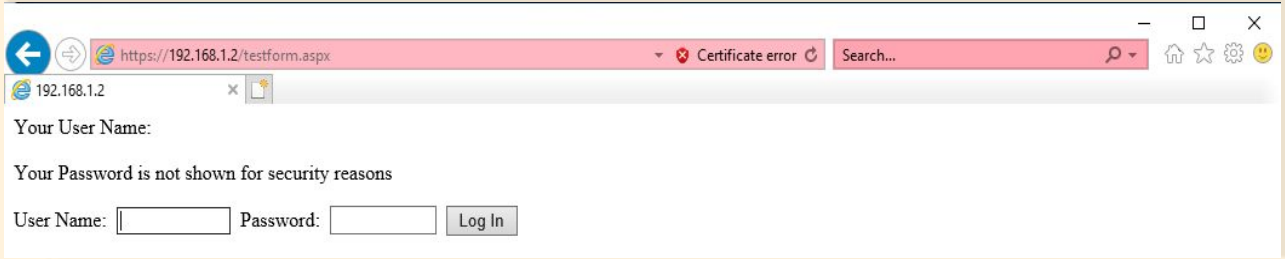
Now that we are there, we create our own self signed certificate, we call it wwwcert. As you can see the Windows Server that it is issued to, and issued by are the same, hence the name “self signed certificate”.



Now that we have added our own certificate to our web server, lets try it out and see what happens.

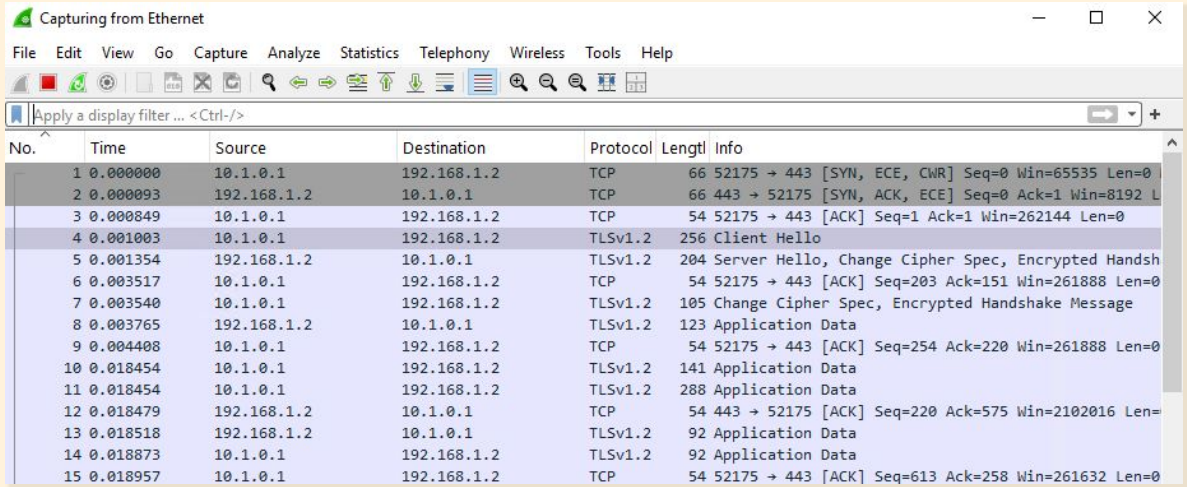
We'll use Internet Explorer for this part, and when we connect to our secured web server "https://192.168.1.2/testform.aspx" we see this page. "This site is not secure" and we see the reason is because the certificate is invalid

After we secured the web server with our self signed certificate we see this error that we didn't see before we secured it, very weird.



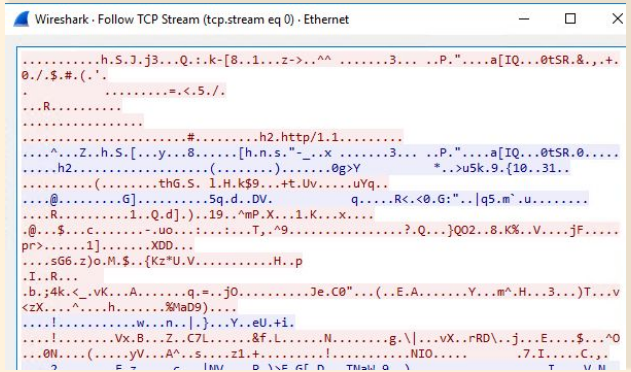
We continue to the page, and we can see at the top it's in red, warning us of the certificate error. Very weird we're getting warnings now vs. before we secured it with a certificate.

I'll enter the same username and password as before

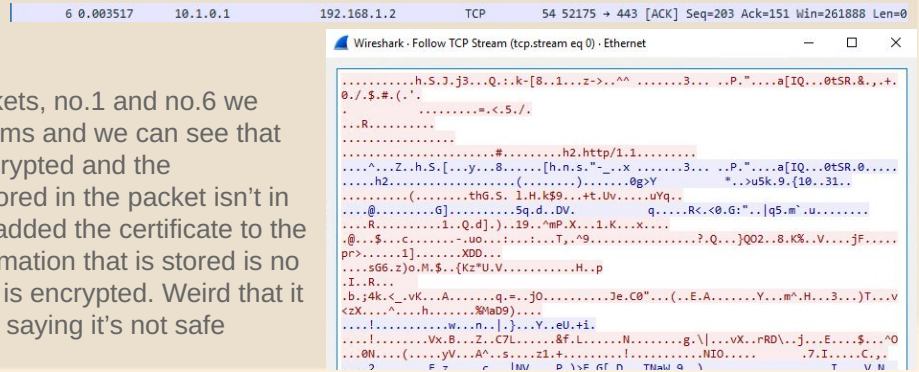


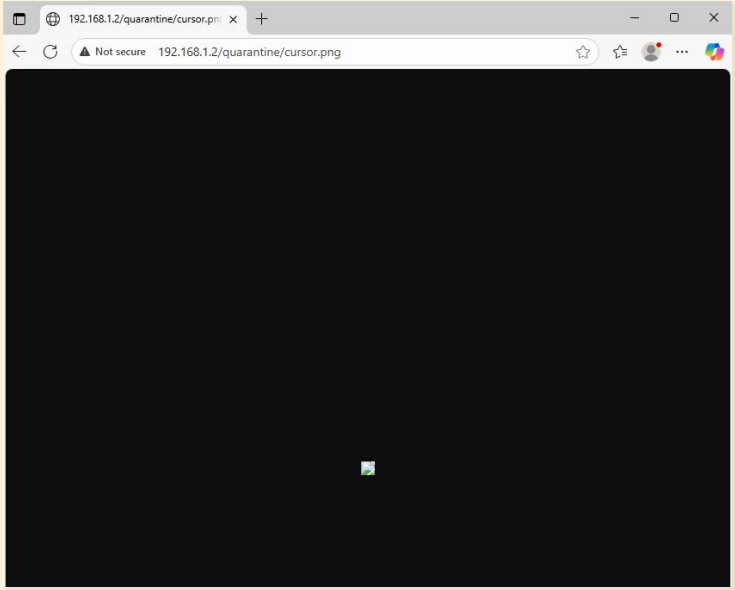
We're back analyzing our network again in wireshark, and after submitting the username and password on our new secured web server, these are the packets we see.

Lets pick a couple packets out and analyze them to see what we have here.



From these two packets, no.1 and no.6 we follow the TCP Streams and we can see that both of them are encrypted and the information that is stored in the packet isn't in plain text. Since we added the certificate to the web server, the information that is stored is no longer plain text and is encrypted. Weird that it gives us the warning saying it's not safe





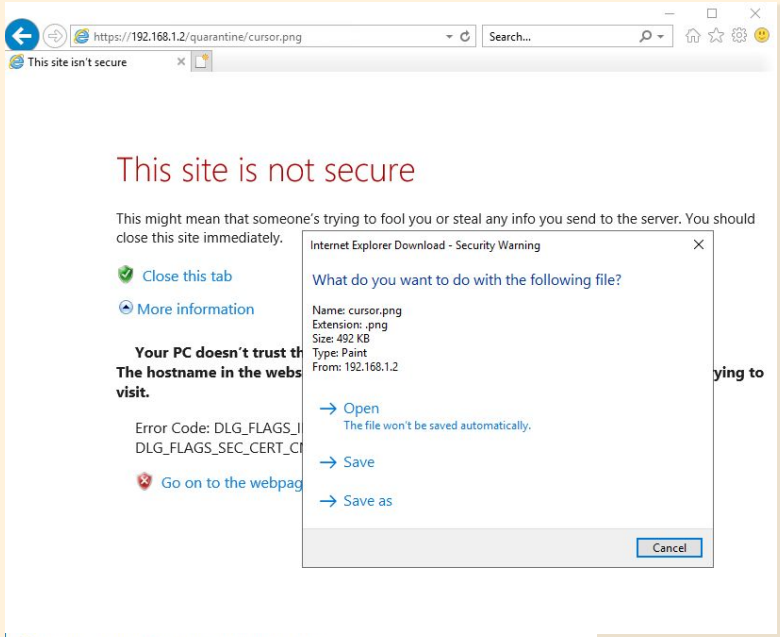
We have added a virus from our last exercise (cursor.png) to this web server and attempted to access it. Now this was just a .png file and not a shortcut or .exe file that would've activated or downloaded when it was accessed but, we can see that if this wasn't a .png file without any warning the server would've let us access the virus file, and activate or download it.

Below is the packets from Wireshark that were sent when it was accessed and to the right is the TCP Stream from packet no. 8

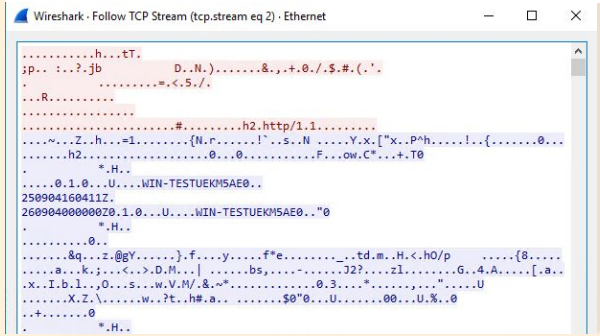
No.	Time	Source	Destination	Protocol	Lengtl	Info
1	0.000000	192.168.1.2	104.208.203.88	TLSv1.2	127	Application Data
2	0.006748	104.208.203.88	192.168.1.2	TLSv1.2	228	Application Data
3	0.053420	104.208.203.88	192.168.1.2	TCP	228	[TCP Retransmission] 443 → 49674 [PSH, ACK] Seq=1 Ack=74 Win=734…
4	0.053452	192.168.1.2	104.208.203.88	TCP	66	49674 → 443 [ACK] Seq=74 Ack=175 Win=1026 Len=0 SLE=1 SRE=175
5	0.680109	10.1.0.1	192.168.1.2	TCP	66	50714 → 80 [SYN, ECE, CWR] Seq=0 Win=64240 Len=0 MSS=1460 WS=256…
6	0.680216	192.168.1.2	10.1.0.1	TCP	66	80 → 50714 [SYN, ACK, ECE] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 W…
7	0.686557	10.1.0.1	192.168.1.2	TCP	54	50714 → 80 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
8	0.724008	10.1.0.1	192.168.1.2	HTTP	600	GET /quarantine/cursor.png HTTP/1.1
9	0.733490	10.1.0.1	192.168.1.2	TCP	66	50718 → 80 [SYN, ECE, CWR] Seq=0 Win=64240 Len=0 MSS=1460 WS=256…
10	0.733546	192.168.1.2	10.1.0.1	TCP	66	80 → 50718 [SYN, ACK, ECE] Seq=0 Ack=1 Win=8192 Len=0 MSS=1460 W…
11	0.737499	10.1.0.1	192.168.1.2	TCP	54	50718 → 80 [ACK] Seq=1 Ack=1 Win=2102272 Len=0
12	0.740838	192.168.1.2	10.1.0.1	HTTP	219	HTTP/1.1 304 Not Modified
13	0.756903	10.1.0.1	192.168.1.2	TCP	54	50714 → 80 [ACK] Seq=547 Ack=166 Win=2102016 Len=0

Wireshark · Follow HTTP Stream (tcp.stream eq 1) · Ethernet

```
GET /quarantine/cursor.png HTTP/1.1
Host: 192.168.1.2
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, li
ke Gecko) Chrome/140.0.0.0 Safari/537.36 Edg/140.0.0.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp
,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
If-None-Match: "1d54efe6622dc1:0"
If-Modified-Since: Wed, 10 Sep 2025 15:24:25 GMT


HTTP/1.1 304 Not Modified
Accept-Ranges: bytes
ETag: "1d54efe6622dc1:0"
Server: Microsoft-IIS/10.0
X-Powered-By: ASP.NET
Date: Wed, 10 Sep 2025 15:45:28 GMT
```



Now we accessed the same virus but this time on a secured server. The server didn't let us access the image, but instead told us that it isn't secure and prompted us with a popup to download the .png file cursor.png. Unlike the unsecured search where it just opened the file without warning this time we're prompted with a warning and told that this might not be safe to download or to open.



Here's what the query looked like on wireshark this time, a lot more packets were sent across compared to the unsecured search, and the TCP Streams were all encrypted unlike the HTTP packet where it was in plaintext.