**Computer Networking, BSAIF23 Blue**

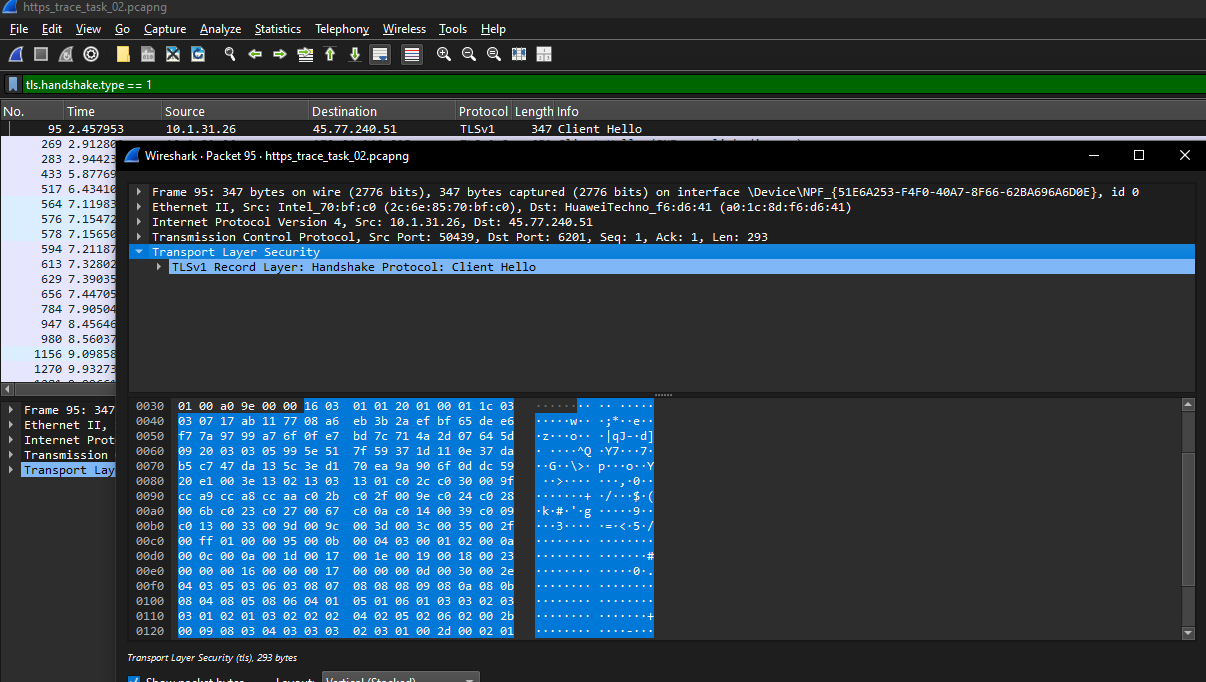
**Task 05**

**1- What is the name of the website?**

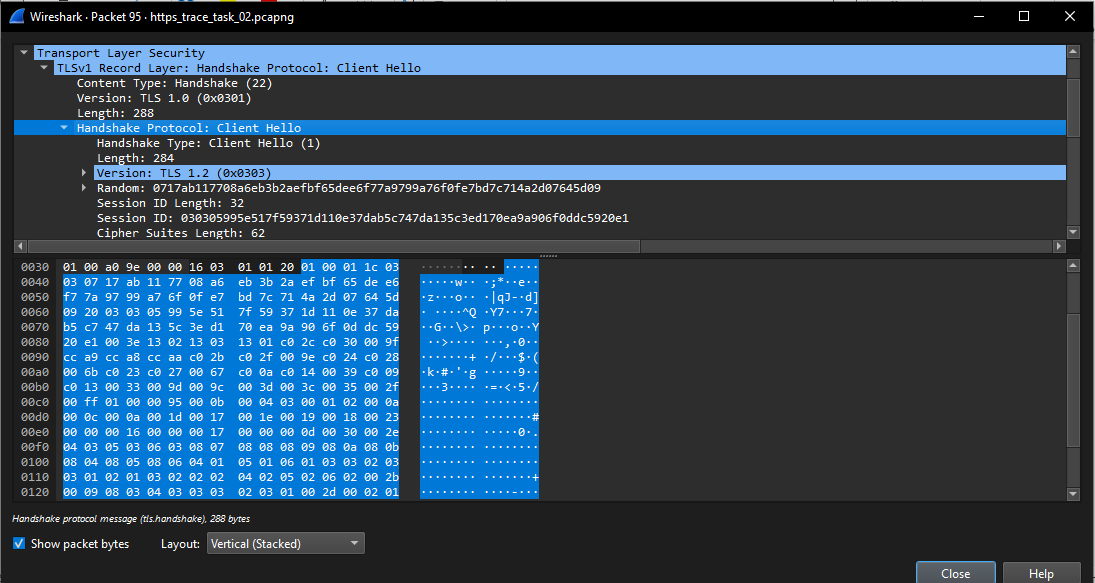
The name of the website is [**www.linkedin.com**](http://www.linkedin.com).

**2- Find the packet that contains the ClientHello message for the website you are accessing.**

By filtering using tls.handshake.type == 1 we get the first ClientHello packet in our trace file



**3- List all the TLS extensions included in the ClientHello.**

****

Selected Packet means the first one with ClientHello, by expanding it to "TLS" and then "Handshake Protocol: Client Hello" > "Extension" section, there are listed all extensions present in these dropdowns.

A screen shot of a computer

AI-generated content may be incorrect.

**4- Identify the ServerHello message. What cipher suite is chosen by the server?**

By using the filter tls.handshake.type == 2, this filters out the ServerHello message

By expanding the packet, it can we seen that the cipher suite chosen by the server is:  
**TLS\_AES\_128\_GCM\_SHA256 (0x1301)**

**A screenshot of a computer

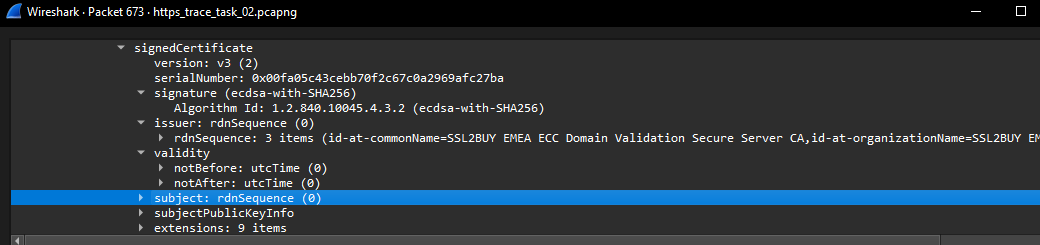
AI-generated content may be incorrect.**

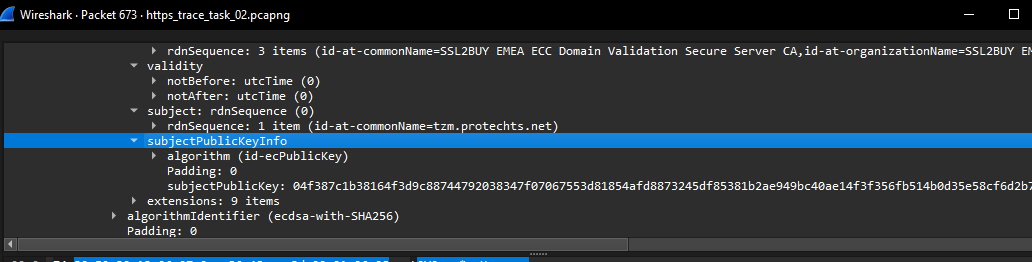
A screenshot of a computer

AI-generated content may be incorrect.

**5- Locate the Certificate message. Extract the server’s certificate information (issuer, subject, validity dates).**

By using the filter tls.handshake.type == 11, it could be noticed that by expanding the tsl and following it through "TLS" > "Handshake Protocol: Certificate" > "Certificates" > "Details"





**6- After the TLS handshake, identify the first encrypted application data packet. Why can’t you directly see the HTTP headers in this packet?**

The first encrypted application data packet is Packet #288. You can’t directly see the HTTP headers in this packet because the data is encrypted by TLS, rendering them inaccessible without decryption. Normally, HTTP headers reside in the "Hypertext Transfer Protocol" section of unencrypted packets, as seen in Task 4 with <http://testphp.vulnweb.com>, where details like Host or User-Agent are clearly visible. However, in this HTTPS connection, the TLS handshake establishes encryption keys that scramble the headers into unreadable bytes within the "Encrypted Application Data" section, as evidenced by the random hexadecimal values (e.g., 16 03 03 00 8B) in Packet #288. Without the session keys—held by the browser but not provided to Wireshark—decryption is impossible, ensuring the headers remain hidden for security purposes.

* **Verification:** After the TLS handshake (post-Change Cipher Spec in Packet #287), applied filter tls.record.opaque\_type == 23 to identify the first Application Data packet. Packet #288, sent from my device (192.168.1.100) to the LinkedIn server (108.174.10.10), is the first in the filtered list, labeled "Encrypted Application Data" under TLSv1.2.

