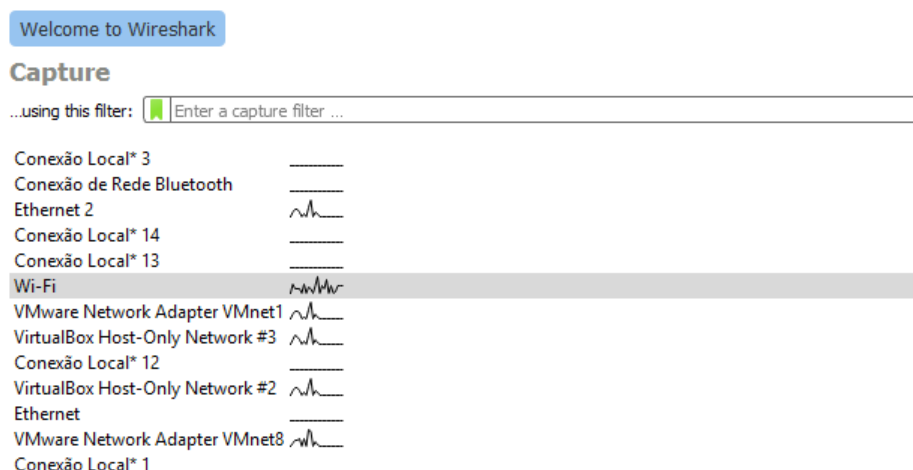


Quando estabelecemos uma sessão de forma segura com o HTTPS o servidor envia para o cliente uma chave pública, dentro de um certificado digital, a qual será utilizada posteriormente pelo cliente para que possa criptografar os dados que só poderão ser descriptografados pela chave privada que está no servidor.

Vamos acessar uma aplicação que trabalha com HTTPS para que possamos ver esse fluxo, primeiramente inicialize o Wireshark no seu computador local e escolha para o wireshark mostrar os resultados respectivos ao adaptador de rede que está sendo utilizado para conexão na internet, por exemplo, se seu computador estiver utilizando wi-fi para se conectar na internet, utiliza o adaptador wireless correspondente, se estiver utilizando conexão cabeada, utilize o adaptador respectivo.



Para facilitar o filtro posteriormente no Wireshark, feche todas as abas no browser e abra somente uma jabela para acessar a aplicação original da Alura www.alura.com.br

Ao acessarmos a aplicação da Alura, devemos ter capturado no Wireshark todos os pacotes referentes a comunicação entre nosso computador e o servidor com a aplicação da Alura. Para filtrarmos esses dados de comunicação, vamos utilizar a ferramenta administrativa do nslookup para sabermos qual endereço IP que responde pelo domínio **alura.com.br**

```
C:\Users\Rafael>nslookup www.alura.com.br
Servidor:  dlinkrouter
Address:  192.168.65.1

Não é resposta autoritativa:
Nome:      ghs.googlehosted.com
Addresses: 2800:3f0:4001:80d::2013
           216.58.202.115
Aliases:   www.alura.com.br
```

Copie o endereço IP e coloque no Wireshark o seguinte fitro:

ip.addr==[Endereço IP obtido com nslookup]

COPIAR CÓDIGO

Procure pelo pacote **Certificate** para ver o certificado digital passado do servidor para o cliente:

No.	Time	Source	Destination	Protocol	Length	Info
1149	10.647131	192.168.65.223	216.58.202.115	TCP	66	13166 → 443 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SAC...
1168	10.685810	216.58.202.115	192.168.65.223	TCP	66	443 → 13166 [SYN, ACK] Seq=0 Ack=1 Win=42780 Len=0 MSS=1380...
1177	10.686216	192.168.65.223	216.58.202.115	TCP	54	13166 → 443 [ACK] Seq=1 Ack=1 Win=66048 Len=0
1185	10.691671	192.168.65.223	216.58.202.115	TLSv1.2	258	Client Hello
1219	10.947997	192.168.65.223	216.58.202.115	TCP	258	[TCP Retransmission] 13166 → 443 [PSH, ACK] Seq=1 Ack=1 Win...
1229	10.995856	216.58.202.115	192.168.65.223	TCP	54	443 → 13166 [ACK] Seq=1 Ack=205 Win=44832 Len=0
1268	11.066148	216.58.202.115	192.168.65.223	TLSv1.2	1484	Server Hello
1261	11.066148	216.58.202.115	192.168.65.223	TCP	1484	[TCP segment of a reassembled PDU]
1262	11.066149	216.58.202.115	192.168.65.223	TCP	1290	[TCP segment of a reassembled PDU]
1263	11.066149	216.58.202.115	192.168.65.223	TLSv1.2	1484	Certificate TCP segment of a reassembled PDU]
1264	11.066150	216.58.202.115	192.168.65.223	TLSv1.2	362	Server Key Exchange, Server Hello Done

Na sequência, na parte inferior do Wireshark, temos os detalhes referentes a esse pacote, dentro da aba **Secure sockets layer** procure pelo certificado referente ao domínio da Alura. Qual o resultado? Você consegue encontrar a chave pública dentro do certificado digital?

Opinião do instrutor

Ao analisarmos o resultado referente a esse pacote, nós temos justamente os dados do certificado digital com a chave pública passada para o nosso cliente, com isso nosso browser sabe que de fato estamos acessando a aplicação verdadeira da Alura, fornecendo assim uma segurança para o usuário final.

```

  ▾ Handshake Protocol: Certificate
    Handshake Type: Certificate (11)
    Length: 5406
    Certificates Length: 5403
    ▾ Certificates (5403 bytes)
      Certificate Length: 1361
      ▾ Certificate: 3082054d30820435a00302010202104f2d77ce440b4ed7f6... (id-at-commonName=www.alura
        ▾ signedCertificate
          version: v3 (2)
          serialNumber: 0x4f2d77ce440b4ed7f64221f0710f9c94
          > signature (sha256WithRSAEncryption)
          > issuer: rdnSequence (0)
          > validity
          > subject: rdnSequence (0)
          ▾ subjectPublicKeyInfo
            > algorithm (rsaEncryption)
            > subjectPublicKey: 3082010a0282010100bb8e7988241646dc3fa29ce2a270d2...
            > extensions: 9 items
          > algorithmIdentifier (sha256WithRSAEncryption)

```