计算机学院 计算机网络 课程实验报告

实验题目： TLS		学号：202200130048
日期：5.28	班级： 6	姓名： 陈静雯
Email：1205037094@qq.com		

实验方法介绍：

通过分析在通过 HTTPS 检索网页期间捕获的 Wireshark 数据包跟踪来研究 TLS。

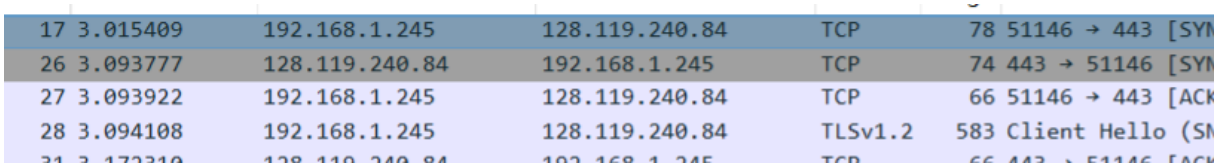
实验过程描述：

1. 捕获 TLS 会话中的报文
2. 首先看一下捕获的跟踪
3. TLS 握手:客户端你好消息
4. TLS 握手:结束握手
5. 应用程序数据

结论分析：

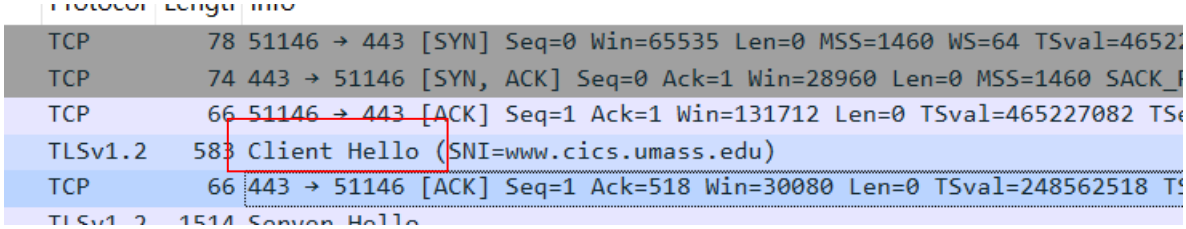
1. What is the packet number in your trace that contains the initial TCP SYN message?

Packet number：17



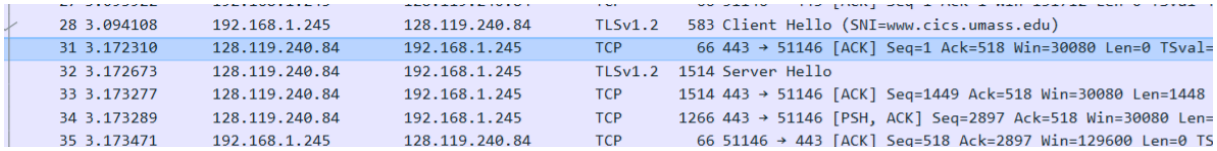
2. Is the TCP connection set up before or after the first TLS message is sent from client to server?

在 TLS 之前



3. What is the packet number in your trace that contains the TLS Client Hello message?

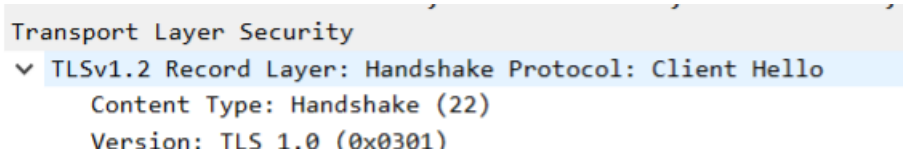
Packet number：28



4. What version of TLS is your client running, as declared in the Client Hello

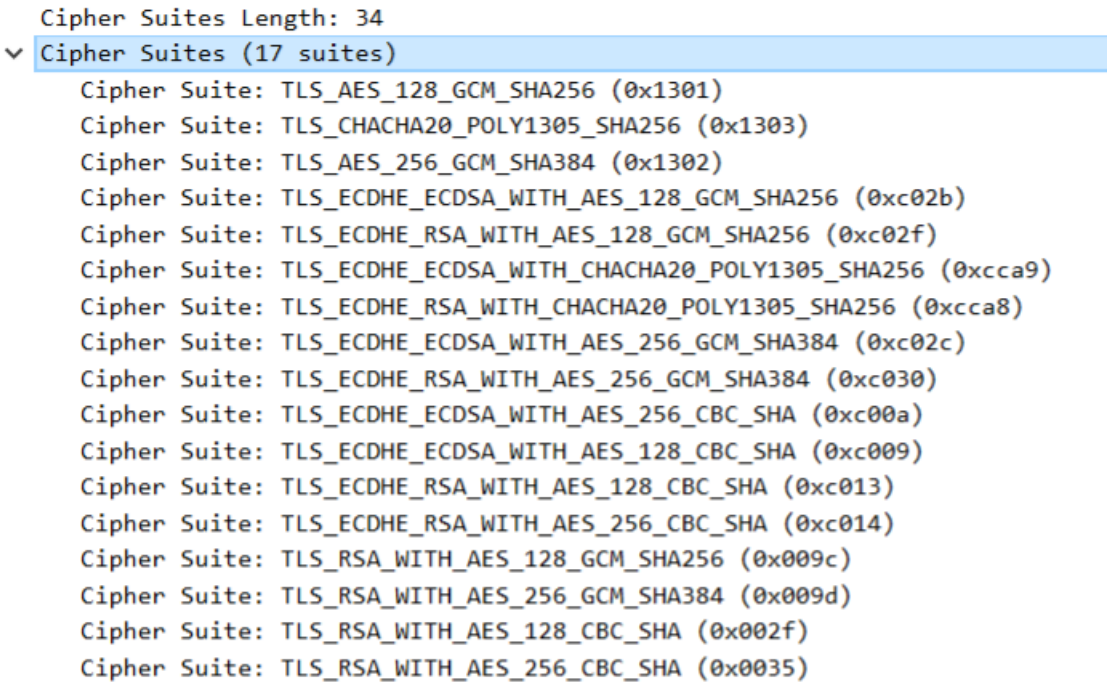
message?

TLSv1.2



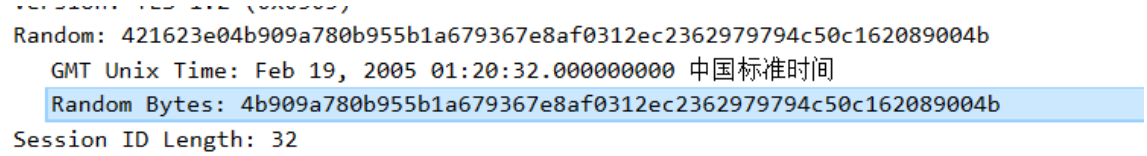
5. How many cipher suites are supported by your client, as declared in the Client Hello message?

17



6. What are the first two hexadecimal digits in the random bytes field of the Client Hello message?

4b



7. What is the purpose(s) of the "random bytes" field in the Client Hello message?

（1）唯一性：每次 TLS 握手时，客户端都会生成一个新的随机数。这个随机数与服务器随后在"Server Hello"消息中提供的随机数一起，用于生成会话的唯一密钥材料。这有助于确保即使对于相同的客户端和服务器，每次连接的加密密钥也是不同的，增加了安全性。

（2）密钥衍生：这些随机字节与其他握手信息一起被输入到一个密钥导出函数中（如 HKDF 或 PRF），用于生成对话密钥（包括对称加密密钥、MAC 密钥和初始化向量等），这些密钥用于之后的通信加密和完整性校验。

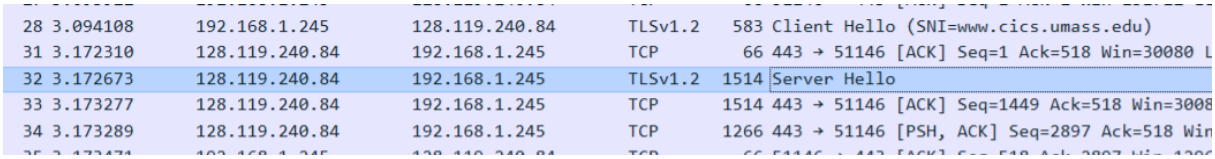
（3）防重放攻击：通过在每次握手时使用不同的随机数，可以防止旧的握手消息被截取并重新用于未来的连接尝试，这是一种基本的防重放保护机制。

（4）协议版本协商：虽然"随机字节"字段本身不直接用于协议版本的选择，但它作为

"Client Hello"消息的一部分，该消息整体上允许客户端声明它支持的 TLS 协议版本列表，从而促进了客户端与服务器之间就最安全的共同协议版本达成一致。

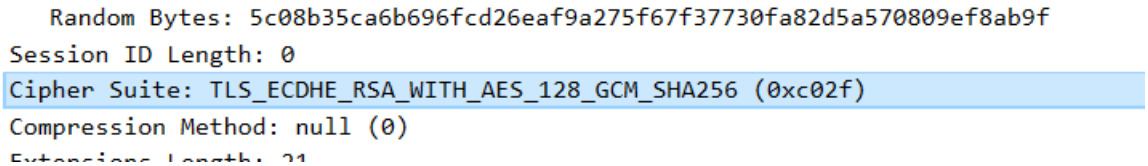
8. What is the packet number in your trace that contains the TLS Server Hello message?

Packet number：32



9. Which cipher suite has been chosen by the server from among those offered in the earlier Client Hello message?

Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)



10. Does the Server Hello message contain random bytes, similar to how the Client Hello message contained random bytes? And if so, what is/are their purpose(s)?

YES。"Server Hello"中的随机字节不仅是密钥派生过程的关键组件，也是确保每次 TLS 握手独特性和完整性的基础。

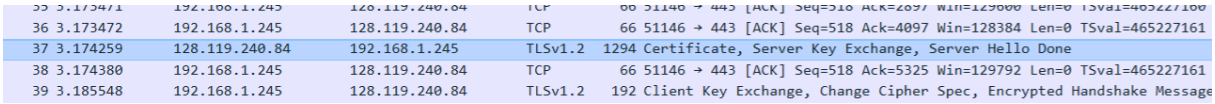
（1）密钥衍生的贡献：服务器生成的这个随机数与客户端提供的随机数结合，共同作为密钥导出函数的输入。这两个随机数的组合增加了生成的密钥材料的不可预测性，进一步强化了加密的安全性。

（2）防重放和会话标识：服务器的随机数与客户端的随机数一起，帮助确保每个新的 TLS 会话都是唯一的，这对于防止重放攻击至关重要。即使攻击者截获了一个有效的握手过程，由于缺少服务器特定的随机数，他们无法利用这些信息来冒充合法的服务器或重放之前的通信。

（3）协商一致的确认：服务器通过发送其自己的随机数，确认了它已经收到了并处理了客户端的"Client Hello"消息，这是双方协商过程的一个重要部分。这有助于确保双方都同意了即将建立的会话的具体细节，包括加密套件和协议版本等。

11. What is the packet number in your trace for the TLS message part that contains the public key certificate for the www.cics.umass.edu server (actually the www.cs.umass.edu server)?

packet number：37



12. If more than one certificate is returned, are all of these certificates for www.cs.umass.edu? If not all are for www.cs.umass.edu, then who are these other certificates for? You can determine who the certificate is for by checking the id-at-commonName field in the retuned certificate.

不是所有都给 www.cs.umass.edu，

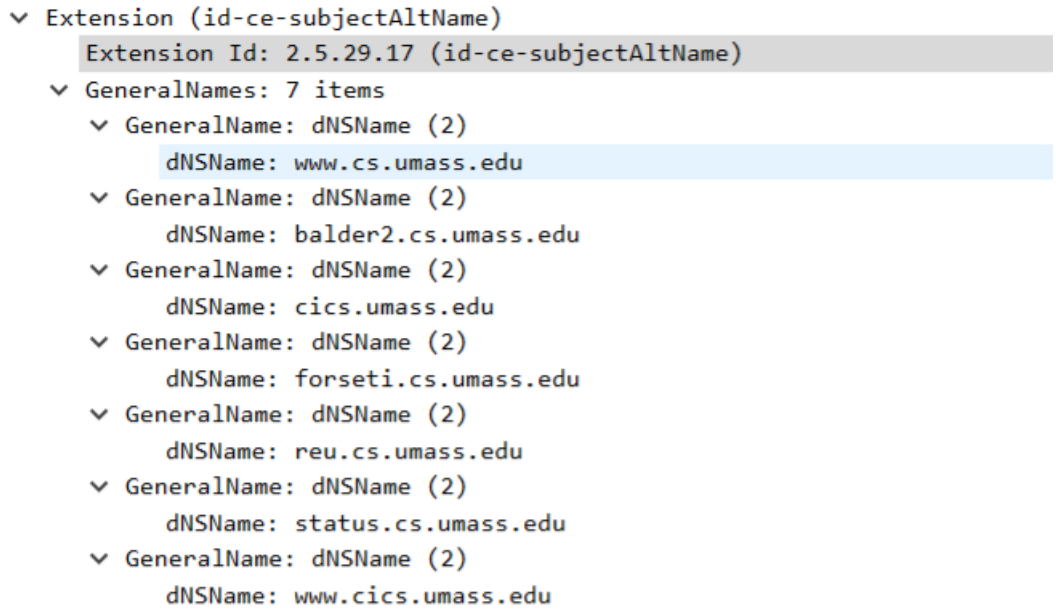
还有

balder2.cs.umass.edu ,cics.umass.edu,forseti.cs.umass.edu,reu.cs.umass.edu,

status.cs.umass.edu， www.cics.umass.edu

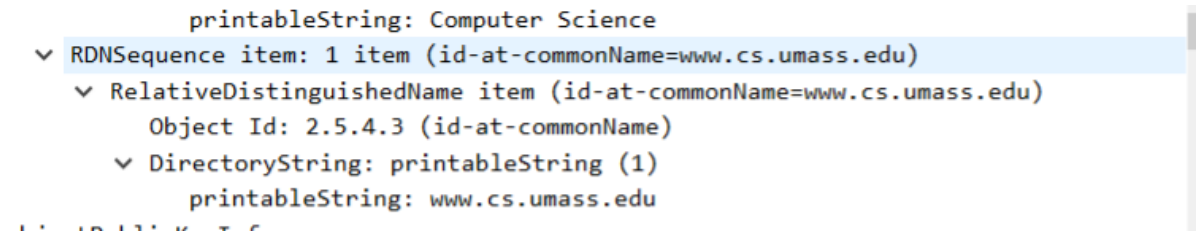
（1）证书链：最常见的情况是，服务器不仅仅发送服务器证书本身，还会发送一连串的中级 CA（证书颁发机构）证书直至根 CA 证书（或一个已知的根 CA 证书）。这样做是为了构建一条信任链，让客户端可以验证服务器证书的真实性。这些中级 CA 证书并不是直接为 www.cs.umass.edu 的，而是为了证明服务器证书是由一个可信的机构签发的。

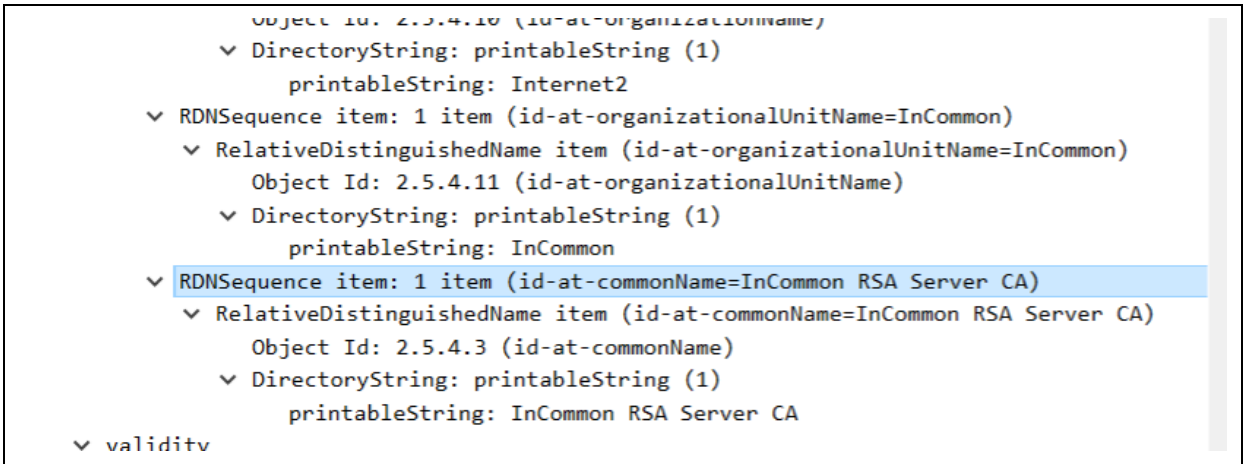
（2）多域名和通配符证书：有时候，一个证书可以被颁发给多个不同的域名或子域名（通过 subjectAltName 字段）。如果返回的多个证书中包含了不同的域名或子域名，它们可能都是为了支持同一个服务器或服务集群的不同访问点，而不一定全都是针对 www.cs.umass.edu。



13. What is the name of the certification authority that issued the certificate for id-at-commonName=www.cs.umass.edu?

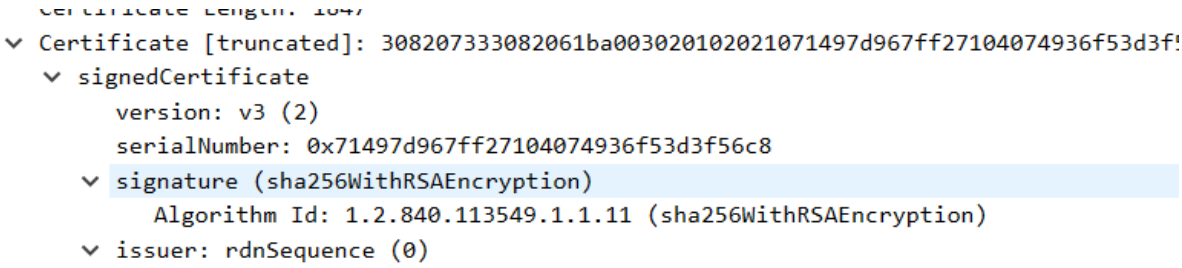
Issuer:(id-at-commonName=InCommon RSA Server CA, id-at-organizationalUnitName=InCommon, id-at-organizationName=Internet2, id-at-localityName=Ann Arbor, id-at-stateOrProvinceName=MI, id-at-countryName=US)





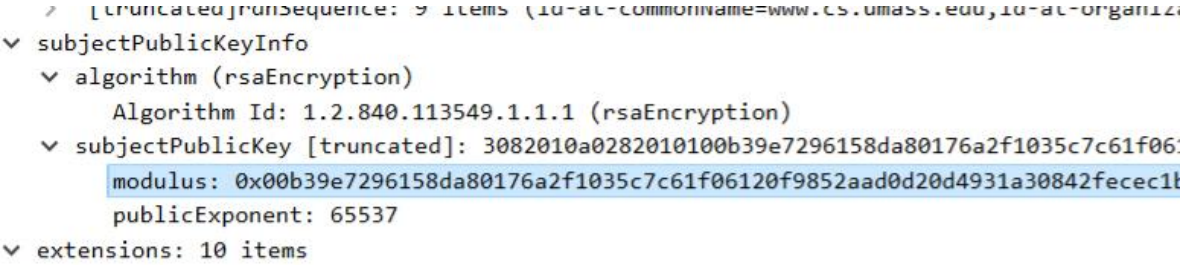
14. What digital signature algorithm is used by the CA to sign this certificate?

1.2.840.113549.1.1.11



15. What are the first four hexadecimal digits of the modulus of the public key being used by www.cics.umass.edu?

00b3



16. Do you see such message in your trace? If so, what is the number in the trace of the first packet sent from your client to the CA? If not, explain why the client did not contact the CA.

不会直接看到客户端与认证机构（CA）之间的消息来获取 CA 的公钥信息。这是因为 TLS 协议的设计并不直接要求客户端与 CA 进行通信来验证服务器提供的证书。

验证过程：

（1）服务器发送证书链：服务器在 TLS 握手的“Certificate”消息中发送其证书以及可能的证书链（包括中间证书和根证书的指纹或整个证书）。这些证书中包含了由上级 CA 签名的信息，一路追溯到根 CA。

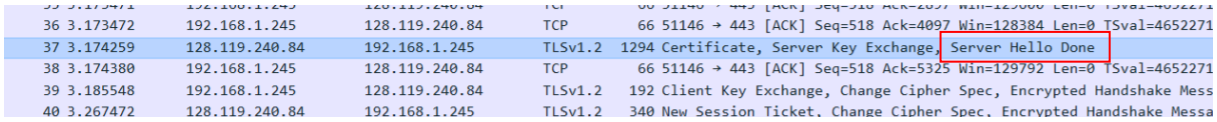
（2）客户端本地验证：客户端使用它本地信任的根证书存储（这些根证书通常由操作系统或浏览器预先安装）来验证服务器证书链。它会检查服务器证书是否由一个受信任的 CA 直接或间接签名，并且验证证书的有效期、撤销状态等。

所以，客户端验证服务器证书的有效性是通过检查本地已知的 CA 公钥完成的，无需单独向 CA 发起请求。

如果要查看这些证书和验证过程的细节，“Certificate”消息以及随后客户端如何使用这些信息进行证书链的验证，而不需要寻找客户端直接联系 CA 的网络包。因此，不会有从客户端到 CA 的第一包的编号，因为这样的交互不在标准 TLS 握手流程之内。

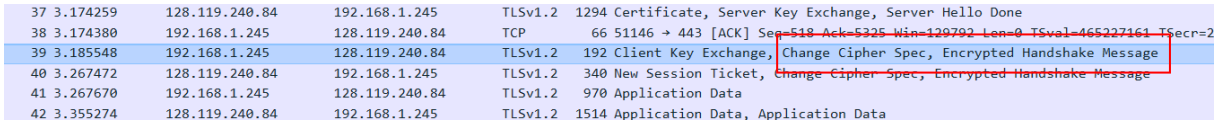
17. What is the packet number in your trace for the TLS message part that contains the Server Hello Done TLS record?

packet number：37



18. What is the packet number in your trace for the TLS message that contains the public key information, Change Cipher Spec, and Encrypted Handshake message, being sent from client to server?

packet number：39

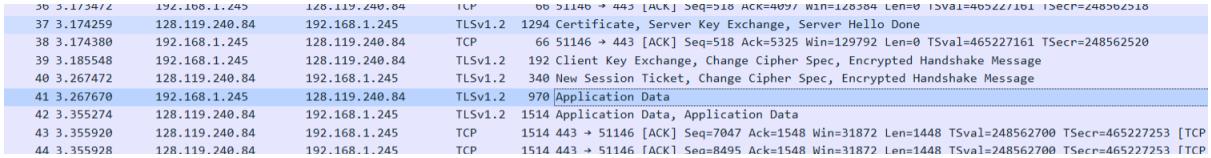


19. Does the client provide its own CA-signed public key certificate back to the server? If so, what is the packet number in your trace containing your client's certificate?

客户端没有像服务器提供自己的 CA（证书颁发机构）签署的公钥证书。

在一般的 TLS 握手过程中，客户端通常不会向服务器提供自己的 CA（证书颁发机构）签署的公钥证书，除非是特定的应用场景需求，比如（双向认证）。双向认证要求不仅服务器向客户端证明其身份，客户端也需要向服务器证明自己的身份。这种情况下，客户端会在握手过程中,通常紧跟在"Client Key Exchange"之后,发送一个"Certificate"消息给服务器，里面包含客户端的证书。

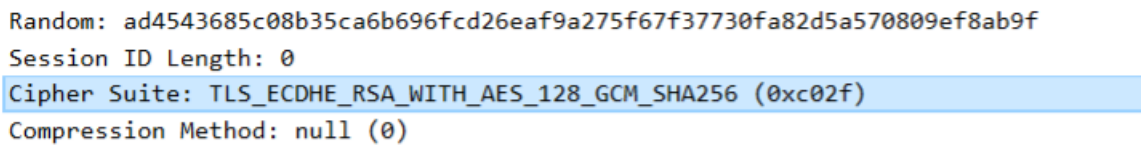
如果存在客户端证书的交换，您会在 TLS 握手序列中找到相应的"Certificate"消息，其具体包编号会根据您的捕获文件和实际网络交互的上下文而变化。如果没有进行双向认证，您就不会在抓包数据中看到客户端发送证书给服务器的记录。



20. What symmetric key cryptography algorithm is being used by the client and server to encrypt application data

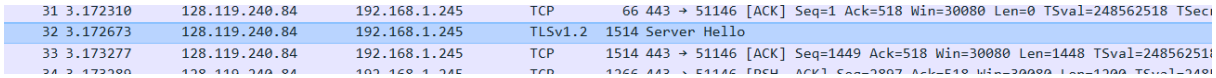
使用 AES-128-GCM 算法来加密和解密应用数据。

AES（Advanced Encryption Standard）的 128 位密钥长度，模式为 GCM（Galois/Counter Mode）。AES-GCM 是一种高性能的加密模式，它同时提供了数据加密和数据完整性校验（通过消息认证码 MAC），适合于高安全性且对性能有要求的场景。



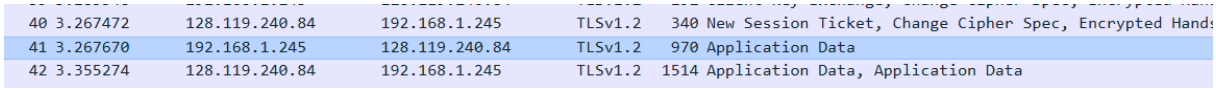
21. In which of the TLS messages is this symmetric key cryptography algorithm finally decided and declared?

服务器会从客户端提议的一系列密码套件（Cipher Suites）中选择一个支持的套件，并在“Server Hello”消息中确认。



22. What is the packet number in your trace for the first encrypted message carrying application data from client to server?

packet number：41



22. What do you think the content of this encrypted application-data is, given that this trace was generated by fetching the homepage of www.cics.umass.edu?

（1）HTML 页面内容：主页的 HTML 源代码，包括 HTML 标签、文本内容、CSS 样式链接、JavaScript 脚本引用等，这些都是构成网页基本结构和外观的数据。

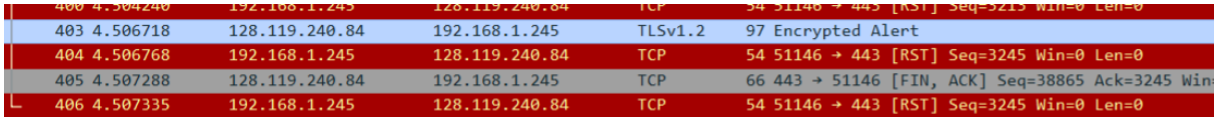
（2）图片、CSS 和 JavaScript 文件：网页通常还会请求加载外部资源，如图片文件（JPEG、PNG 等格式）、CSS 样式表和 JavaScript 脚本文件。这些资源也可能被封装在后续的 Application Data 包中，尽管它们可能各自通过不同的 HTTP/HTTPS 请求获取。

（3）cookies 和网站偏好设置：首次访问或每次访问时，服务器可能会发送或更新 cookie 给客户端，用于存储用户的会话状态、个性化设置等信息。

（4）HTTP 头信息：虽然 HTTP 头信息（如请求行、请求头、响应头）在 TLS 握手之后的明文形式下已经交换，但实际响应体内容（即 HTML 和其他资源数据）会被加密在 Application Data 中传输，以保护用户数据的隐私和安全。

24. What packet number contains the client-to-server TLS message that shuts down the TLS connection?

packet number：403



结论：

1. Client Hello 消息是客户端与服务器建立连接时握手阶段客户端发送的第一个消息。"随机字节"字段（Random Bytes）在"Client Hello"消息中扮演着几个关键角色（唯一性、密钥衍生、防重放攻击）是 TLS 协议安全性和握手过程灵活性的基础组成部分。
2. "Server Hello"消息也包含一个随机数（Random Bytes）字段，这个随机数的作用与"Client Hello"消息中的随机字节类似，但也有一些特定于服务器端的用途（密钥衍生的贡献、防重放和会话标识、协商一致的确认），因此，"Server Hello"中的随机字节不仅是密钥派生过程的关键组件，也是确保每次 TLS 握手独特性和完整性的基础。
3. 当服务器返回多个证书时，这些证书可能有以下几种情况：证书链、多域名和通配符证书、错误配置。
4. TLS 协议的设计并不直接要求客户端与 CA 进行通信来验证服务器提供的证书。验证

过程是这样进行的：

服务器发送证书链：服务器在 TLS 握手的“Certificate”消息中发送其证书以及可能的证书链（包括中间证书和根证书的指纹或整个证书）。这些证书中包含了由上级 CA 签名的信息，一路追溯到根 CA。

客户端本地验证：客户端使用它本地信任的根证书存储（这些根证书通常由操作系统或浏览器预先安装）来验证服务器证书链。它会检查服务器证书是否由一个受信任的 CA 直接或间接签名，并且验证证书的有效期、撤销状态等。

5. Server Key Exchange: 此消息主要用于某些密钥交换协议，如非预共享密钥（non-PFS）的密钥交换方法，如 DH（Diffie-Hellman）或 ECDH（Elliptic Curve Diffie-Hellman）。在这一步中，服务器会发送必要的参数（例如 DH 参数或 ECDHE 曲线和公钥）给客户端，使得客户端可以计算出共享密钥。如果使用了 RSA 密钥交换（RSA 密钥交换不常见于现代 TLS 连接，因为它不支持 PFS），这一步可能不会出现，因为共享密钥的计算方式不同。这个消息的目的是为了在客户端和服务器之间建立一个安全的密钥，用于后续通信的加密。
6. Server Hello Done: 这个消息标志着服务器端“Hello”阶段的结束。它是一个简单的指示器，告诉客户端服务器已经发送了所有初始握手消息，现在轮到客户端响应。这之后，客户端会验证服务器的证书，可能进行密钥交换（如果之前未完成），发送密钥确认（如"Client Key Exchange"消息），以及进行握手的最后步骤，如发送"Change Cipher Spec"和"Finished"消息。
7. Client Key Exchange: 在这一步骤中，客户端回应服务器的密钥交换信息（如果之前有 Server Key Exchange 消息）。对于基于密钥交换协议如 DH 或 ECDH，客户端会发送它的公钥或密钥交换的必要参数给服务器。如果使用 RSA 密钥交换，客户端则会使用服务器的公钥加密一个预主密钥（Premaster Secret），并将其发送给服务器。这一步是确保客户端和服务器双方能够独立计算出相同的主密钥，该主密钥随后用于生成会话密钥。
8. Change Cipher Spec: 这不是一个实际的数据包，而是一个特殊的消息，告知对方接下来的通信将使用之前协商好的加密参数进行加密。这是一个重要的转换点，意味着从这刻起，双方的通信内容都将被加密。
9. Encrypted Handshake Message / Finished: 在发送"Change Cipher Spec"之后，客户端紧接着发送一个"Finished"消息。这个消息包含了一个校验码（Message Authentication Code, MAC），它是之前握手消息的哈希值，并使用新协商的密钥进行了加密。这一步是验证密钥交换成功并确保握手过程中没有被篡改的最后验证。服务器也会发送自己的"Change Cipher Spec"和"Finished"消息作为响应，完成握手过程。
10. "Encrypted Alert" 消息是 TLS 协议中用于在已经建立的安全连接上传输警示信息的一种方式，其中包括了连接关闭（Close Notify）的警示。当您在 Wireshark 中看到标记为“Encrypted Alert”的数据包时，这表明有一方（客户端或服务器）正在通过加密通道发送一个警报来请求关闭 TLS 连接或者报告其他需要对方知晓的警示信息。