**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC BÁCH KHOA**

**KHOA KHOA HỌC VÀ KỸ THUẬT MÁY TÍNH**

----🙠🕮🙢----



**MẠNG MÁY TÍNH TN (CO3094)**

**LAB 8**

**Wireshark Lab: SSL v8.0**

**HK: 241 - LỚP: L09**

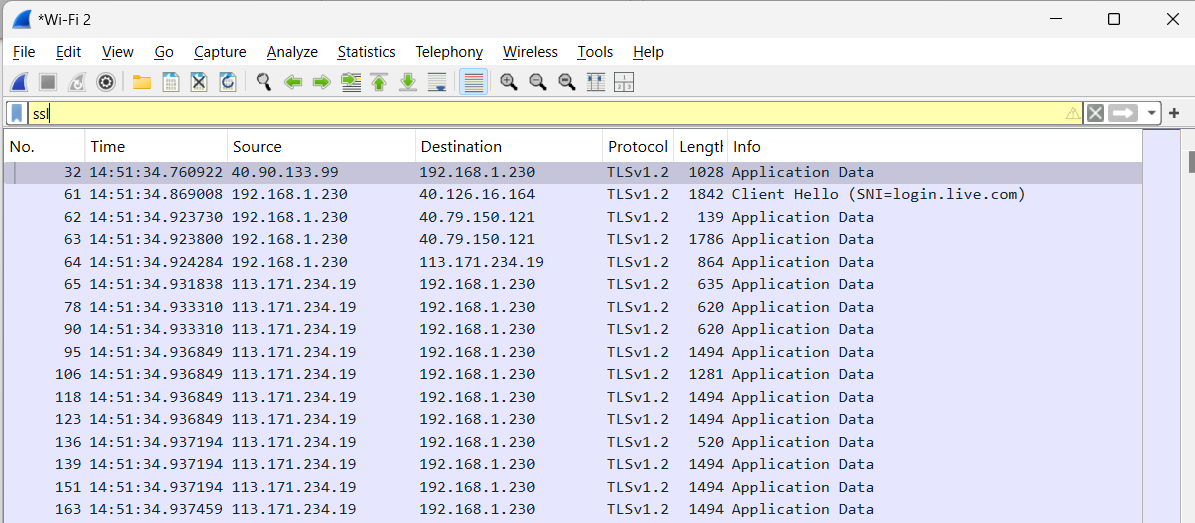
**GVHD: Bùi Xuân Giang**

|  |  |  |
| --- | --- | --- |
| **Sinh viên thực hiện** | | |
| Nguyễn Tấn Tài | : | 2212990 |

Thành phố Hồ Chí Minh, tháng 11 năm 2024

**Wireshark Lab: SSL v8.0**

Trong bài lab này, chúng ta sẽ tìm hiểu về giao thức Lớp Cổng Bảo mật (SSL - Secure Sockets Layer), tập trung vào các bản ghi SSL được gửi qua kết nối TCP. Chúng ta sẽ thực hiện điều này bằng cách phân tích dấu vết của các bản ghi SSL được gửi giữa máy của bạn và một máy chủ thương mại điện tử. Chúng ta sẽ tìm hiểu các loại bản ghi SSL khác nhau cũng như các trường trong các thông điệp SSL. Bạn có thể muốn xem lại Mục 8.6 trong sách để có thêm thông tin. Chúng tôi hiện đang phát triển một bài lab Wireshark bao gồm TLS, một giao thức thay thế cho SSL trong ấn bản thứ 8 của sách.



**1. Capturing packets in an SSL session**

Bước đầu tiên là thu thập các gói tin trong một phiên SSL. Để làm điều này, bạn nên truy cập vào một trang thương mại điện tử yêu thích và bắt đầu quá trình mua một món hàng (nhưng dừng trước khi thực hiện thanh toán thực tế!). Sau khi thu thập các gói tin với Wireshark, bạn cần đặt bộ lọc để chỉ hiển thị các khung Ethernet có chứa các bản ghi SSL được gửi và nhận bởi máy của bạn. (Một bản ghi SSL có cùng ý nghĩa với một thông điệp SSL). Bạn cần có kết quả như hình chụp màn hình ở trang trước.

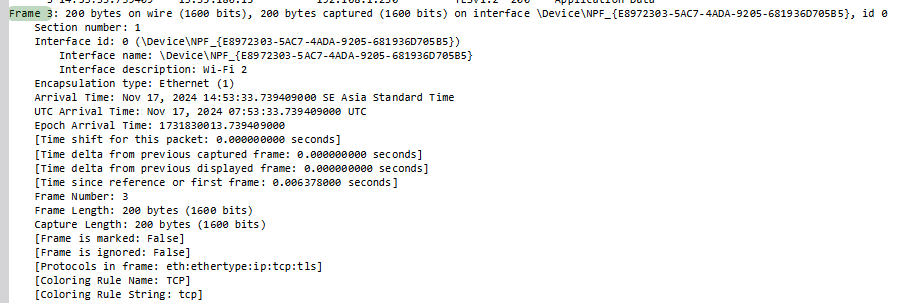
Nếu gặp khó khăn trong việc tạo dấu vết, bạn có thể tải tệp nén từ liên kết: http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip và giải nén tệp ssl-ethereal-trace-1 để sử dụng.

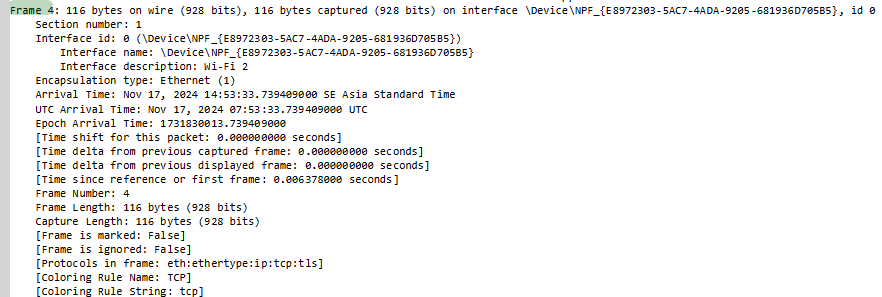
**2. A look at the captured trace**

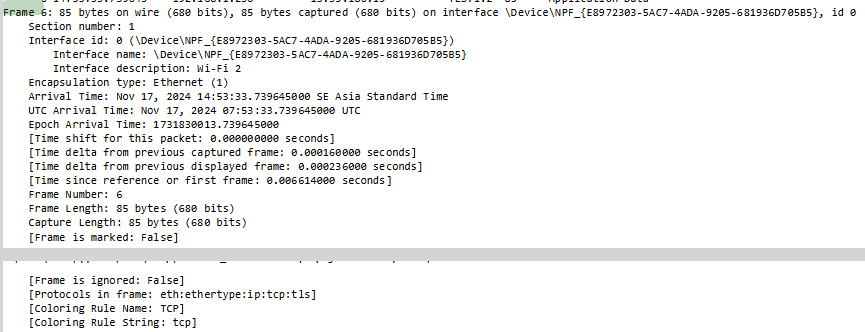
Giao diện Wireshark của bạn nên chỉ hiển thị các khung Ethernet chứa các bản ghi SSL. Lưu ý rằng một khung Ethernet có thể chứa một hoặc nhiều bản ghi SSL. (Điều này khác biệt với HTTP, trong đó mỗi khung chứa toàn bộ thông điệp HTTP hoặc một phần của thông điệp HTTP). Ngoài ra, một bản ghi SSL có thể không hoàn toàn vừa với một khung Ethernet, và trong trường hợp đó cần nhiều khung để chứa toàn bộ bản ghi.

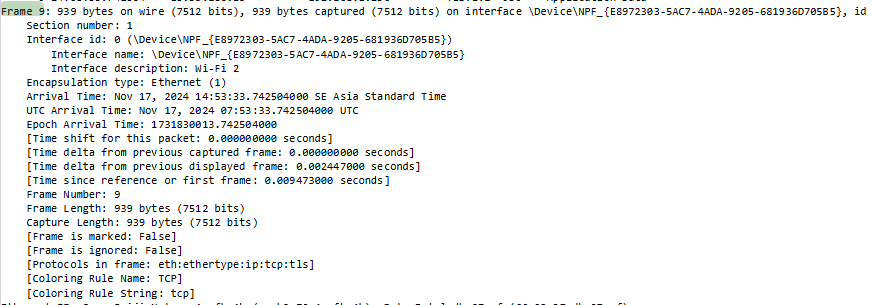
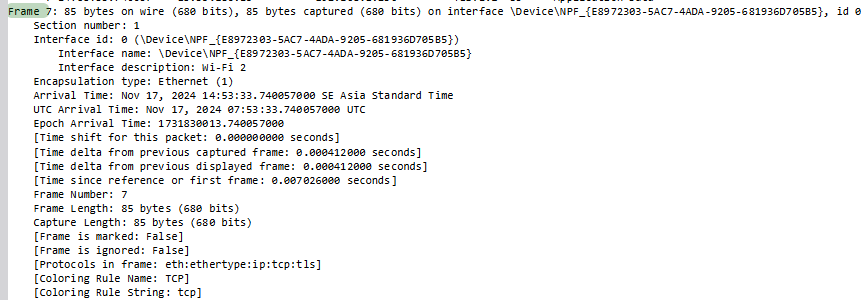
Mỗi khi có thể, khi trả lời một câu hỏi bên dưới, bạn nên in ra các gói tin trong dấu vết mà bạn sử dụng để trả lời câu hỏi đó. Chú thích bản in để giải thích câu trả lời của bạn. Để in một gói tin, sử dụng File->Print, chọn Selected packet only, chọn Packet summary line, và chọn mức chi tiết tối thiểu để trả lời câu hỏi.

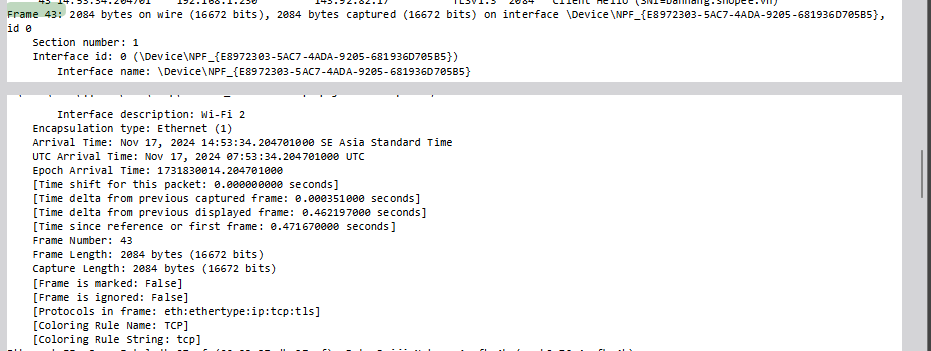
**Câu hỏi 1:** Đối với 8 khung Ethernet đầu tiên, xác định nguồn của khung (máy khách hoặc máy chủ), xác định số lượng bản ghi SSL được bao gồm trong khung và liệt kê các loại bản ghi SSL được bao gồm trong khung đó. Vẽ sơ đồ thời gian giữa máy khách và máy chủ, với một mũi tên cho mỗi bản ghi SSL.

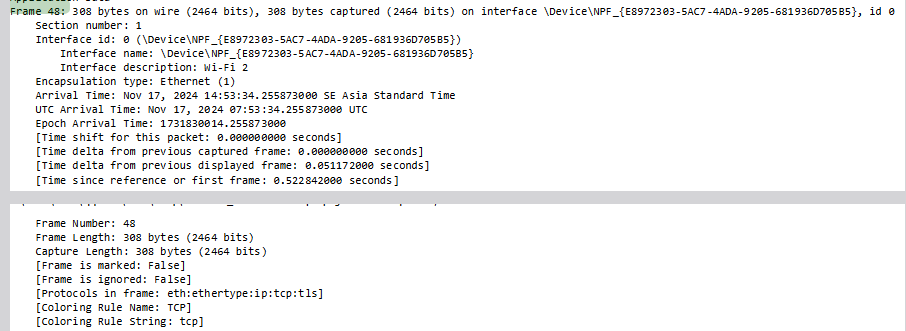


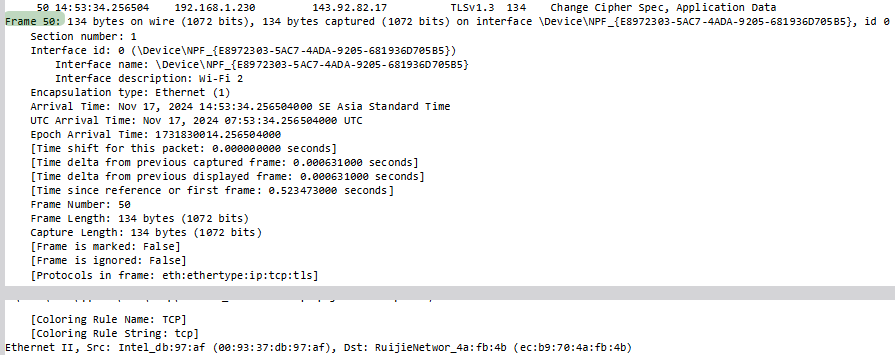












|  |  |
| --- | --- |
| Khung 3   * Nguồn: Máy chủ (13.35.186.15) * Đích: Máy khách (192.168.1.230) * Số lượng bản ghi SSL: 1 * Loại bản ghi SSL: Application Data | Khung 4   * Nguồn: Máy chủ (13.35.186.15) * Đích: Máy khách (192.168.1.230) * Số lượng bản ghi SSL: 1 * Loại bản ghi SSL: Application Data |
| Khung 6   * Nguồn: Máy khách (192.168.1.230) * Đích: Máy chủ (13.35.186.15) * Số lượng bản ghi SSL: 1 * Loại bản ghi SSL: Application Data | Khung 7   * Nguồn: Máy chủ (13.35.186.15) * Đích: Máy khách (192.168.1.230) * Số lượng bản ghi SSL: 1 * Loại bản ghi SSL: Application Data |
| Khung 9   * Nguồn: Máy chủ (13.35.186.15) * Đích: Máy khách (192.168.1.230) * Số lượng bản ghi SSL: 1 * Loại bản ghi SSL: Application Data | Khung 43   * Nguồn: Máy khách (192.168.1.230) * Đích: Máy chủ (143.92.82.17) * Số lượng bản ghi SSL: 1 * Loại bản ghi SSL: Client Hello |
| Khung 48   * Nguồn: Máy chủ (13.35.186.17) * Đích: Máy khách (192.168.1.230) * Số lượng bản ghi SSL: 4 * Loại bản ghi SSL: Server Hello, Change Cipher Spec, Application Data, Application Data | Khung 50   * Nguồn: Máy khách (192.168.1.230) * Đích: Máy chủ (13.35.186.17) * Số lượng bản ghi SSL: 2 * Loại bản ghi SSL: Change Cipher Spec, Application Data |

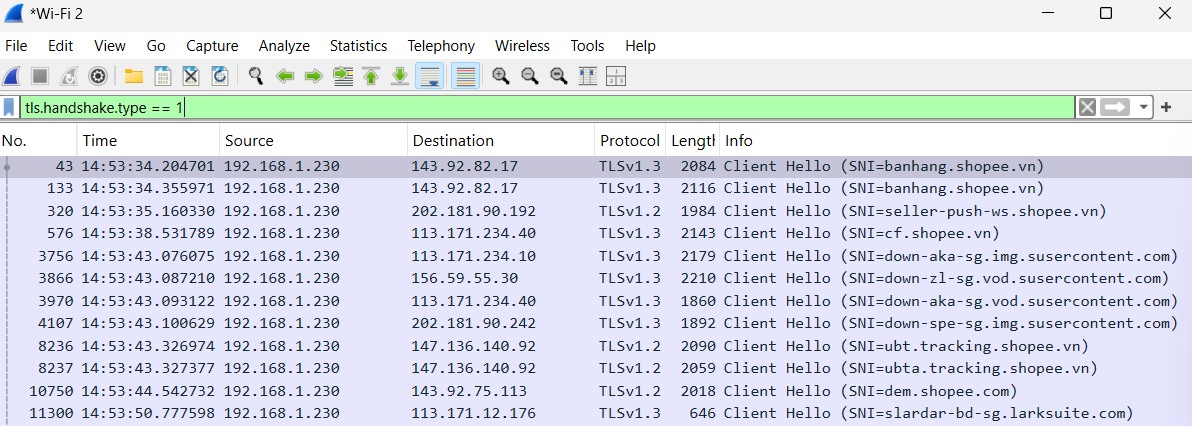
**Câu hỏi 2:** Mỗi bản ghi SSL bắt đầu với ba trường giống nhau (có thể có giá trị khác nhau). Một trong các trường này là "content type" (loại nội dung) và có độ dài một byte. Liệt kê cả ba trường và độ dài của chúng.

Trong giao thức SSL/TLS, mỗi bản ghi bắt đầu với ba trường chính, như sau:

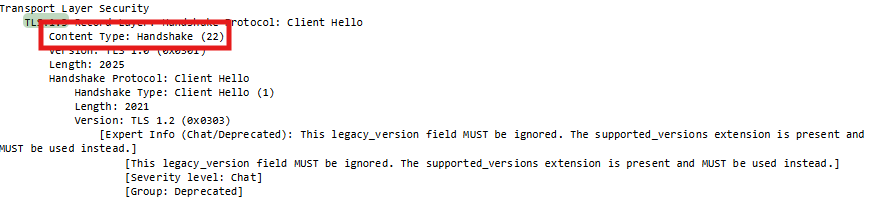
|  |  |
| --- | --- |
| **Content Type (Loại Nội Dung)**   * Mô tả: Trường này xác định loại bản ghi SSL/TLS, ví dụ: Handshake, Application Data, Change Cipher Spec. * Độ dài: 1 byte. * Giá trị ví dụ: 0x16 cho Handshake, 0x17 cho Application Data, 0x14 cho Change Cipher Spec. | **Version (Phiên Bản)**   * Mô tả: Trường này chỉ ra phiên bản SSL/TLS được sử dụng trong bản ghi. * Độ dài: 2 byte. * Giá trị ví dụ: 0x0301 cho TLS 1.0, 0x0303 cho TLS 1.2, 0x0304 cho TLS 1.3. |
| **Length (Độ Dài)**   * **Mô tả**: Trường này cho biết độ dài của phần dữ liệu đi kèm trong bản ghi. * **Độ dài**: 2 byte. * **Giá trị ví dụ**: Giá trị này phụ thuộc vào kích thước của dữ liệu bên trong bản ghi. | |

**ClientHello Record**

Sử dụng bộ lọc tls.handshake.type == 1 trong Wireshark để chỉ hiển thị các gói tin có bản ghi ClientHello. (Trong giao thức TLS, 1 là mã định danh cho bản ghi ClientHello).

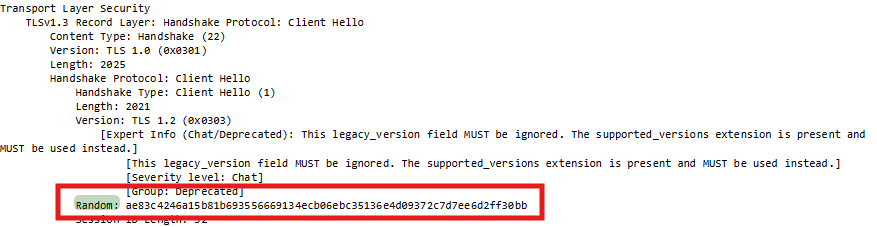


**Câu hỏi 3:** Mở rộng bản ghi ClientHello. (Nếu dấu vết của bạn chứa nhiều bản ghi ClientHello, hãy mở rộng khung chứa bản ghi đầu tiên). Giá trị của trường content type là gì?



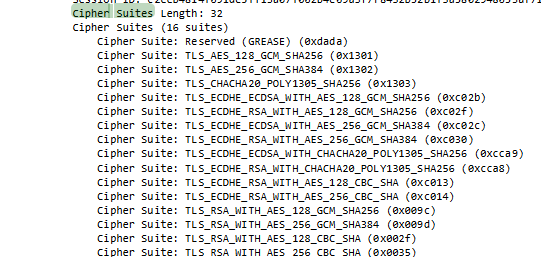
Trong gói tin ClientHello, trường Content Type có giá trị là 22 (0x16), đại diện cho loại bản ghi Handshake.

**Câu hỏi 4:** Bản ghi ClientHello có chứa một nonce (còn gọi là "challenge") không? Nếu có, giá trị của thách thức là gì dưới dạng mã thập lục phân?



Có, bản ghi ClientHello chứa một nonce (challenge) trong trường Random. Giá trị nonce là: ae83c4246a15b81b693556669134ecb06ebc35136e4d09372c7d7ee6d2ff30bb.

**Câu hỏi 5:** Bản ghi ClientHello có liệt kê các bộ mật mã mà nó hỗ trợ không? Nếu có, trong bộ đầu tiên được liệt kê, thuật toán mã công khai, thuật toán mã hóa đối xứng và thuật toán băm là gì?

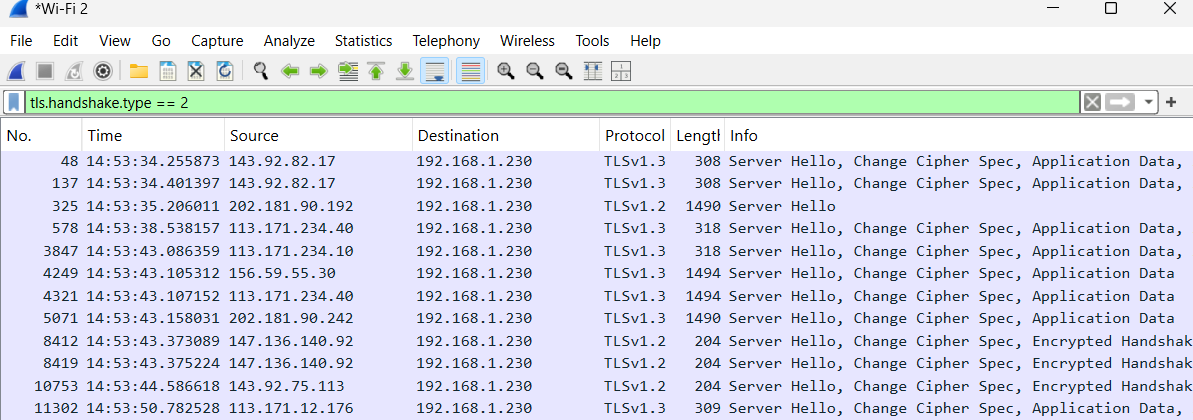


Có, bản ghi ClientHello liệt kê các bộ mật mã mà máy khách hỗ trợ. Bộ mật mã đầu tiên được liệt kê là TLS\_AES\_128\_GCM\_SHA256 với các thành phần:

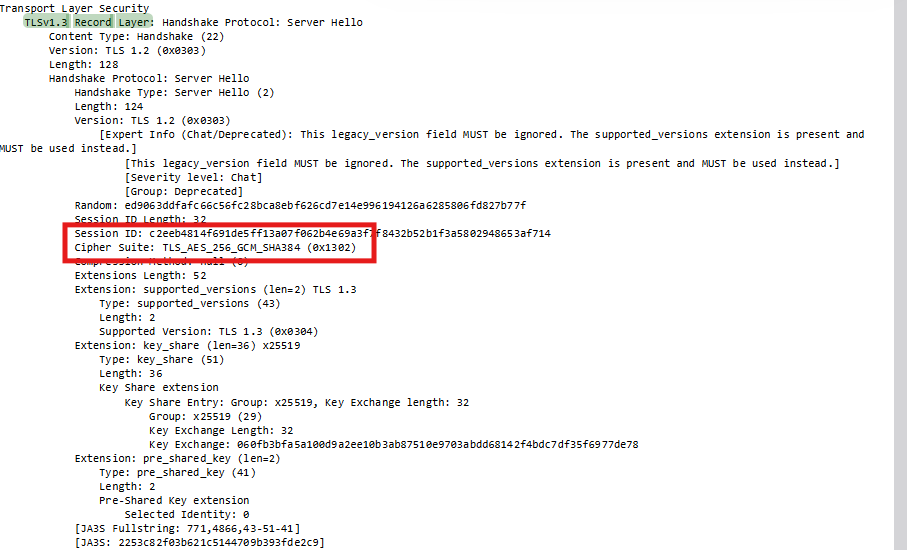
* Thuật toán mã công khai: Không có thuật toán mã công khai riêng lẻ trong bộ mật mã này, vì đây là một mã hóa tổng hợp cho TLS 1.3.
* Thuật toán mã hóa đối xứng: AES\_128\_GCM
* Thuật toán băm: SHA256

**ServerHello Record**

Sử dụng bộ lọc tls.handshake.type == 2 trong Wireshark để chỉ hiển thị các gói tin chứa bản ghi ServerHello. (Trong giao thức TLS, 2 là mã định danh cho bản ghi ServerHello).

****

**Câu hỏi 6:** Xác định bản ghi SSL ServerHello. Bản ghi này có chỉ định một bộ mật mã đã chọn không? Các thuật toán trong bộ mật mã đã chọn là gì?



Trong bản ghi ServerHello, bộ mật mã đã chọn là TLS\_AES\_256\_GCM\_SHA384. Bộ này bao gồm:

* Thuật toán mã hóa đối xứng: AES 256 trong chế độ GCM (Galois/Counter Mode)
* Thuật toán băm: SHA-384

**Câu hỏi 7:** Bản ghi này có chứa một nonce không? Nếu có, nó dài bao nhiêu? Mục đích của các nonce của máy khách và máy chủ trong SSL là gì?

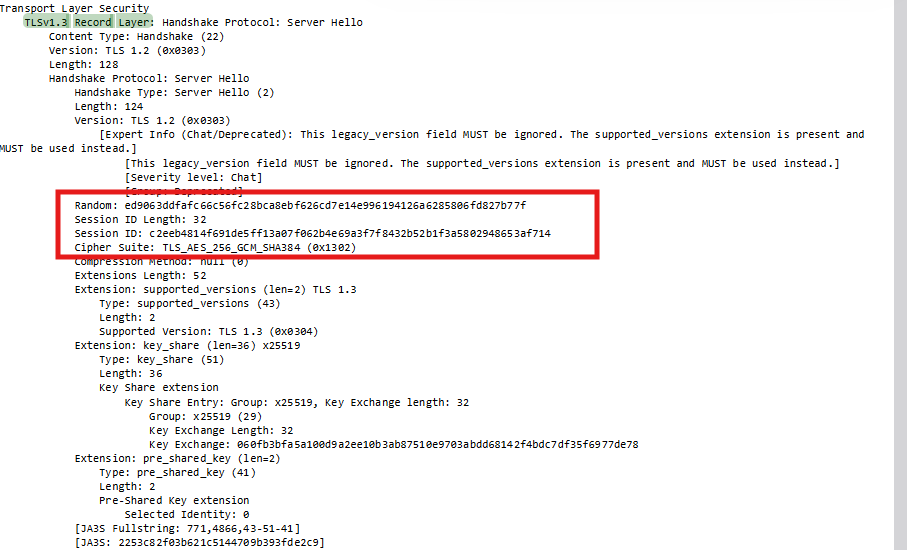


Có, bản ghi ServerHello chứa một nonce trong trường Random. Nonce này có độ dài 32 byte, giá trị:

ed9063ddfafc66c56fc28bca8ebf626cd7e14e996194126a6285806fd827b77f.

Mục đích: Các nonce của máy khách và máy chủ được sử dụng để tạo ra các khóa mã hóa duy nhất cho mỗi phiên, giúp đảm bảo tính bảo mật và độc nhất cho mỗi phiên SSL/TLS.

**Câu hỏi 8:** Bản ghi này có chứa một ID phiên không? Mục đích của ID phiên là gì?



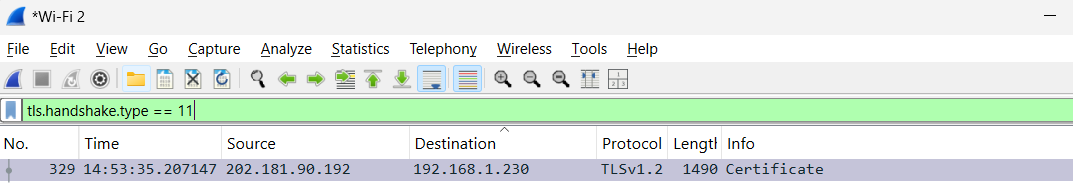
Có, bản ghi ServerHello chứa một Session ID với giá trị: c2eeb4 814f691de5ff13a07f062b4e69a3f7f8432b52b1f3a5802948653af714.

Mục đích của Session ID: Session ID giúp duy trì trạng thái phiên giữa máy khách và máy chủ, cho phép các phiên SSL/TLS tái sử dụng khóa mã hóa đã thương lượng trước đó, từ đó giảm thời gian thiết lập kết nối cho các lần kết nối tiếp theo.

**Câu hỏi 9:** Bản ghi này có chứa một chứng chỉ không, hay chứng chỉ được bao gồm trong một bản ghi riêng biệt? Chứng chỉ có vừa với một khung Ethernet không?

Bản ghi ServerHello không chứa chứng chỉ. Thay vào đó, chứng chỉ của máy chủ thường được gửi trong một bản ghi riêng có loại Certificate.

Kích thước của chứng chỉ: Chứng chỉ thường khá dài và có thể không vừa trong một khung Ethernet duy nhất, do đó, nó có thể cần được chia thành nhiều khung.

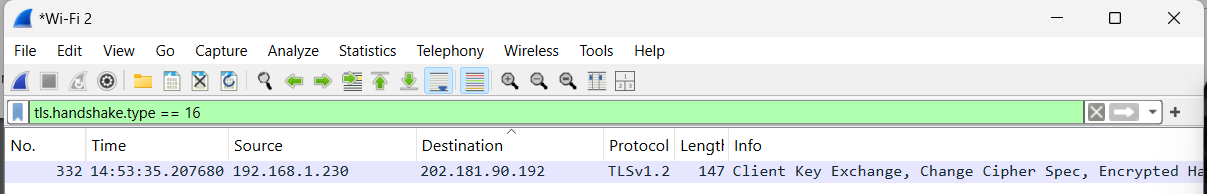


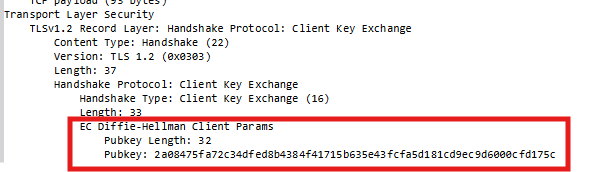
Để tìm bản ghi Certificate, ta có thể sử dụng bộ lọc tls.handshake.type == 11 trong Wireshark, vì bản ghi này không nằm trong ServerHello mà được gửi trong một bản ghi riêng biệt.

**Client Key Exchange Record**

**Câu 10:** Xác định bản ghi trao đổi khóa của máy khách. Bản ghi này có chứa một bí mật sơ bộ không? Bí mật này được dùng để làm gì? Bí mật này có được mã hóa không? Nếu có, nó dài bao nhiêu?

Trong giao thức TLS, bản ghi trao đổi khóa của máy khách có thể được tìm thấy trong các gói tin thuộc loại Client Key Exchange (TLS 1.2 trở xuống) hoặc Key Share (TLS 1.3). Sử dụng bộ lọc sau để tìm gói tin này: TLS 1.2 trở xuống: tls.handshake.type == 16; TLS 1.3: tls.handshake.type == 1 (ClientHello) và tìm trường **Key Share** trong gói tin này.





Bản ghi chứa bí mật sơ bộ: Bản ghi Client Key Exchange có chứa bí mật sơ bộ dưới dạng các tham số khóa công khai của khách hàng trong quá trình trao đổi khóa Diffie-Hellman (EC Diffie-Hellman Client Params).

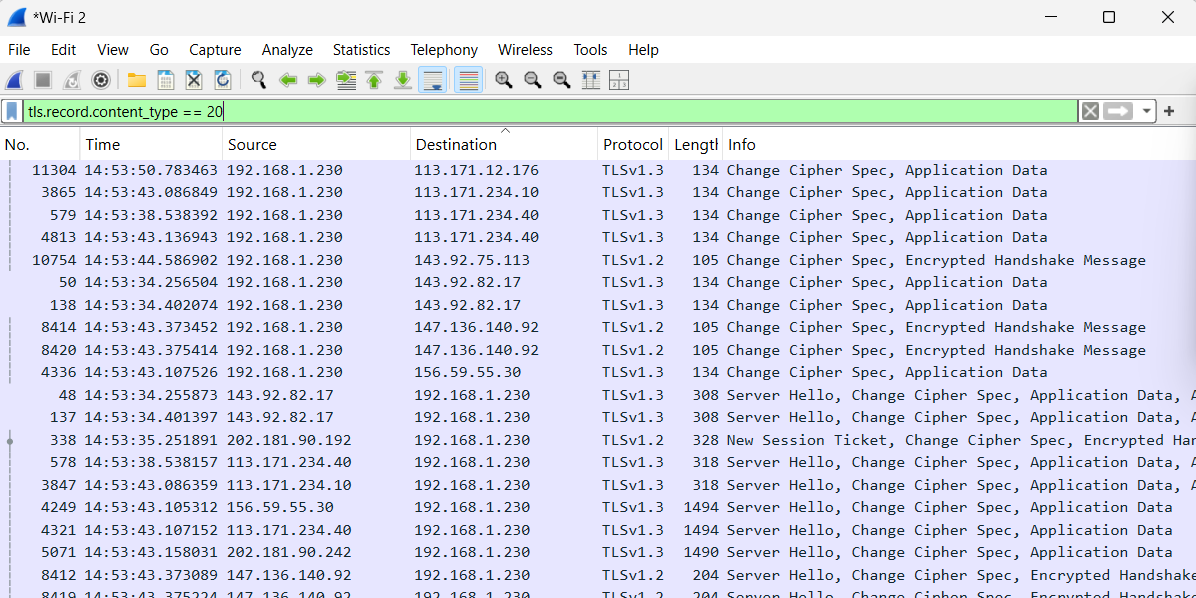
Mục đích của bí mật sơ bộ: Bí mật sơ bộ này được sử dụng để tạo khóa phiên giữa máy khách và máy chủ. Khóa phiên này sẽ được dùng để mã hóa và giải mã dữ liệu trong suốt phiên SSL/TLS nhằm bảo đảm tính bảo mật và toàn vẹn của dữ liệu.

Mã hóa bí mật sơ bộ: Trong phương thức EC Diffie-Hellman, bí mật sơ bộ không được mã hóa trực tiếp bằng khóa công khai của máy chủ. Thay vào đó, nó được trao đổi theo cách mà cả hai bên có thể tạo ra khóa phiên chung mà không cần truyền khóa thực tế.

Độ dài của bí mật sơ bộ: Bí mật sơ bộ có độ dài 32 byte.

**Change Cipher Spec Record (sent by client) and Encrypted Handshake Record**

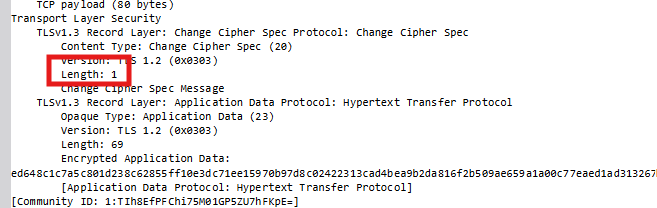
Sử dụng bộ lọc tls.record.content\_type == 20 trong Wireshark để tìm các gói tin có chứa bản ghi Change Cipher Spec (Loại bản ghi này có giá trị 20).



**Câu 11:** Mục đích của bản ghi Đổi Bộ Mã là gì? Số byte của bản ghi này trong dấu vết của bạn là bao nhiêu?

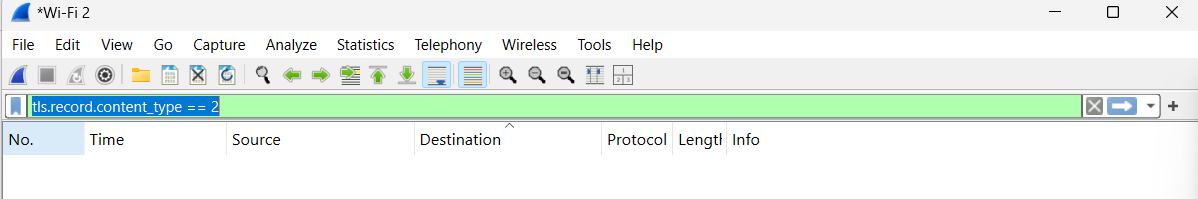
Mục đích của bản ghi Change Cipher Spec là báo hiệu cho phía đối tác (máy chủ hoặc máy khách) rằng từ thời điểm này trở đi, tất cả dữ liệu sẽ được mã hóa và giải mã bằng khóa phiên vừa thiết lập trong quá trình trao đổi khóa.

Trong gói tin chứa bản ghi Change Cipher Spec, xem trường Length để xác định số byte của bản ghi này. Bản ghi Change Cipher Spec chỉ có 1 byte.



**Câu 12:** Trong bản ghi bắt tay đã mã hóa, cái gì được mã hóa? Bằng cách nào?

Sử dụng bộ lọc tls.handshake.type == 22 trong Wireshark để tìm các gói tin chứa bản ghi Encrypted Handshake Message. Loại bản ghi này thường là bản ghi Finished và có thể xuất hiện sau bản ghi Change Cipher Spec.

 Bản ghi Encrypted Handshake Message chứa bản ghi Finished, là thông điệp cuối cùng của quy trình bắt tay SSL/TLS và được mã hóa để đảm bảo rằng cả hai bên đều có cùng khóa phiên. Nội dung của bản ghi Finished bao gồm một giá trị xác thực được tạo dựa trên tất cả các bản ghi bắt tay trước đó.

Bản ghi này được mã hóa bằng khóa phiên (session key) được tạo ra sau khi trao đổi khóa giữa máy khách và máy chủ. Phương thức mã hóa phụ thuộc vào bộ mật mã đã chọn, ví dụ như AES-GCM trong trường hợp của TLS\_AES\_256\_GCM\_SHA384.

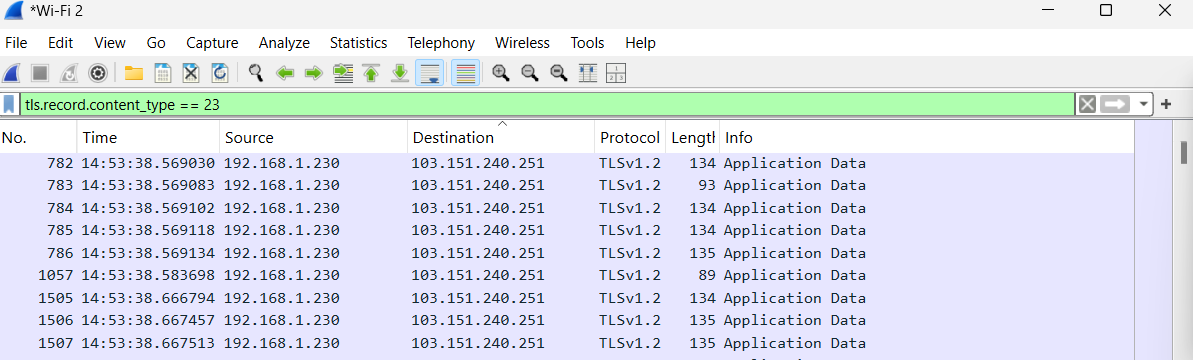
**Câu 13:** Máy chủ cũng có gửi một bản ghi đổi bộ mã và bản ghi bắt tay đã mã hóa cho máy khách không? Các bản ghi này có khác biệt so với các bản ghi được gửi bởi máy khách không?

Bản ghi Change Cipher Spec từ máy khách và máy chủ có cùng mục đích, nhưng được gửi từ các phía khác nhau để báo hiệu rằng mỗi bên đã sẵn sàng sử dụng mã hóa.

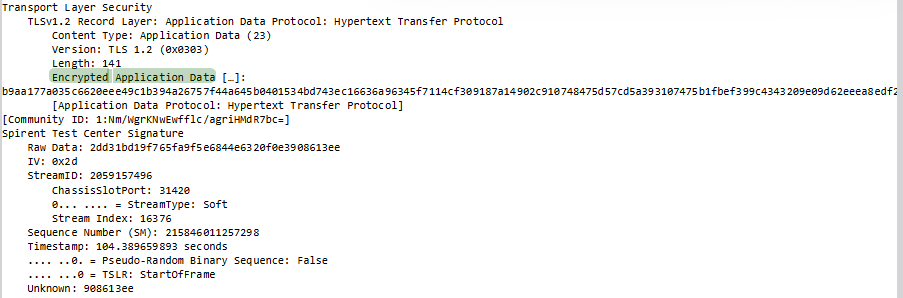
Bản ghi Encrypted Handshake Message từ máy chủ cũng chứa giá trị xác thực cho tất cả các bản ghi trước đó, tương tự như của máy khách, nhưng sẽ có giá trị khác biệt do quá trình mã hóa và các tham số khác nhau.

**Application Data**

Sử dụng bộ lọc tls.record.content\_type == 23 trong Wireshark để tìm các gói tin chứa bản ghi Application Data. Loại bản ghi này chứa dữ liệu ứng dụng đã được mã hóa.



**Câu 14:** Dữ liệu ứng dụng được mã hóa như thế nào? Các bản ghi chứa dữ liệu ứng dụng có bao gồm MAC không? Wireshark có phân biệt được giữa dữ liệu ứng dụng đã mã hóa và MAC không?



Phương thức mã hóa: Dữ liệu ứng dụng trong bản ghi Application Data được mã hóa bằng phương thức AES 256 GCM, dựa trên bộ mật mã đã thương lượng trong quá trình bắt tay SSL/TLS. GCM (Galois/Counter Mode) là chế độ mã hóa tích hợp xác thực, giúp đảm bảo tính toàn vẹn và bảo mật cho dữ liệu truyền tải.

Sự hiện diện của MAC: Trong chế độ GCM, MAC được tích hợp vào quá trình mã hóa, do đó không có MAC riêng biệt cho từng bản ghi dữ liệu.

Khả năng của Wireshark: Wireshark hiển thị dữ liệu ứng dụng đã mã hóa dưới dạng Application Data mà không phân biệt rõ ràng giữa dữ liệu và MAC, vì MAC đã được tích hợp trong mã hóa GCM.

**Câu 15:** Bình luận và giải thích bất cứ điều gì mà bạn thấy thú vị trong dấu vết.

Trong dấu vết này, quá trình bắt tay SSL/TLS sử dụng bộ mật mã TLS\_AES\_256\_GCM\_SHA384, một bộ mật mã mạnh mẽ với mã hóa AES 256-bit và hàm băm SHA-384, đảm bảo mức độ bảo mật cao cho kết nối. Điểm đáng chú ý là cả máy khách và máy chủ đều gửi bản ghi Change Cipher Spec trước khi chuyển sang mã hóa dữ liệu, báo hiệu rằng mỗi bên đã đồng thuận về việc bắt đầu mã hóa phiên. Tất cả dữ liệu ứng dụng sau đó được mã hóa hoàn toàn, đảm bảo tính bí mật và an toàn cho thông tin trao đổi.