Wireshark Homework

미디어학과 박성범

다음 문제를 확인하여 정리하시오. 문제의 이해에 애매한 부분이 있으면, 스스로 이해한바에 맞추어 분석하여 보이면 됩니다.

1. (50점) (HTTP 분석) 다음과 같이 Web Server에 접속하고 캡쳐한후, 다음 내용을 분석하여 정리하시오.

- 이전에 접속하여 임시저장된 페이지가 있고 Cookie를 사용한 적이 있는 Web Server에 접속하여 캡쳐함.

- http와 https를 모두 지원하는 Web Server를 선택함.

- 이 문제는 http로 접속한 경우에 대하여 답하면 됨.

(1) Request와 Response 각각 메시지 한개씩을 선택하여, 이를 나타내시오.

```
Hypertext Transfer Protocol
  GET /_resources/new/img/index/btn_pop_close.gif HTTP/1.1\r\n
  Host: www.ajou.ac.kr\r\n
  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:63.0) Gecko/20100101 Firefox/63.0\r\n
  Accept: */*\r\n
  Accept-Language: ko-KR,ko;q=0.8,en-US;q=0.5,en;q=0.3\r\n
  Accept-Encoding: gzip, deflate\r\n
  Referer: http://www.ajou.ac.kr/main/index.jsp\r\n
  DNT: 1\r\n
  Connection: keep-alive\r\n
  [truncated]Cookie: PHAROS_VISITOR=00006cab01655c787fe45b11ca1e0013; JSESSIONID=3lIbYVyT0k8lrw
```

http://www.ajou.ac.kr에 접속했을 때, source(192.168.0.6)에서 destination(202.30.0.19)으로 HTTP request를 보낸 모습이다.

```
Hypertext Transfer Protocol
  HTTP/1.1 200 OK\r\n
  Date: Thu, 11 Oct 2018 13:02:29 GMT\r\n
  ETag: "0-608-58ae60a8"\r\n
  Last-Modified: Thu, 23 Feb 2017 04:10:16 GMT\r\n
  Accept-Ranges: bytes\r\n
  Content-Length: 1544\r\n
  Content-Type: image/gif\r\n
  Connection: Keep-Alive\r\n
  Keep-Alive: timeout=60\r\n
  \r\n
  [HTTP response 2/2]
  [Time since request: 0.017741000 seconds]
  [Prev request in frame: 1395]
  [Prev response in frame: 1486]
  [Request in frame: 1504]
  File Data: 1544 bytes
Compuserve GIF, Version: GIF89a
```

앞선 request에 대한 response 메시지의 모습이다.

(2) (1)의 메시지들로부터 알수있는 정보를 기록하시오.

요청 메시지에서는 ▲source가 destination의 */_resources/new/img/index* 경로에 위치한 *btn_pop_close.gif* 파일을 요청 ▲destination의 호스트네임은 www.ajou.ac.kr ▲source는 윈도우 64비트 기반 Firefox 운영체제를 사용 ▲현재 요청된 페이지의 이전 웹 페이지 주소가 http://www.ajou.ac.kr/main/index.jsp ▲source가 persistent connection을 요청 등의 정보를 알 수 있다.

응답 메시지에서는 ▲HTTP 요청이 성공했고, 리소스가 body에 담겨 전송 ▲메시지가 만들어진 날짜는 2018년 10월 11일 목요일 ▲캐시된 데이터가 마지막으로 수정된 것은 2017년 2월 23일 목요일 ▲콘텐츠 타입이 gif 이미지 ▲persistent connection으로 연결 등의 정보를 알 수 있다.

(3) Cookie가 적용된 경우, 어떻게 사용되었는지를 확인하고 정리하시오.

JSESSIONID와 PHAROS_VISITOR라는 이름의 쿠키가 사용되었으며, 그 값은 암호화되어 있다. JSESSIONID는 톰캣 환경에서 JSP를 실행할 때 세션ID를 저장하기 위해 만들어지는 쿠키다. PHAROS_VISITOR는 어떤 목적으로, 어떻게 사용되었는지 확인할 수 없었다.

(4) Conditional GET이 적용된 경우, 어떻게 사용되었는지 확인하여 적으시오.

(1)의 패킷은conditional GET이 적용되지 않아 다른 패킷을 캡처했다.

```
Hypertext Transfer Protocol
  GET /_resources/main/img/banner/20181004_pr.jpg HTTP/1.1\r\n
  Host: www.ajou.ac.kr\r\n
  User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:63.0) Gecko/20100101 Firefox/63.0\r\n
  Accept: */*\r\n
  Accept-Language: ko-KR,ko;q=0.8,en-US;q=0.5,en;q=0.3\r\n
  Accept-Encoding: gzip, deflate\r\n
  Referer: http://www.ajou.ac.kr/main/index.jsp\r\n
  DNT: 1\r\n
  Connection: keep-alive\r\n
  [truncated]Cookie: PHAROS_VISITOR=00006cab01655c787fe45b11ca1e0013; JSESSIONID=31IbYVyT0k8lrw
  If-Modified-Since: Thu, 11 Oct 2018 02:43:57 GMT\r\n
  If-None-Match: "0-5335-5bbeb8ed"\r\n
```

www.ajou.ac.kr/_resources/main/img/banner/20181004_pr.jpg를 요청하는 메시지이며, 앞선 메시지와 달리 If-Modified-Since 헤더 라인이 붙어있는 것을 볼 수 있다.

```
Hypertext Transfer Protocol
  HTTP/1.1 304 Not Modified\r\n
  Date: Thu, 11 Oct 2018 15:39:41 GMT\r\n
  Connection: Keep-Alive\r\n
  Keep-Alive: timeout=60\r\n
  ETag: "0-5335-5bbeb8ed"\r\n
  \r\n
  [HTTP response 1/2]
  [Time since request: 0.015261000 seconds]
  [Request in frame: 6418]
  [Next request in frame: 6445]
```

해당 요청에 대한 응답 메시지는 304 Not Modified였다. Conditional GET 요청을 통해 캐시된 리소스의 Last-Modified와 비교했지만, 리소스가 수정되지 않았다는 의미다. 따라서 업데이트 없이 캐시된 리소스를 그대로 출력했다.

2. (30점) 1의 Web Server에 https로 접속하시오.

(1) 1과 비교하여 추가되는 과정이 있는지 확인하여 적으시오.

https://www.ajou.ac.kr에 접속했다.

```
46302 2316.437381   202.30.0.19     192.168.0.6     TLSv1   91    Encrypted Alert
46462 2332.208828   192.168.0.6     202.30.0.19     TLSv1   599   Client Hello
46464 2332.236639   202.30.0.19     192.168.0.6     TLSv1   1514  Server Hello
46469 2332.251024   202.30.0.19     192.168.0.6     TLSv1   475   Certificate, Server Hello Done
46471 2332.252311   192.168.0.6     202.30.0.19     TLSv1   380   Client Key Exchange, Change Cipher Spec, Encrypted Handshake Message
46473 2332.270108   202.30.0.19     192.168.0.6     TLSv1   304   New Session Ticket, Change Cipher Spec, Encrypted Handshake Message
```

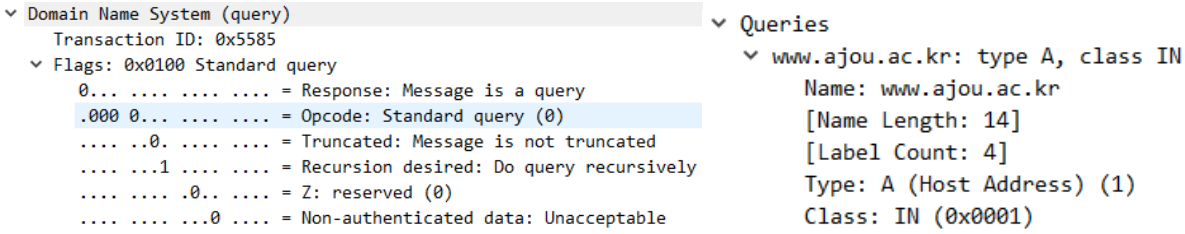
접속 초기에 TCP 프로토콜의 three-way handshake과 별도로 TLS handshake를 수행한다. Client Hello와 Server Hello 메시지, Certificate, Server Hello Done 메시지를 주고받으며 인증서를 확인하고 클라이언트 키를 교환하는 과정을 거친다.

(2) 1과 비교하여 Request와 Response 메시지가 어떻게 되었는지를 확인하여 적으시오.

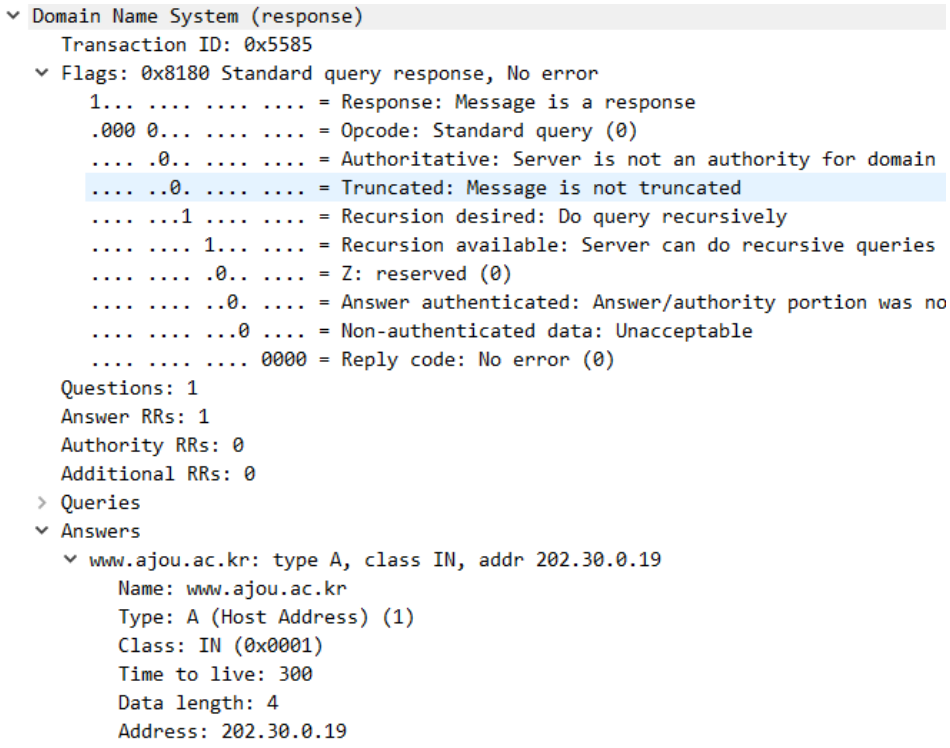
```
Secure Sockets Layer
  TLSv1 Record Layer: Application Data Protocol: http-over-tls
      Content Type: Application Data (23)
      Version: TLS 1.0 (0x0301)
      Length: 656
      Encrypted Application Data: a906b20191e1148849ed19c89e6fad1798dbbd8cd4f34a81...
```

해당 패킷의 콘텐츠 타입이 Application Data라는 점만 알 수 있을 뿐, 실제 데이터는 암호화되어 어떤 내용인지 알 수 없었다. 즉, HTTP 메시지와 달리 네트워크의 패킷을 캡처하는 것만으로 클라이언트가 네트워크를 통해 어떤 정보를 주고받는지 확인할 수 없었다.

3. (20점) 1의 과정에서 적용된 DNS 과정 하나를 분석하여 보이시오.



클라이언트에서 로컬 DNS 서버로 쿼리를 보낼 때 Flags 값을 '0x0100 Standard query'라고 보낸 것을 볼 수 있었다. 이때 1은 RD값으로, recursive하게 쿼리를 보내겠다는 것을 의미한다. 더불어, 레코드 타입이 A이니까 IPv4를 받을 수 있을 것이다.



응답에 IPv4 주소가 담겨왔다. 응답 Flags는 '0x8180 Standard query response, No error'로 왔고, 여기서 8은 response, 1은 RD값을 의미해 역시 recursive query 방식을 사용하겠다는 응답을 보냈다. 따라서 DNS 쿼리는 recursive하게 동작했을 것이다.

Wireshark는 클라이언트에서 전송한 패킷과 최종 응답 패킷만 캡처하는 것 같아서 보다 자세한 정보를 얻기 위해 dig 명령으로 쿼리 과정을 추적해봤다.

```
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 52907
;; flags: qr ra; QUERY: 1, ANSWER: 13, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;.                              IN      NS

;; ANSWER SECTION:
.                       123765  IN      NS      a.root-servers.net.
.                       123765  IN      NS      l.root-servers.net.
.                       123765  IN      NS      e.root-servers.net.
.                       123765  IN      NS      m.root-servers.net.
.                       123765  IN      NS      b.root-servers.net.
.                       123765  IN      NS      k.root-servers.net.
.                       123765  IN      NS      c.root-servers.net.
.                       123765  IN      NS      i.root-servers.net.
.                       123765  IN      NS      f.root-servers.net.
.                       123765  IN      NS      h.root-servers.net.
.                       123765  IN      NS      d.root-servers.net.
.                       123765  IN      NS      g.root-servers.net.
.                       123765  IN      NS      j.root-servers.net.

;; Received 239 bytes from 210.220.163.82#53(210.220.163.82) in 12 ms
```

초기 쿼리는 ". IN NS"로, 로컬 DNS 서버(210.220.163.82)로부터 root DNS 서버 리스트를 응답 받았다. 이때 flags에 'ra'가 포함되어 있으므로 recursive query 방식을 사용했음을 알 수 있다.

```
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 63473
;; flags: qr; QUERY: 1, ANSWER: 0, AUTHORITY: 8, ADDITIONAL: 10

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;ajou.ac.kr.                    IN      A

;; ADDITIONAL SECTION:
b.dns.kr.               172800  IN      A       61.74.75.1
c.dns.kr.               172800  IN      A       203.248.246.220
d.dns.kr.               172800  IN      A       203.83.159.1
e.dns.kr.               172800  IN      A       202.30.124.100
f.dns.kr.               172800  IN      A       218.38.181.90
g.dns.kr.               172800  IN      A       202.31.190.1
d.dns.kr.               172800  IN      AAAA    2001:dcc:4::1
e.dns.kr.               172800  IN      AAAA    2001:dcc:5::100
g.dns.kr.               172800  IN      AAAA    2001:dc5:a::1

;; Received 654 bytes from 202.12.27.33#53(m.root-servers.net) in 45 ms
```

이어서 "ajou.ac.kr. IN A"라는 쿼리를 보내 root DNS 서버(m.root-servers.net)로부터 .kr top-level DNS 서버 리스트를 응답 받았다.

```
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 22397
;; flags: qr; QUERY: 1, ANSWER: 0, AUTHORITY: 6, ADDITIONAL: 3

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;ajou.ac.kr.                    IN      A

;; ADDITIONAL SECTION:
madang.ajou.ac.kr.      86400   IN      A       202.30.0.11
madang1.ajou.ac.kr.     86400   IN      A       202.30.0.7

;; Received 609 bytes from 203.248.246.220#53(c.dns.kr) in 14 ms
```

그 다음은 kr. top-level DNS 서버(c.dns.kr)에게 destination의 authoritative DNS 서버에 대한 응답이 왔다.

```
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 9418
;; flags: qr aa ra; QUERY: 1, ANSWER: 1, AUTHORITY: 2, ADDITIONAL: 3

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags: do; udp: 4096
;; QUESTION SECTION:
;ajou.ac.kr.                    IN      A

;; ANSWER SECTION:
ajou.ac.kr.             300     IN      A       202.30.0.11

;; ADDITIONAL SECTION:
madang.ajou.ac.kr.      300     IN      A       202.30.0.11
sambong.ajou.ac.kr.     300     IN      A       202.30.0.48

;; Received 130 bytes from 202.30.0.11#53(madang.ajou.ac.kr) in 17 ms
```

마지막으로 destination의 authoritative DNS 서버(madang.ajou.ac.kr)에서 “ajou.ac.kr. 300 IN A 202.30.0.11” 응답을 받았다. 여기서는 생략된 최종 응답 메시지는 Wireshark에서 로컬 서버가 클라이언트에게 보냈음을 확인할 수 있었다.