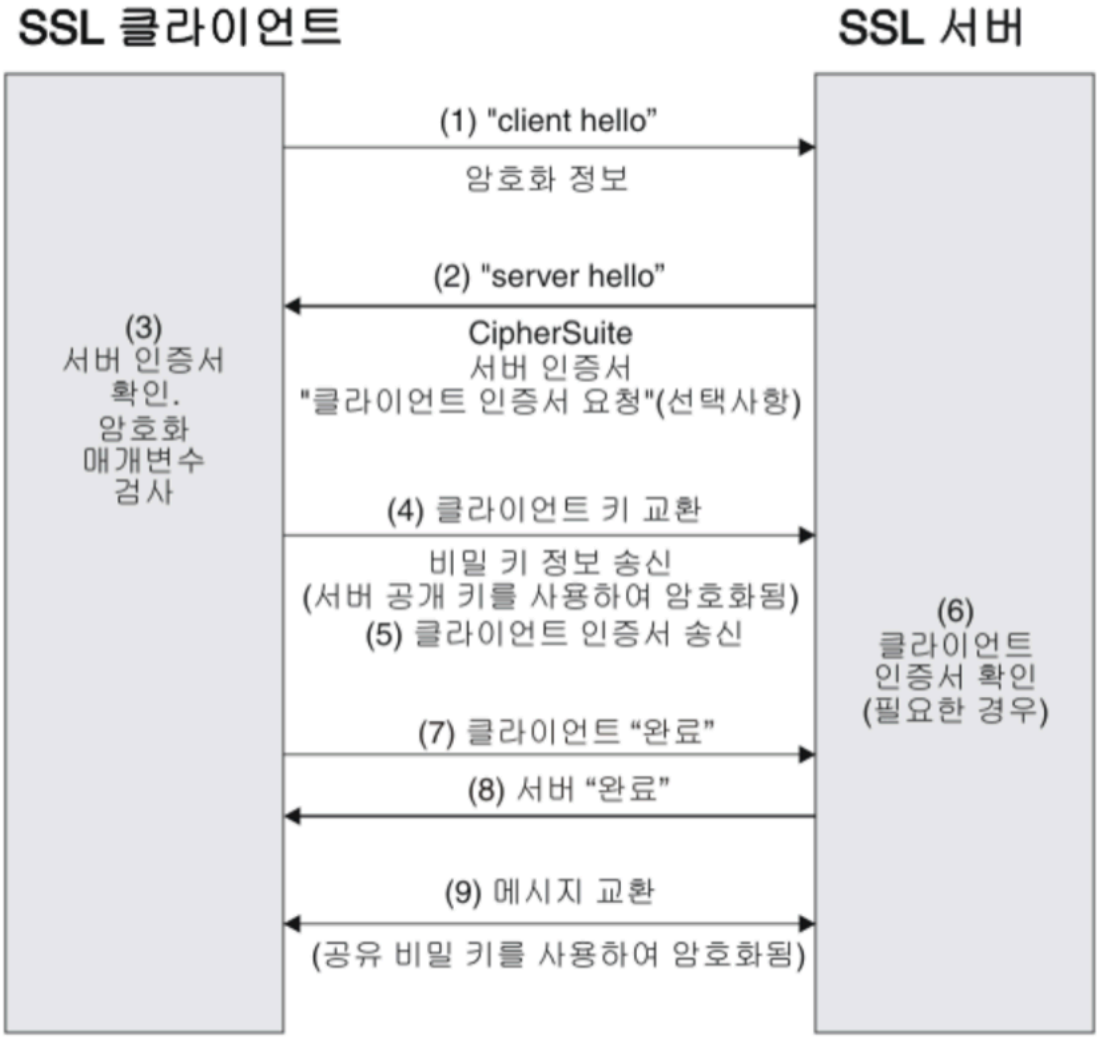
유해사이트 차단 우회 방법

- 사이트 접속 패킷(get,client hello)을 분할하여 보내는 방식
  - http 의 경우 get 요청을 분할.
  - https 의 경우 handshake 과정에서 client hello 패킷에 접속하려는 사이트의 도메인이 있는데 이 패킷을 분할.
    - SSL Handshake 설명



1. `client Hello`: client에서 SSL버전 정보와 지원하는 암호화 방식, 무작위 바이트 문자열(이후에 사용하게 되는 중요한 데이터입니다)이 포함 되어 전달 됩니다. 이미 SSL handshake를 했었다면 세션을 재사용 할 수 있습니다.

2. `server Hello`: 지원하는 암호화 방식 중 서버에서 어떤것을 사용할 지, 세션 ID, 서버측에서 생성한 무작위 바이트 문자열을 전송합니다. 클라이언트에서 인증서를 요구하게 되면 SSL 인증서를 전송하게 됩니다.

3. 인증서를 받게 되면 위에서 언급했던 방식(인증서가 서버가 신뢰할 수 있는지 어떻게 판단할까?)로 신뢰할 수 있는 사이트 인지 판단하게 됩니다.이 과정을 통해 클라이언트는 공개키를 얻게 됩니다.

4. 이후 클라이언트는 자신이 만든 무작위 바이트 문자열과 서버쪽에서 전송된 무작위 바이트 문자열을 조합하여 `pre master secret`키를 생성합니다. 이 키는 이후 데이터를 주고 받을 때 대칭키 방식을 사용할 때 사용하게 됩니다. 이 `pre master secret`키를 서버로 전송할 때 인증서에서 받았던 키를 이용하여 공개키 방식 암호화를 하게 됩니다. 그리고 서버쪽으로 전송하게 됩니다.

5. 서버쪽에서는 수신한 `pre master secret`키를 비밀키를 이용하여 복호화 하여 얻게된다.

6. server client 둘 다 일련의 과정을 거쳐 `pre master secret`키를 `master key`로 만들게 되고 이 master key를 이용하여 `session key`를 만들게 된다. 이후 데이터를 주고 받을 때 `session key`를 `대칭키 방식`으로 이용하여 통신하게 됩니다.

7. 통신이 끝나면 세션키를 폐기합니다.

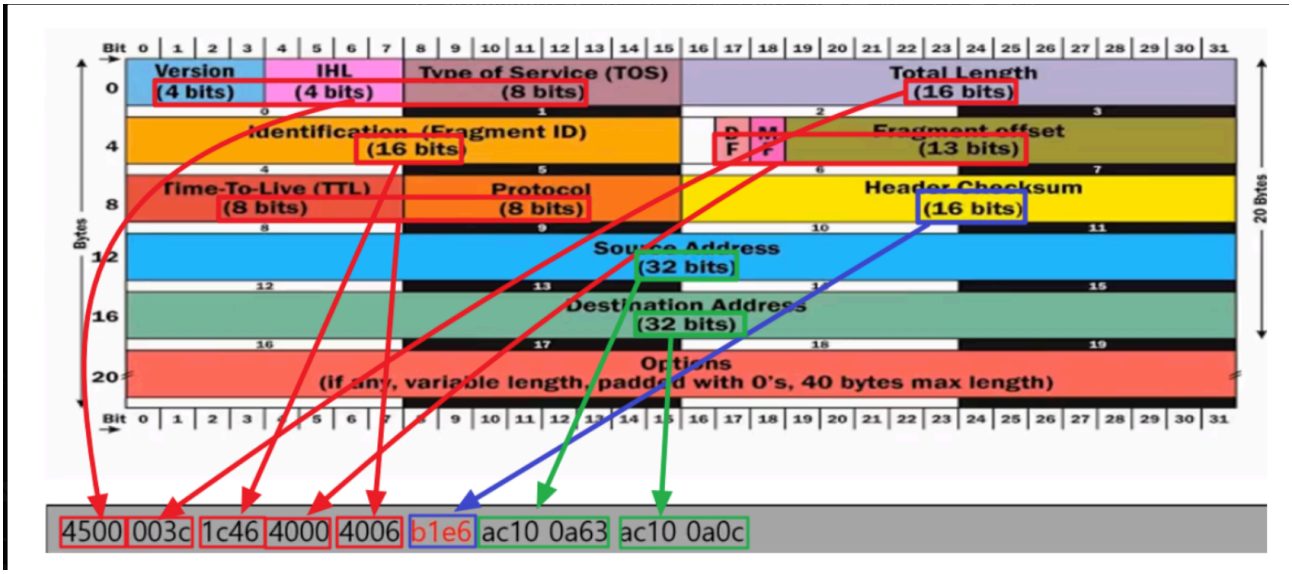
- iptables 의 규칙을 이용하여 패킷을 관리.
  - iptables -A OUTPUT -j NFQUEUE —queue-num 0 (코드작성 후 변경 예정)

[RAW Socket 특징]

- 응용 계층과 전송 계층, 네트워크 계층에서 모두 접근이 가능하다.
- 네트워크 계층 헤더와 전송 계층 헤더를 직접 제어가능하다.
- 네트워크 계층으로 전송되는 모든 패킷을 전부 모니터링 및 감지 가능하다.

:: raw socket coding

- queue 에 적재된 패킷을 조작하기 위해 raw socket 사용
  1. 유해사이트에 접근시 기존 패킷을 복사 해두고 queue 에 대기중인 패킷은 Drop 시킨다.
  2. 복사된 패킷을 조작하고, raw socket을 이용하여 사이트 접속 패킷 전송.
     - 패킷을 조작하면 checksum 값이 달라지므로 변조된 패킷에 맞춰 checksum 값을 변동.
       1. checksum 값 계산하기 위해서 bit단위로 쪼개진 부분을 합쳐야함.(shift 연산 사용 예정)
       2. sift연산을 코드로 작성해야함.(현재 방법 찾는중)
     - ip header 의 checksum 계산법



1. 수신측에서 IP헤더를 16bit(2byte) 씩 나눈다.
2. 나눈 비트중 체크섬은 메세지를 보낸쪽에서 체크섬을 구해서 포함시켜 보낸값이므로 이 체크섬 값으로는 내가 구한 체크섬 값과 비교를 위해 사용한다 따라서 체크섬 값을 제외하고 나머지를 모두 더한다.
3. 캐리값이 발생하면 더한다.(2byte 초과하면)
4. 1의 보수를 취한다.
   - 0100 1110 0001 1001
   → 1011 0001 1110 0110 = b 1 e 6 (checksum)

5. 구한 값과 전달받은 체크섬을 비교한다.

- tcp header 의 checksum 계산법
  - ip 에서 구하는 checksum과 유사하다

- Pseudo Header를 참조해야한다

· Pseudo Header는 총 12바이트 길이로써 일부 IP Header를 참조하여 만들어 집니다.

필드명	크기	설 명
Source IP	4	출발지 IP
Destination IP	4	목적지 IP
Reserved (항상 0)	1	8비트, 항상 0이다
프로토콜	1	IP헤더에서 알아낸 프로토콜 필드 값
TCP 길이	2	TCP 헤더 + DATA의 총 길이(바이트)

- Pseudo Header 합 + TCP Segment(TCP header + Data) 합
- 위의 결과를 가지고 1의 보수.
- checksum 부분은 ip와 마찬가지로 비교용으로 사용한다.

3. Coding 에 사용되는 문제가 되는 구조체
   - #include <netinet/ip.h> //linux

```jsx
struct ip {
#if BYTE_ORDER == LITTLE_ENDIAN
        u_char ip_hl:4,         /* header length */
               ip_v:4;                  /* version */
#endif
#if BYTE_ORDER == BIG_ENDIAN
        u_char ip_v:4,                  /* version */
               ip_hl:4;         /* header length */
#endif
        u_char ip_tos;                  /* type of service */
        short   ip_len;                 /* total length */
        u_short         ip_id;                  /* identification */
        short   ip_off;                 /* fragment offset field */
#define         IP_DF 0x4000                    /* dont fragment flag */
#define         IP_MF 0x2000                    /* more fragments flag */
        u_char ip_ttl;                  /* time to live */
        u_char ip_p;                    /* protocol */
        u_short         ip_sum;                         /* checksum */
        struct   in_addr ip_src,ip_dst; /* source and dest address */
};
```

- 2byte로 합치기 위해 4bit를 shift 연산을 코드로 작성 방법 찾는 중.

<다음주 진행 내용>

raw socket 을 이용하여 유해사이트 차단 우회 실패시 원인 찾고, 코드 수정 .

성공시 코드 최적화 및 공유기에 적용방법 찾기.