22.12.20

📌GC 복습

- 이야기 못한 것

Concurrent mark and sweep

동작 과정

1. initial mark : GC Root로부터 바로 참조되거나, young 영역의 살아있는 객체로부터 바로 참조되는 old 영역의 객체를 mark (STW 발생)
2. Concurrent mark : initial mark단계에서 mark한 객체부터 시작해서 old영역을 순회하면서 살아있는 모든 객체들을 markt한다. (STW 발생하지 않고 애플리케이션 스레드를 멈추지 않고 동작)
3. Concurrent reclean :

Card라는 힙영역 안에 dirty로 표시한 객체로부터 참조되고 있는 객체를 mark (STW 발생하지 않고 애플리케이션 스레드를 멈추지 않고 동작)

4. Concurrent aborable Preclean

변화한 객체들을 계속 스캔

(STW 발생하지 않고 애플리케이션 스레드를 멈추지 않고 동작, Final Remark의 수행시간을 최대한 짧게 만든다)

5. Final Remark

STW를 통해 애플리케이션 스레드를 잠시 멈춤으로써 객체들의 상태를 완전히 반영한다. 이 단계를 young 영역이 거의 비워져있을 때 수행하게 하도록 하여 STW을 일으키는 동작이 연쇄적으로 발생하는 것을 방지

6. Concurrent sweep

애플리케이션 스레드와 병렬적으로 수행, 사용하지 않는 객체들을 정리하여 빈공간을 확보한다.

📌TCP 연결

클라이언트와 서버와의 관계라 가정한다.

1. 3 way HandShake

1) 클라이언트에서 요청 (SYN)

- SYN은 'synchronize sequence numbers', 다른 컴퓨터로 전송 된 TCP 패킷으로 연결이 이루어 지도록 요청
- SYN는 TCP에서 세션을 성립할때 가장 먼저 보내는 패킷
- 시퀀스 번호를 임의적으로 설정하여 세션을 연결하는 데에 사용되며, 초기에 시퀀스 번호를 보내게 된다.
- 클라이언트는 SYN_SENT 상태가 된다.

2) 서버가 상대방으로부터 패킷을 받았다는 응답(ACK)과 SYN flag가 설정된 패킷을 발송하고 다시 ACK로 응답하기를 기다린다. 'acknowledgment' 의 약자

- 서버는 SYN_RECEIVED 상태

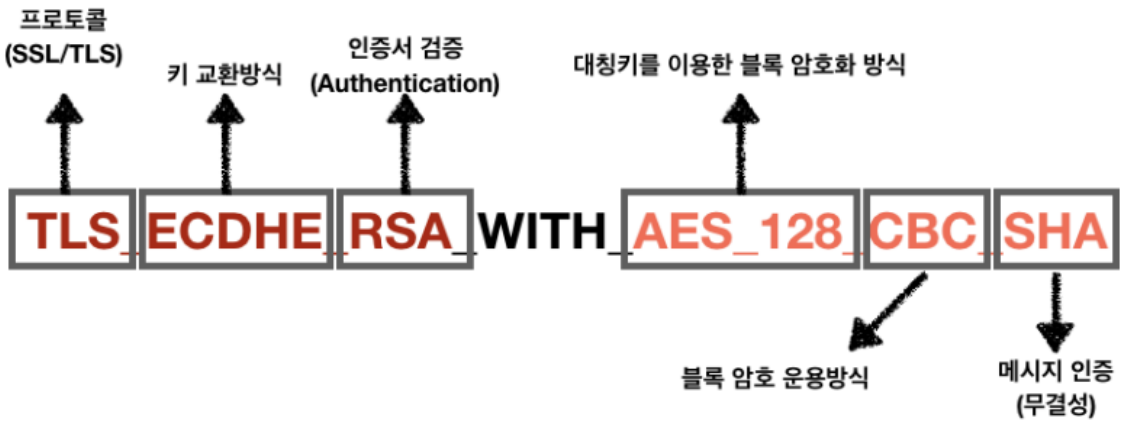
3) 클라이언트가 서버에게 ACK로 응답한다.

- 클라이언트는 ESTABLISHED 상태(확인, 확립이 된 상태. 연결이 확인된 상태)

2. 데이터 전송전 데이터를 암호화하기 위한 공개키 만들기 (TLS 3 way handshake)

1) Client Hello (클라이언트)

- 클라이언트가 서버에 연결을 시도하며 전송하는 패킷
- 자신이 사용가능한 Cipher Suite 목록, Session ID, SSL protocol version, random byte 전달
  - Ciper Suite : SSL protocol version, 인증서 검정, 암호화 프로토콜, 해시 방식 등의 정보를 담고 있다.
  - Ciper Suite의 알고리즘에 따라 데이터를 암호화 한다.



2) 서버

(1) Server Hello

- 클라이언트가 보내온 ClientHello 패킷에서 Cipher Suite 중 하나를 선택해서 알린다.
- 자신의 SSL Protocol Version 도 같이 보낸다.

(2) Certificate

- 서버가 자신의 SSL 인증서를 클라이언트에 전달
- SSL 인증서 내부에는 서버가 발생한 공개키 존재 (개인키는 서버만 소유)

3) 인증서 검증 및 공개키 암호화 (클라이언트) + Client Key Exchange

- 인증서 검증 : 서버가 보낸 CA의 개인키로 암호화하된 SSL인증서를 CA의 공개키로 복호화
- 공개키를 해시함수로 공유키(데이터를 실제로 암호화하는 키)를 만들고 SSL 인증서 내부에서 가져온 공개키를 암호화하여 서버에게 전송(Client Key Exchange) → 체크할 것.

만약, 서버의 공개키가 SSL 인증서에 없을 경우, 서버가 직접 전달하는 Server Key Exchange과정이 발생한다.

4) Client Key Exchange (클라이언트)

참고) https://aws-hyoh.tistory.com/entry/HTTPS-통신과정-쉽게-이해하기-3SSL-Handshake

3. 4 way HandShake

세션 종료를 위해 수행된다.

1) FIN 플래그 전송 (클라이언트)

- 클라이언트가 연결을 종료하겠다는 요청
- 클라이언트는 **FIN-WAIT** 상태

2) 확인 메세지인 ACK 전송 (서버)

- **CLOSE_WAIT** 상태

3) 클라이언트에게 FIN 전송(서버)

- 연결해지를 위한 준비가 되었다는 알림.
- **LAST-ACK** 상태

4) 서버에게 ACK 전송(클라이언트)

- **TIME-WAIT** 상태

📌 TCP 타이머

1) 재전송 타이머 : 정해진 시간(RTO, Retransmission Timeout) 내 수신 확인응답(ACK)이 안되면 재전송

2) 영속 타이머 :

- 윈도우 크기 결정을 위한 타이머
- 만일 윈도우 크기가 0 으로 도착되고도 일정시간 이후에 아무런ACK가 없으면, 즉, 수신측에서 송신측에 더이상데이터 보내지 말라는 뜻이므로, 수신측 상황을 알아보기 위해 1바이트 길이의 데이터를 전송하여 보고, 이에대한ACK를 기다리게 되는데, 이때 알아보는패킷을 `WindowProbe패킷`

3) 시간 대기 타이머

- TCP 연결 종료 후에 이 기간 동안 만 연결을 유지
- 이전 연결 종료 전의 어떤 패킷이 늦게, 중복지연 도착하게되는 것을 방지

4) keepalive 타이머 (연결 유지 타이머)

- 이미 설정된 연결이 오랫동안 휴지 상태에 있지 않도록 하기 위함
- 킵얼라이브프로브(Probe)패킷을 75초 간격으로 10번 송신하고도 응답이 없으면, 연결을끊음