|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | **작성자** | **2018180033 이세철 2018180046 허재성 2018182009 김승환** | **팀명** | XD |
| **계획 시트** | <https://docs.google.com/spreadsheets/d/1rUOOKpcVfkma18YsvTZXZz44z6UhbGfOsrimVgs-mzE/edit#gid=0> | | | | |
| **주차** | **8~9** | **기간** | **2023.03.28~2023.04.13** | **지도교수** | **정 내 훈**(서명) |
| **이번주 한일** | * **이세철:** * **허재성:** * **김승환: 1. 서버 이중화 버그 수정**   **① Failover가 이루어지면서 서버의 세션 정보가 초기화되던 문제해결**  **② 두번째 Failover부터 클라이언트-서버 연동이 안되던 문제해결**  **2. 릴레이 서버 제작**  **3. 서버 구조 개선** | | | | |

**<상세 수행내용>**

1. **이세철**

* 1

1. **허재성**
2. 1
3. **김승환**
4. **Failover가 이루어지면서 서버의 세션 정보가 초기화되던 문제해결**

텍스트이(가) 표시된 사진

자동 생성된 설명

오른쪽에 있는 1번서버(Active)가 왼쪽의 0번서버(Standby)에게 자신이 갖고 있는 세션 정보를 넘겨주고, Standby서버가 이를 받아 자신의 세션 정보를 업데이트 하는 모습입니다.

저번주에는 저 상태에서 Active서버를 종료하게 되면 기존의 Standby서버는 Active서버로 승격되어 서비스를 이어받게 되는데, 이때 클라이언트가 서버에게 로그인 패킷을 보내는 바람에 지금까지 업데이트 해 놓은 세션 정보들이 초기화되는 문제가 있었습니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

현재는 문제를 해결하였고, Failover가 이루어질 때 세션 정보가 초기화되지 않는 것을 확인할 수 있었습니다.

1. **두번째 Failover부터 클라이언트-서버 연동이 안되던 문제해결**

원인①: 실행될 때부터 Active상태인 서버만 데이터동기화 패킷을 보내주도록 되어있어서, 실행 도중 Failover로 인해 Active상태가 된 서버는 데이터동기화 패킷을 보내지 못하는 문제가 있었습니다. 이로 인해 첫번째 Failover 이후로는 Active서버가 Standby서버로 데이터동기화 패킷을 보내지 못해 다음 번 Failover가 이루어졌을 때 Active로 승격된 서버에는 세션 정보가 없는 문제가 발생한 것이었습니다.

원인②: 클라이언트는 서버의 Failover가 이루어졌을 때 현재 연결된 서버와의 통신을 종료하고, 새로 승격된 Active서버에게 연결을 요청하도록 구현하였는데, 0번 서버와 연결되었을 때 1번 서버로 새로 연결해야 정상이지만 0번 서버로 다시 연결을 시도하는 문제가 있었습니다. 이는 단순한 실수여서 금방 해결할 수 있었습니다.

현재는 이러한 문제들을 모두 수정하였고, 두번째 Failover 이후에도 문제없이 동작하고 있습니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

(▲ 첫번째 Failover 직후의 모습입니다.)

텍스트이(가) 표시된 사진

자동 생성된 설명

(▲ 두번째 Failover 직후의 모습입니다.)

텍스트이(가) 표시된 사진

자동 생성된 설명

(▲ 세번째 Failover 직후의 모습입니다.)

1. **릴레이서버 제작 (진행중)**

텍스트이(가) 표시된 사진

자동 생성된 설명

클라이언트와 로직 서버의 사이에서 중계 및 Connection Pool의 역할을 수행할 릴레이 서버를 제작하고 있습니다. 현재 ‘릴레이 서버-로직 서버들’, ‘릴레이 서버-클라이언트’ 연결에서 릴레이 서버의 Accept와 Recv는 잘 이뤄지고 있지만 Send만 작동하지 않는 문제가 있습니다. 아마도 비정상 상태의 소켓이 원인이지 않을까 싶어서 현재 그쪽 방향으로 확인해보고 있습니다.

1. **서버 구조 개선**

기존의 코드는 IOCP 객체를 여러 개 만들었고, 워커 스레드 함수도 로직 전용 함수와 이중화 전용 함수 두 가지로 만들었기 때문에 코드가 굉장히 길어지고 지저분했었습니다.

몇 주전 게임서버 수업에서 교수님께서 IOCP 객체를 여러 개 만드는 학생이 있는데 그러면 안 된다고 말씀하셨을 때 저는 속으로 아차 싶었습니다. 그 때 교수님께서 IOCP 객체를 여럿 만드는 것이 오버헤드를 발생시킬 수도 있다고 하신 점이 생각나 이번에 서버 구조를 개선하게 되었습니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

IOCP 객체는 하나만 사용하도록 수정하였고, 클라이언트 로직을 위한 워커 함수와 이중화를 위한 워커 함수를 하나로 합치면서 key값을 통해 이 작업이 클라이언트의 로직인지 서버 이중화 관련된 작업인지를 구분하도록 수정하였습니다.

또한 기존 코드에는 CreateIoCompletionPort의 세 번째 파라미터인 Completion Key값에 매직넘버를 넣어 사용했는데, 이 역시 상수들로 바꿔 워커 스레드 함수 내에서 작업들을 보다 편하게 구분할 수 있게 되었고, 동시에 가독성도 높아졌습니다.

이러한 과정에서 예상치 못한 버그를 접하면서 작업에 지연을 불러왔던 것 같습니다. 처음부터 설계를 잘하였더라면 이러한 번거로운 일은 하지 않았을 거란 생각이 들었습니다. 코딩을 하기 전에 설계를 먼저 잘 구성해야 하는 이유를 몸소 느끼게 된 계기가 된 것 같습니다.

|  |  |  |  |
| --- | --- | --- | --- |
| **문제점 정리** | 1. 클라이언트:  2. 서버: 릴레이 서버에서 Send가 되지 않는 문제. | **해결 방안** |  |
| **다음 주차** | **10** | **다음 기간** | **2023.04.14~2023.04.28** |
| **다음주 할 일** | 이세철:  허재성:  김승환: ① 릴레이서버 문제해결  ② ‘클라이언트-릴레이서버-로직서버’ 구조 완성  ③ 릴레이 서버의 이중화  ④ 로그인+매치메이킹 서버 제작 | | |
| **지도교수**  **Comment** |  | | |