Redis Week4

Redis administration

Redis setup tips

Linux

- 메모리의 오버커밋을 1로 설정한다. vm.overcommit_memory=1
- Transparent Huge Pages (THP)를 disable(never)로 설정한다.(**echo never > /sys/kernel/mm/transparent_hugepage/enabled**)
- Max number of open files를 65536으로 설정한다.

Memory

- 주기적으로 메모리 사용령을 관찰한다. 메모리가 부족하여 메모리 스와핑이 발생하지 않도록 한다. 메모리 부족은 레디스 서버 성능에 치명적이다. 가능한 한 30% 정도 여유 메모리를 갖도록 서버를 운영한다. 시스템 메모리가 16GB면 레디스 서버가 12GB 정도 사용하도록 한다.
- Maxmemory를 설정하는 것도 고려해 볼 수 있다. 예) maxmemory 12gb
- 인스톤스에서 명시적 maxmemory 옵션 제한을 설정을 하면 시스템 메모리 제한에 거의 도달했을 때 실패하지 않고 오류를 보고하도록 한다. mexmory 데이터 이외의 Redis에 대한 오버헤드와 Fragmentation 오버헤드를 계산하여 설정해야 한다. 따라서 여유 메모리가 10GB라 생각되면 8 또는 9로 설정해야한다.
- 쓰기 작업이 많은 어플리케이션에서 레디스를 사용하는 경우 RDB 파일을 디스크에 저장하거나 AOF 로그를 다시 작성하는 동안 redis는 일반적으로 사용되는 메모리를 최대 2배까지 사용할 수 있다. 사용되는 추가 메모리는 저장 프로세스 동안 쓰기로 수정된 메모리 페이지 수에 비례하므로 이 시간 동안 터치된 키의 수에 비례하는 경우가 많다. 그에 따라 메모리 크기를 조정해야한다.

Imaging

- 로그 파일을 설정한다. default "" -> **logfile "redis.log"**

- Working directory를 설정한다. default ./ -> **dir /redis/6000/**
- 레디스 서버가 데몬(background)으로 실행되도록 설정한다.default no -> **daemonize yes**
- maxclients수를 예상 클라이언트 수를 고려해 설정한다.default 10000 -> **maxclients 1128**

Replication

- 레디스가 사용하는 메모리 양에 비례하여 사소하지 않은 복제 백로그를 설정한다. 백로그를 사용하면 복제본이 마스터 인스턴스와 더 쉽게 동기화 할 수 있다.
- Replication backlog size 설정: default 1mb -> **repl-backlog-size 50mb** (10~100mb 사이로 설정한다)
- 복제를 사용하는 경우 레디스는 지속성이 비활성화 된 경우에도 RDB 저장을 수행한다. (이는 디스크 없는 복제에는 적용되지 않는다.) 마스터에 디스크 사용량이 없으면 디스크 없는 복제를 활성화 한다.
- 복제를 사용하는 경우 마스터가 persistence를 활성화 했는지 또는 충돌 시 자동으로 재시작을 하지 않는지 확인을 한다. 복제본은 마스터의 정확한 복제본을 유지하려고 시도하기 때문에 마스터가 빈 데이터 셋으로 다시 시작되면 복제본도 지워진다.

Security

- bind IP를 설정한다. default 127.0.0.1 -> **bind 192.168.1.100 127.0.0.1**
- protected-mode를 설정한다. **default yes**
- 기본 포트 이외의 포트를 사용하자. default 6379 -> **port 6000**

Redis Security

security model

Redis는 신뢰할 수 있는 환경 내에서 신뢰할 수 있는 클라이언트가 액세스하도록 설계되어 있다. 즉, 일반적으로 Redis 인스턴스를 인터넷에 직접 노출하거나 일반적으로 신뢰할 수 없

는 클라이언트가 Redis TCP 포트 또는 UNIX 소켓에 직접 액세스할 수 있는 환경에 노출하는 것은 좋지 않다.

예를 들어 데이터베이스, 캐시 또는 메시징 시스템으로 Redis를 사용하여 구현된 웹 애플리케이션의 일반적인 컨텍스트에서 애플리케이션의 프런트 엔드(웹 측) 내부 클라이언트는 Redis를 쿼리하여 페이지를 생성하거나 요청된 작업을 수행한다.

이 경우 웹 애플리케이션은 Redis와 신뢰할 수 없는 클라이언트(웹 애플리케이션에 액세스하는 사용자 브라우저) 간의 액세스를 중재한다.

일반적으로 Redis에 대한 신뢰할 수 없는 액세스는 항상 ACL을 구현하고 사용자 입력을 검증하며 Redis 인스턴스에 대해 수행할 작업을 결정하는 계층에 의해 조정되어야 한다.

Network security

레디스 포트에 대한 액세스는 네트워크에서 신뢰할 수 있는 클라이언트를 제외한 모든 사람에게 거부 되어야 하므로 레디스를 실행하는 서버는 레디스를 사용하여 어플레케이션을 구현하는 컴퓨터에서만 직접 액세스 할 수 있어야 한다.

redis.conf 파일 에 다음과 같은 줄을 추가하여 Redis를 단일 인터페이스에 바인딩할 수 있다.

```
bind 127.0.0.1
```

Protected mode

버전 3.2.0 부터 레디스는 암호없이 기본 구성으로 실행되면 보호 모드라는 특수 모드에 들어간다. 이 모드에서 레디스는 루프백 인터페이스의 쿼리에만 응답하고 다른 주소에서 연결하는 클라이언트에 문제와 레디스를 레디스를 올바르게 구성하는 방법을 설명하는 오류로 응답한다.

Authentication

ACL

Access Control List의 줄임말인 Redis ACL은 실행할 수 있는 명령과 액세스할 수 있는 키 측면에서 특정 연결을 제한할 수 있는 기능이다. 작동 방식은 연결 후 클라이언트가 인증을 위해 사용자 이름과 유효한 암호를 제공해야 한다는 것이다. 인증에 성공하면 연결이 지정된 사용자 및 사용자가 가진 제한과 연결된다. 새 연결이 이미 "기본" 사용자로 인증되도록

Redis를 구성할 수 있다(이것이 기본 구성임). 기본 사용자를 구성하면 부작용으로 명시적으로 인증되지 않은 연결에 특정 기능 하위 집합만 제공할 수 있다.

TLS

Redis는 클라이언트 연결, 복제 링크 및 Redis 클러스터 버스 프로토콜을 포함한 모든 통신 채널에서 TLS를 선택적으로 지원한다.

Disallowing specific commands

Redis에서 명령을 허용하지 않거나 추측할 수 없는 이름으로 이름을 변경하여 일반 클라이언트가 지정된 명령 집합으로 제한되도록 할 수 있습다.

예를들어 redis config 명령을 일반 사용자가 호출 할 수 없어야 하지만 인스턴스를 관리하는 시스템은 할 수 있어야 한다. 이 경우 명령 테이블에서 명령의 이름을 바꾸거나 명령을 완전히 숨길 수 있어야 한다. 이 기능은 redis.conf 구성 파일 내에서 사용할 수 있는 명령문으로 제공된다. 예를들어

```
rename-command CONFIG b840fc02d524045429941cc15f59e41cb7be6c52
```

위의 예시에서 CONFIG 명령은 추측할 수 없는 이름으로 변경되었다. 다음 예와 같이 이름을 빈 문자열로 변경하여 완전히 허용하지 않을 수도 있다.

```
rename-command CONFIG ""
```

Redis configuration

http://redisgate.kr/redis/server/redis_conf_han.php#auto-aof-rewrite-spec-time

Redis replication

레디스 고가용성 기능은 마스터-레플리카 복제로 이루어진다, 복제용 레디스 인스턴스가 마스터 인스턴스의 정확한 복사본이 될 수 있다. 복제본은 링크가 끊어질때마다 마스터에 자동으로 다시 연결되며 마스터에 어떤 일이 발생하든 관계없이 복제본의 정확한 복사본이 되려고 시도한다.

1. 마스터와 복제 인스턴스가 잘 연결된 경우 마스터는 클라이언트 쓰기, 키 만료 또는 삭제, 마스터 데이터 셋 변경 등으로 인해 마스터 측에서 발생하는 데이터 셋에 대한 영향을 복제하기 위해 일련의 명령을 복제본에 전송하여 복제본을 업데이트한다.
2. 네트워크 문제로 인해 마스터와 복제본 간의 연결이 끊어지거나 마스터 또는 복제본에서 시간 초과가 감지되어 복제본이 다시 연결되고 부분적인 재동기화를 시도한다. 즉, 연결이 끊기는 동안 놓친 명령 스트림의 일부만 가져오려 한다.
3. 부분적인 재동기화가 불가능한 경우, 복제본은 전체 재동기화를 요청한다. 여기에는 마스터가 모든 데이터의 복제본으로 전송한 다음 데이토 셋이 변경될 때 명령 스트림을 계속 전송해야 하는 더 복잡한 프로세스가 포함된다.

클라이언트는 WAIT 명령을 사용하여 특정 데이터의 동기식 복제를 요청할수 있다. 그러나 다른 레디스 인스턴스에 지정된 수의 승인된 복사본이 있는지만 확인할 수 있고, 레디스 인스턴스 집합을 강력한 일관성을 가진 CP 시스템으로 전환하지는 않는다. 레디스 지속성의 정확한 구성에 따라 페일오버 중에도 승인된 쓰기가 손실될 수 있다. 그러나 WAIT를 사용하면 실패 이벤트 후 쓰기가 손실될 확률이 트리거 하기 어려운 특정 실패모드로 크게 줄어든다.

- 레디스는 비동기(asynchronous) 복제를 한다.
- 마스터는 복제서버를 여러 개 둘 수 있다.
- 복제서버는 또 복제서버를 둘 수 있습니다.마스터 -> 복제1 -> 복제2 이런 구성이 된다.
- 마스터에 많은 데이터가 있는 상태에서 복제서버를 시작하면, 대량의 마스터 데이터가 복제서버로 보내질 것이다. 이 때에도 마스터는 멈추지 않고 정상적으로 요청을 처리한다. 왜냐하면 데이터를 복제서버로 보내는(RDB 파일을 생성하는) 작업은 자식 프로세스가 처리한다.
- 복제서버에게 조회 요청을 처리하도록 하는 것도 부하를 분산하는 좋은 방법이다. 특히 SORT 명령같은 것들은 복제서버에서 수행하는 것이 좋을 것이다.
- 마스터의 부하를 줄이기 위해서, AOF 쓰기나 RDB 파일 생성을 복제서버에서 수행하는 것도 좋은 방법이다. 하지만 이런 설정을 했을 경우에는 마스터 자동 시작 하도록 하면 데이터가 유실 될 수 있다.

Safety of replication when master has persistence turned off

- 서버가 리부트할 때 레디스를 자동 재시작하도록 설정했다면, 마스터에서는 persistence(AOF, RDB)를 사용할 것을 권장한다. Persistence 기능을 사용하지 않는다면 자동 재시작 기능을 이용하지 않는 것이 좋다.
- Persistence 기능을 사용하지 않는 상태에서 자동 재시작하도록 설정했다면, 마스터와 복제서버 모두 데이터를 잃어버릴 수 있다. 아래와 같은 시나리오가 발생할 수 있다
    1. 노드 A를 마스터(persistence 기능 끔)로 하고 노드 B를 복제서버로 설정.
    2. 노드 A가 예외 상항으로 다운되었다 다시 시작했을때, 레디스 마스터 서버가 자동 재시작 했다면, persistence 기능을 꺼 놓았으므로 AOF나 RDB 파일이 없읍니다. 그럼 마스터는 데이터가 없는 빈 상태로 뜰것이다.
    3. 노드 B는 마스터의 아무것도 없는 데이터를 복제하게 되어, 결과적으로 복제서버 데이터도 사라지게 된다. **복제서버에 AOF 파일이 있어도, 초기 복제 후에 AOF Rewrite 기능이 수행되어 복제서버의 AOF 파일에도 데이터는 사라지게 된다.**
- 테이터의 보호는 언제나 중요하기 때문에 자동 재시작 기능을 사용한다면 Persistence를 사용해야한다.