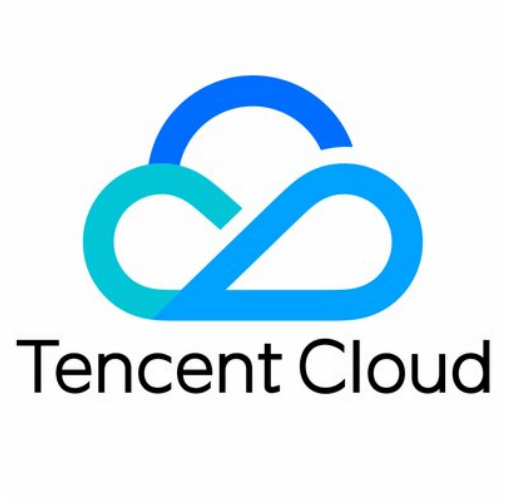
Cloud Load Balancer

OPS 가이드

제품 문서



목록:

OPS 가이드
- 과도한 TIME_WAIT 상태의 클라이언트에 대한 솔루션
- CLB HTTPS 서비스 성능 테스트
- 스트레스 테스트 FAQ
- CLB 인증서 작업 권한

OPS 가이드
과도한 TIME_WAIT 상태의 클라이언트에 대한 솔루션

최종 업데이트 날짜: : 2024-01-04 20:16:37

배경

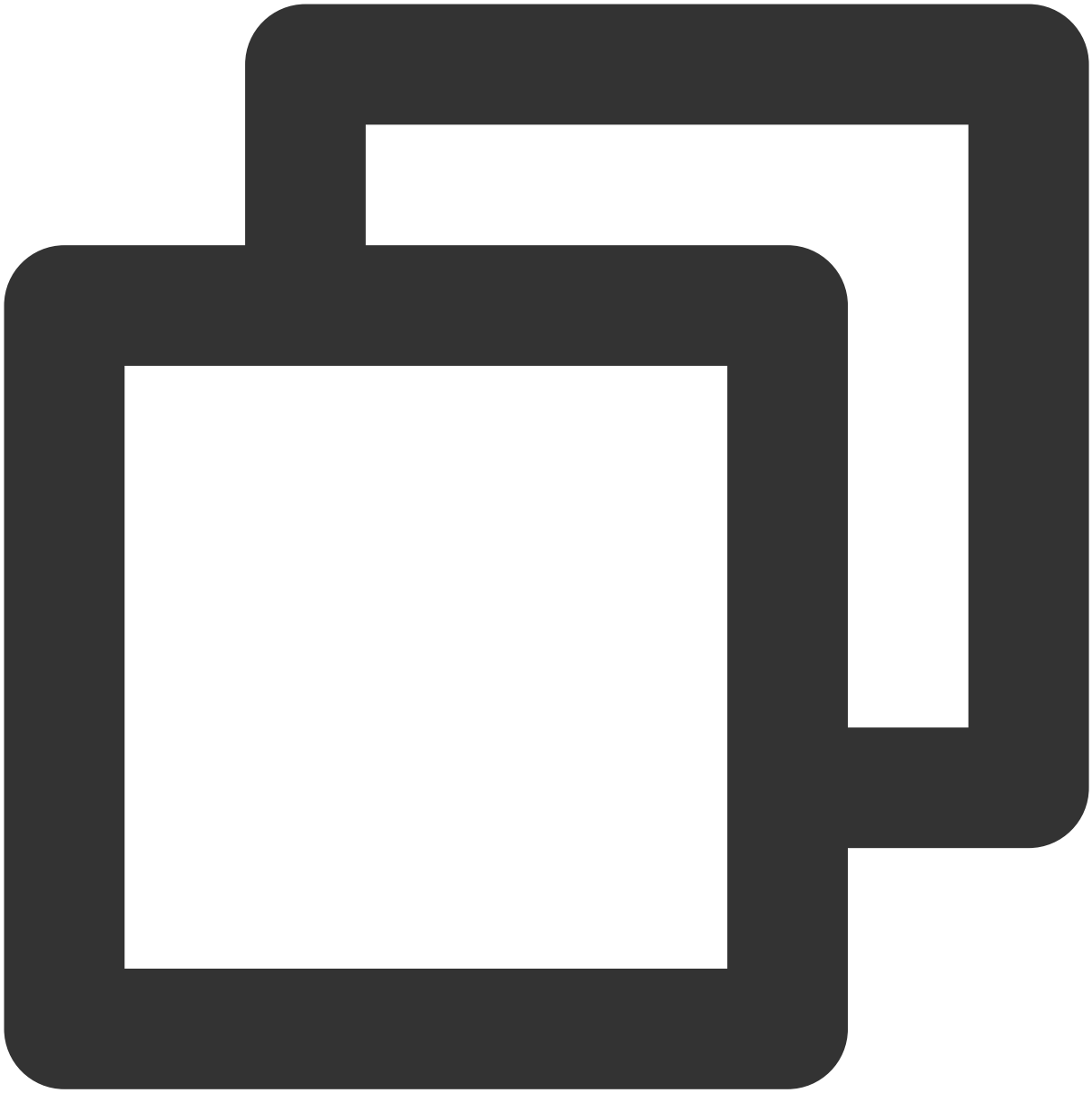
CLB에서 스트레스 테스트를 수행할 때 너무 많은 클라이언트 timewait(모든 포트가 짧은 시간에 점유됨)으로 인해 connect 실패가 발생할 수 있습니다. 원인과 솔루션은 다음과 같습니다.

Linux 매개변수 설명

tcp_timestamps: tcp timestamps 활성화 여부입니다. timestamps는 tcp 3방향 핸드셰이크에서 협의됩니다. 당사자 중 하나가 이 매개변수를 지원하지 않으면 이 연결에서 사용되지 않습니다.
tcp_tw_recycle: tcp time_wait 상태의 재사용 여부입니다.
tcp_tw_reuse: 활성화되면 1s를 초과하는 time_wait 상태의 연결을 직접 재사용할 수 있습니다.

원인 분석

클라이언트가 연결을 사전에 닫기 때문에 클라이언트에 너무 많은 timewait가 있습니다. 클라이언트가 연결을 닫으면 연결이 timewait 상태가 되고 기본적으로 60s 후에 다시 사용됩니다. 이 경우 `tcp_tw_recycle` 및 `tcp_tw_reuse` 매개변수를 활성화하여 timewait 상태에서 연결을 쉽게 재사용할 수 있습니다.
현재 CLB에서 `tcp_timestamps` 가 비활성화되어 있으면 클라이언트에서 활성화한 `tcp_tw_recycle` 및 `tcp_tw_reuse` 매개변수가 적용되지 않으며 timewait 상태의 연결을 빠르게 재사용할 수 없습니다. 다음은 일부 Linux 매개변수와 CLB에서 `tcp_timestamps` 를 활성화할 수 없는 이유를 설명합니다.
1. tcp_tw_recycle 및 tcp_tw_resuse는 tcp_timestamps가 활성화된 경우에만 적용됩니다.
2. FullNAT 시나리오에서는 공중망 클라이언트가 NAT 게이트웨이를 통해 서버에 액세스하지 못할 수 있으므로 tcp_timestamps 및 tcp_tw_recycle을 동시에 활성화할 수 없습니다. 원인은 다음과 같습니다.
tcp_tw_recycle/tcp_timestamps가 모두 활성화된 경우 동일한 원본 IP(동일 서버)의 socket connect 요청에 있는 timestamp는 60s 이내에 증분되어야 합니다. 2.6.32 커널을 예로 들면 세부 사항은 다음과 같습니다.



```
if(tmp_opt.saw_tstamp && tcp_death_row.sysctl_tw_recycle &&
(dst = inet_csk_route_req(sk,req))!= NULL &&
 (peer = rt_get_peer((struct rtable *)dst))!= NULL &&
 peer->v4daddr == saddr){
 if(get_seconds()< peer->tcp_ts_stamp + TCP_PAWS_MSL &&
      (s32)(peer->tcp_ts - req->ts_recent) > TCP_PAWS_WINDOW){
          NET_INC_STATS_BH(sock_net(sk),LINUX_MIB_PAWSPASSIVEREJECTED);
          goto ↓drop_and_release;
 }
}
```

설명：

tmp_opt.saw_tstamp: 이 socket은 tcp_timestamp를 지원합니다.

sysctl_tw_recycle: 이 서버에 대해 tcp_tw_recycle이 활성화되었습니다.

TCP_PAWS_MSL: 60s, 원본 IP의 마지막 TCP 통신이 60s 이내에 발생했습니다.

TCP_PAWS_WINDOW: 1, 원본 IP의 마지막 tcp 통신 timestamp가 이 tcp 통신의 timestamp보다 큽니다.

3. CLB(레이어 7)에서는 공중망 클라이언트가 아래 예시와 같이 NAT 게이트웨이를 통해 서버에 액세스하지 못할 수 있으므로 tcp_timestamps가 비활성화됩니다.

a) 5개는 아직 time_wait 상태입니다. NAT 게이트웨이의 포트 할당 정책에서는 최대 세그먼트 수명(2MSL)의 2배에 동일한 5개를 재사용하고 syn 패킷을 전송합니다.

b) tcp_timestamps가 활성화되고 다음 두 가지 조건이 충족되면 syn 패킷이 삭제됩니다(timestamp 옵션이 활성화되어 패킷이 오래된 것으로 간주되기 때문).

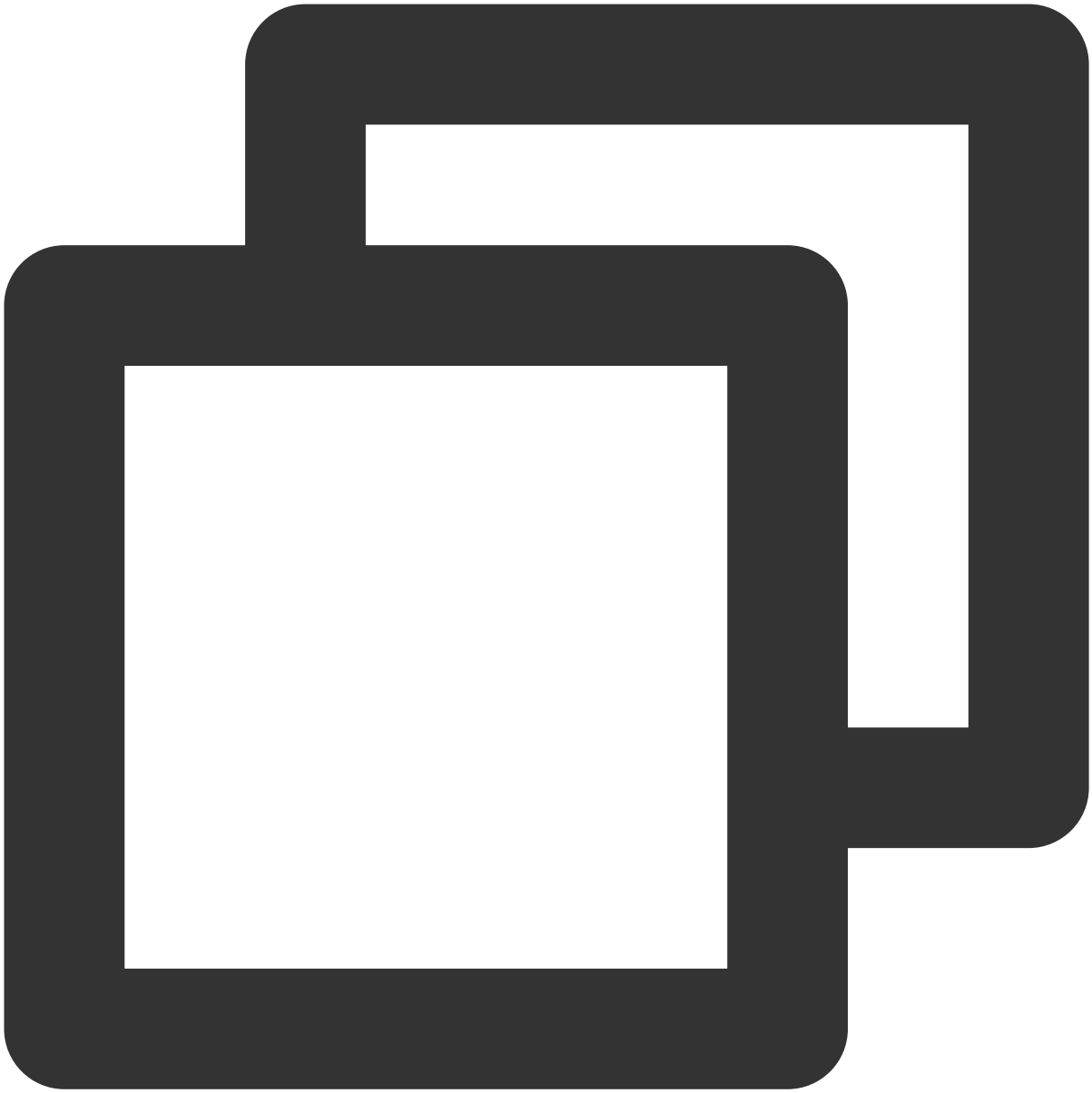
i. 지난 타임스탬프 > 이번 타임스탬프입니다.

ii. 패킷은 24일 이내에 수신됩니다(타임스탬프 필드는 32비트이고 타임스탬프는 Linux에서 기본적으로 1ms당 한 번 업데이트됩니다. 타임스탬프는 24일 후에 래핑됩니다).

설명：

이 문제는 클라이언트가 ISP의 NAT 게이트웨이에서 제한된 공중망 IP를 공유하고 2MSL에서 5개를 재사용할 수 있기 때문에 모바일 장치에서 더 분명합니다. 다른 클라이언트에서 보낸 타임스탬프는 증가하지 않을 수 있습니다.

2.6.32 커널을 예로 들면 세부 사항은 다음과 같습니다.



```
static inline int tcp_paws_check(const struct tcp_options_received *rx_opt,int paws
{
    if((s32)(rx_opt->ts_recent - rx_opt->rcv_tsval)<= paws_win)
    return 1;
    if(unlikely(get_seconds()>=rx_opt->ts_recent_stamp + TCP_PAWS_24DAYS))
    return 1;
    return 0;
}
```

설명：

rx_opt->ts_recent: 마지막 타임스탬프입니다.
rx_opt->rcv_tsval: 이번에 수신된 타임스탬프입니다.
get_seconds(): 현재 시간입니다.
rx_opt->ts_recent_stamp: 이전 패킷을 수신한 시간입니다.

솔루션

클라이언트에 너무 많은 Timewait가 있는 경우 아래 솔루션을 참고하십시오.
HTTP는 비지속 연결을 사용합니다(Connection: close). 이 경우 CLB는 사전에 연결을 닫고 클라이언트는 timewait를 생성하지 않습니다.
시나리오에 지속 연결이 필요한 경우 socket의 SO_LINGER 옵션을 활성화하고 rst를 사용하여 연결을 닫아 timewait 상태를 피하고 빠른 포트 재사용을 달성합니다.

CLB HTTPS 서비스 성능 테스트

최종 업데이트 날짜: : 2024-01-04 20:16:50

1. CLB의 HTTPS 기능

프로토콜 스택과 서버를 최적화함으로써 Tencent Cloud CLB는 HTTPS 성능을 크게 향상시킵니다. 한편, Tencent Cloud는 국제 협력을 통해 인증서 비용을 대폭 절감합니다. Tencent CLB는 다음과 같은 측면에서 귀하의 비즈니스에 도움이 될 수 있습니다.
1. HTTPS 사용은 Client의 액세스 속도에 영향을 미치지 않습니다.
2. 클러스터 내 단일 서버의 SSL 암호화 및 복호화 성능은 고성능 CPU보다 최소 3.5배 높은 초당 최대 6.5W cps의 전체 핸드셰이크를 지원합니다. 이를 통해 서버 비용을 절감하고 비즈니스 운영 및 트래픽 급증 시 서비스 기능을 크게 향상시키며 공격 방지 기능을 강화합니다.
3. 여러 프로토콜의 오프로딩 및 변환이 지원되어 다양한 클라이언트 프로토콜에 적응하는 비즈니스의 스트레스를 줄입니다. 비즈니스 백엔드는 HTTP2, SPDY, SSL 3.0 및 TLS 1.2와 같은 다른 버전의 프로토콜을 사용하기 위해 HTTP1.1만 지원하면 됩니다.
4. 원스톱 SSL 인증서 적용, 모니터링, 교체를 지원합니다. Tencent Cloud는 국제 인증 기관인 comodo 및 symantec과 협력하여 인증서 신청 프로세스를 간소화하고 비용을 절감합니다.
5. Anti-CC 및 WAF 기능을 제공하여 느린 HTTP 공격, 고빈도 표적 공격, SQL 인젝션 및 웹 사이트 트로이 목마와 같은 애플리케이션 레이어 공격을 효과적으로 제거합니다.

2. 테스트 목적

HTTPS 서비스는 본인 인증, 정보 암호화, 무결성 검증 등의 장점이 있습니다. 그러나 SSL 프로토콜을 사용하여 보안 통신을 구현하면 암호화 및 암호 해독에 의한 대기 시간 증가 및 CPU 리소스 소비를 포함하여 특정 성능 손실이 발생합니다. 본문은 SSL 암호화 및 복호화 시 Tencent Cloud HTTPS 서비스의 극한 성능 테스트 데이터가 포함되어 있습니다. 기존 HTTPS 성능 데이터와 비교할 수 있습니다.

3. 테스트 환경

스트레스 테스트 툴: wrk 4.0.2

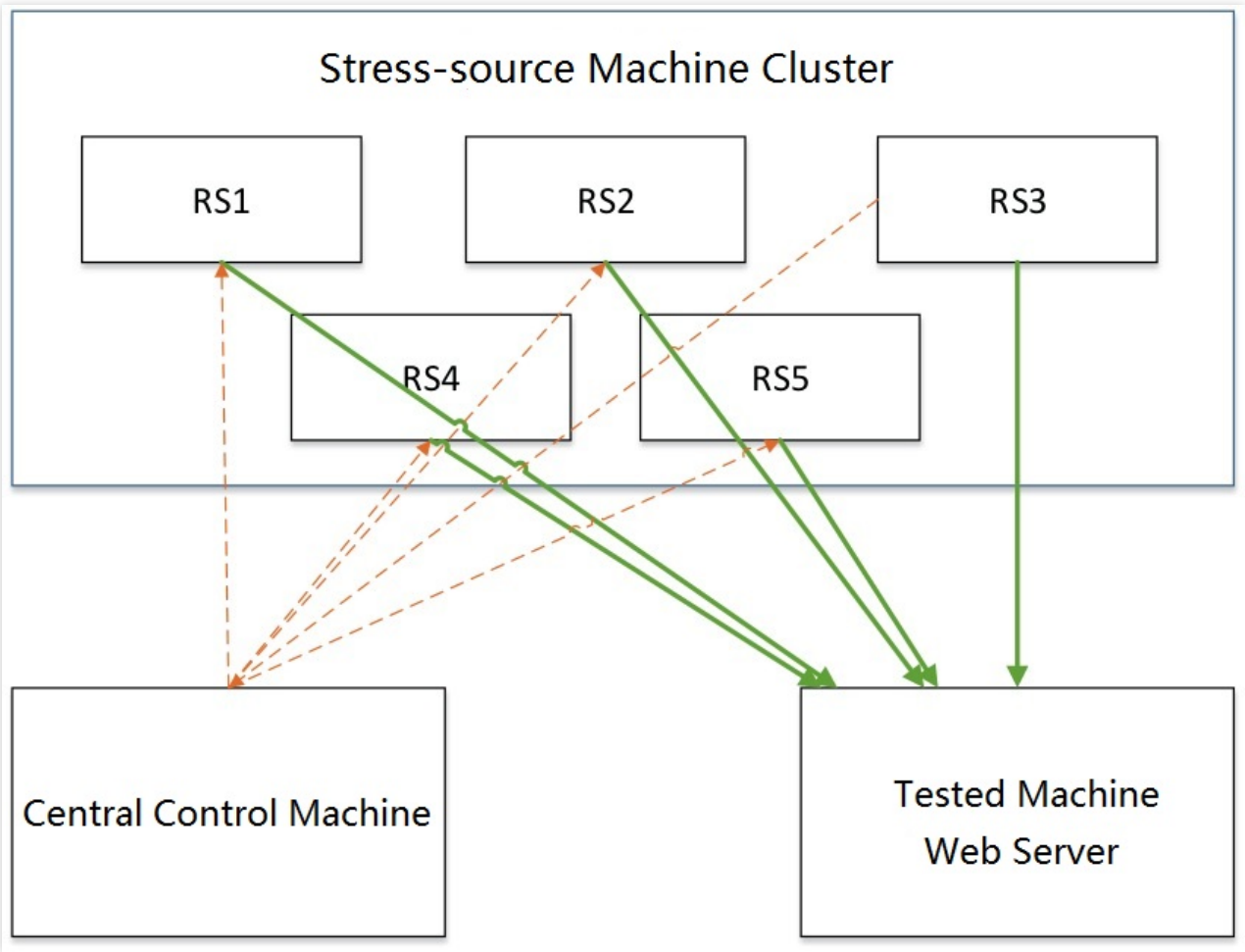
Tencent Cloud의 기본 서비스 환경: Nginx 1.1.6_1.9.9 + Openssl 1.0.2h

– Nginx가 설치된 OS 정보: inux TENCENT64.site 3.10.94-1-tlinux2-0036.tl2 #1 SMP Thu Jan 21 03:40:59 CST 2016 x86_64 x86_64 x86_64 GNU/Linux

다른 스트레스 서버의 OS: Linux TENCENT64.site 2.6.32.43-tlinux-1.0.17-default #1 SMP Tue Nov 17 18:03:12 CST 2015 x86_64 x86_64 x86_64 GNU/Linux

4. WebServer 클러스터 테스트 방식

단일 서버의 스트레스는 Tencent Cloud의 https 서비스의 성능 제한을 테스트하기에 충분하지 않습니다. 여러 스트레스 서버가 필요합니다. 테스트에는 세 부분이 포함됩니다.
1. 스트레스 테스트 클러스터. http/https 스트레스를 분산시키고 단일 스트레스 서버의 스트레스 테스트 결과를 출력하는데 사용됩니다.
2. 스트레스 테스트 클러스터의 시작과 끝을 동기적으로 제어하는 중앙 제어 서버는 각 스트레스 서버에서 테스트 데이터를 가져와 데이터를 집계하여 출력합니다.
3. 웹 서버는 Tencent Cloud의 HTTPS 서비스를 호스팅하는 CVM 인스턴스입니다. WebServer 성능이 테스트되면 upstream 연결 없이 페이지를 직접 반환할 수 있습니다.
연결은 다음과 같습니다.



5. HTTPS WebServer의 성능 테스트 데이터

연결 유형	Session cache	패킷 크기(bytes)	암호화 제품군	성능(qps)
지속	On	230	ECDHE-RSA-AES128-GCM-SHA256	296241
비지속	Off	230	ECDHE-RSA-AES128-GCM-SHA256	65630

6. CLB HTTPS 성능 테스트 결론

상기와 같이 Tencent Cloud의 HTTPS 서비스는 SSL 암호화 및 복호화를 지원합니다. 백엔드에 여러 서버 클러스터가 있으며 클러스터의 단일 서버는 전체 핸드셰이크 중에 최대 65000 qps, 지속 연결 중에 약 300000 qps의 성능을 달성할 수 있습니다.
정상적인 상황에서 HTTPS 프로토콜은 SSL 프로토콜을 사용할 때 하나 이상의 전체 핸드셰이크 프로세스를 추가하고 대기 시간은 2 × 왕복 시간(RTT)만큼 증가합니다. 또한 SSL 대칭/비대칭 암호화는 많은 양의 CPU 리소스를 사용합니다. RSA의 암호 해독 기능은 HTTPS 기반 액세스의 주요 장애물입니다.
Tencent Cloud의 HTTPS 서비스를 사용하면 SSL 암호화 및 복호화를 위한 추가 서비스를 배포할 필요가 없습니다. 추가 요금이 부과되지 않는 이 서비스를 통해 강력한 비즈니스 호스팅 및 공격 방지 기능을 누릴 수 있습니다.

스트레스 테스트 FAQ

최종 업데이트 날짜: : 2024-01-04 20:17:07

스트레스 테스트에 대한 고객 경험을 바탕으로 본문은 스트레스 테스트의 일반적인 성능 문제를 요약하고 문제 해결 솔루션과 제안을 제공합니다.

스트레스 테스트 FAQ

리얼 서버에서 공중망 액세스가 활성화되어 있지 않음

CVM 구매 시 공중망 액세스가 활성화되지 않은 경우 공중망 CLB가 CVM 인스턴스에 마운트될 때 포워딩이 실패할 수 있습니다.

리얼 서버의 대역폭 부족

리얼 서버의 대역폭이 낮으면 임계값을 초과하면 패킷을 CLB로 반환할 수 없습니다. CLB는 클라이언트에 504 또는 502 오류를 반환합니다.

클라이언트 포트 부족

클라이언트 수가 너무 적거나 클라이언트 포트 범위가 너무 좁으면 클라이언트 포트가 부족하여 연결이 설정되지 않습니다. 또한 지속 연결이 설정되었을 때 keep_alive 값이 0보다 크면 연결이 지속적으로 포트를 사용하므로 사용 가능한 클라이언트 포트 수가 줄어듭니다.

리얼 서버에 의존하는 애플리케이션에 성능 문제 발생

요청이 CLB를 통해 리얼 서버에 도달한 후 리얼 서버의 로드는 정상입니다. 그러나 리얼 서버의 애플리케이션은 데이터베이스와 같은 다른 애플리케이션에도 의존하기 때문에 데이터베이스의 성능 문제도 스트레스 테스트 성능에 영향을 줄 수 있습니다.

리얼 서버 비정상

리얼 서버의 상태는 스트레스 테스트에서 무시될 수 있습니다. 리얼 서버에 상태 확인 실패 또는 상태 확인 상태가 불안정한 경우(빠른 변경으로 인해 좋은 경우도 있고 나쁜 경우도 있음) 스트레스 테스트의 성능이 저하될 수 있습니다.

CLB에 대해 세션 지속성을 활성화하면 리얼 서버 간에 트래픽이 고르지 않게 분산됨

CLB에 대해 세션 지속성이 활성화된 후 요청이 고정된 리얼 서버에 배포될 수 있습니다. 트래픽 분포가 고르지 않게 되어 스트레스 테스트의 성능에 영향을 줍니다. 스트레스 테스트 중에는 세션 지속성을 비활성화하는 것이 좋습니다.

스트레스 테스트를 위한 제안

주의 :

다음 구성은 CLB 스트레스 테스트에만 사용됩니다. 프로덕션 환경에 필요하지 않습니다.

CLB의 포워딩 기능을 스트레스 테스트할 때 비지속 연결을 사용하는 것이 좋습니다.

세션 지속성 기능에 대한 검증을 제외하고 스트레스 테스트는 일반적으로 CLB의 포워딩 기능을 검증하기 위해 설계됩니다. 따라서 비지속 연결을 사용하여 CLB와 리얼 서버의 처리 능력을 테스트할 수 있습니다.

대역폭 상한 및 지속 연결 서비스와 같은 CLB의 처리량 테스트를 스트레스 테스트하기 위해 지속 연결을 사용하는 것이 좋습니다.

스트레스 테스트 툴의 초과 시간을 작은 값으로 조정하는 것이 좋습니다. 그렇지 않으면 타임아웃 기간이 증가할 때 평균 응답 시간이 증가하여 스트레스 수준에 도달했는지 여부를 신속하게 판단할 수 없습니다.

I/O, DB 등의 애플리케이션 로직으로 인한 손실을 방지하기 위해 리얼 서버에서 제공하는 정적 웹사이트를 스트레스 테스트용으로 사용할 것을 권장합니다.

리스너에 대한 세션 지속성을 비활성화합니다. 그렇지 않으면 스트레스가 특정 리얼 서버에 집중됩니다. 스트레스 성능이 만족스럽지 않은 경우 CLB에서 리얼 서버의 모니터링 데이터를 확인하여 트래픽이 고르게 분산되었는지 확인할 수 있습니다.

상태 확인 중에 생성되는 리얼 서버에 대한 액세스 요청을 줄이기 위해 리스너에 대한 상태 확인을 비활성화합니다.

스트레스 테스트를 위해 여러 client(> 5)를 사용합니다. 분산된 원본 IP는 실제 온라인 조건을 더 잘 시뮬레이션할 수 있습니다.

CLB 인증서 작업 권한

최종 업데이트 날짜: : 2024-01-04 20:17:29

작업 시나리오

2020년 3월 23일부터 CLB의 모든 인증서 작업은 인증을 위해 CAM(Cloud Access Management)에 연결되었습니다. 따라서 서브 사용자 계정이 CLB 인증서 작업을 수행할 때 '이 작업에 대한 권한이 없습니다. 개발자에게 문의하세요.'가 표시되면 아래 지침에 따라 서브 사용자 계정에 인증서 권한을 부여할 수 있습니다.

전제 조건

로그인한 계정은 루트 계정이거나 CAM 권한이 있는 서브 사용자 계정이어야 합니다(즉, QcloudCamFullAccess 정책과 연결됨).

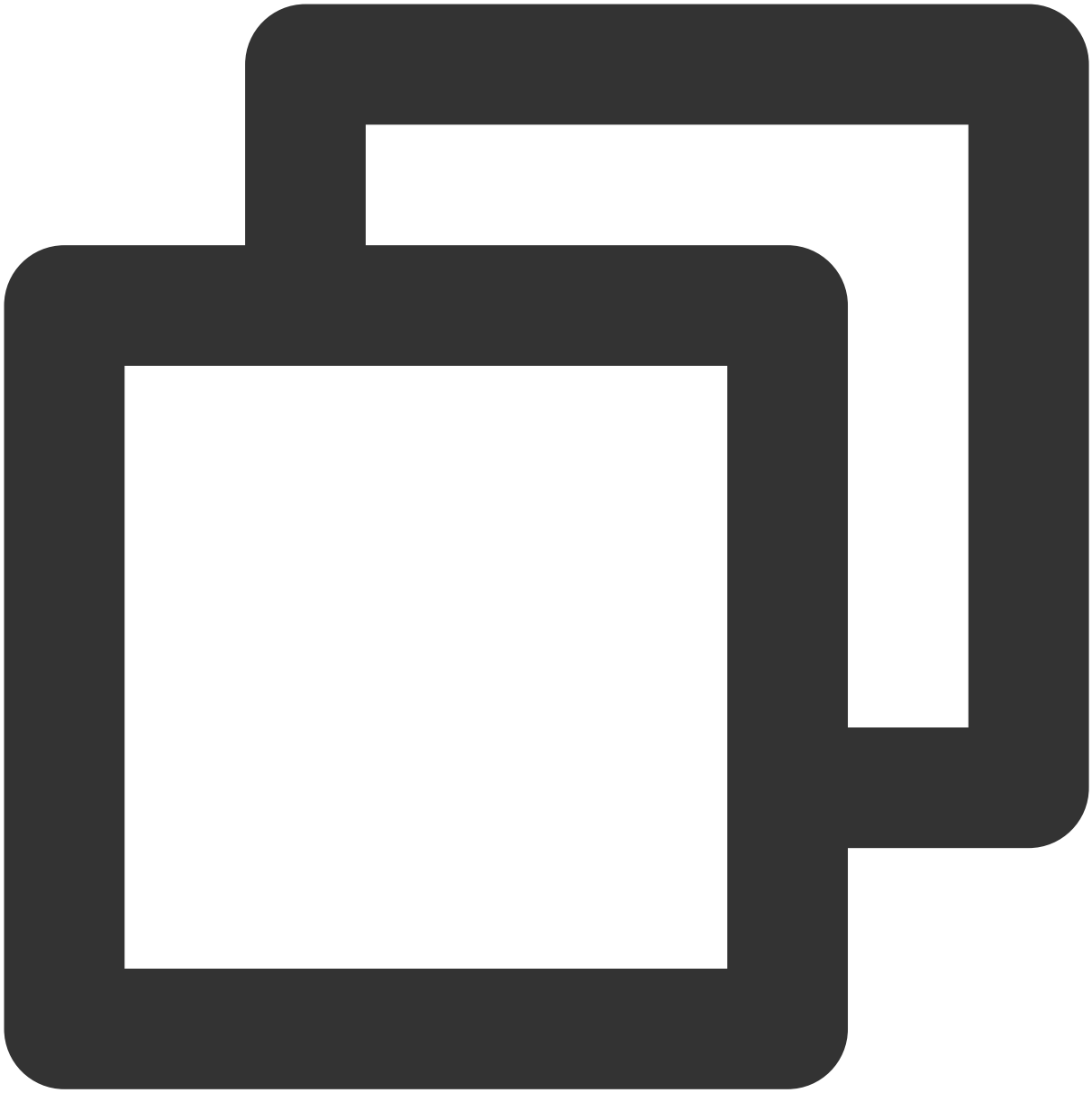
> 서브 사용자 계정에 CAM 권한이 있는지 확인하려면 CAM 콘솔의 사용자 목록으로 이동하여 서브 사용자의 세부 정보 페이지로 이동하여 QcloudCamFullAccess 정책이 연결되었는지 확인합니다.
> QcloudCamFullAccess 정책이 연결되어 있지만 서브 사용자가 인증서 작업을 수행할 때 'API 권한(message:GetReceiversOnAllType)이 없습니다. 개발자에게 문의하세요.'가 표시되면 무시하고 계속 진행할 수 있습니다.

작업 단계

다음 방법으로 인증서 권한을 부여하십시오.

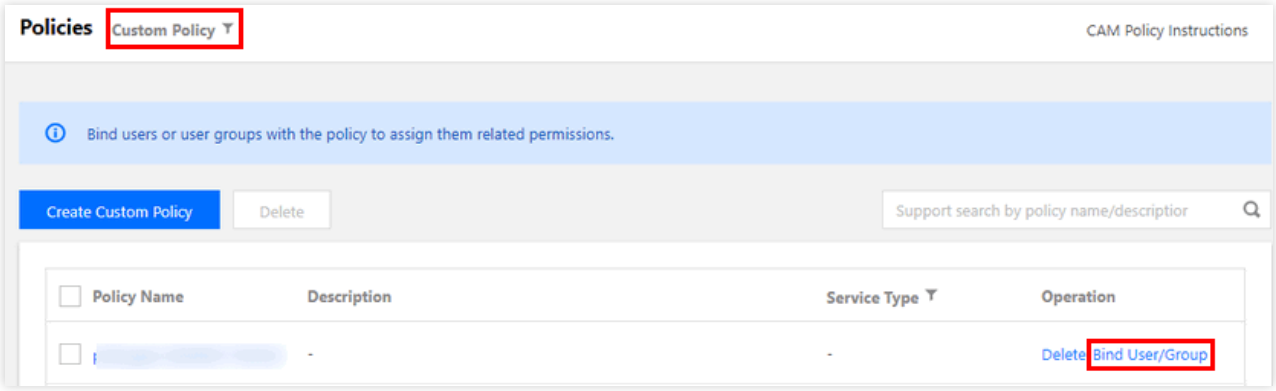
방법1: 사용자 지정 정책 연결

1. CAM 콘솔에 로그인합니다.
2. 왼쪽 사이드바에서 [정책]을 클릭합니다.
3. [사용자 지정 정책 생성]을 클릭하고 팝업 창에서 [정책 구문으로 생성]을 선택합니다.
4. '템플릿 정책 선택' 페이지에서 [빈 템플릿]을 선택하고 [다음]을 클릭합니다.
5. '정책 편집' 페이지에서 정책 이름을 입력하고 '정책 내용 편집' 입력 상자에 다음 정책 내용을 입력합니다.

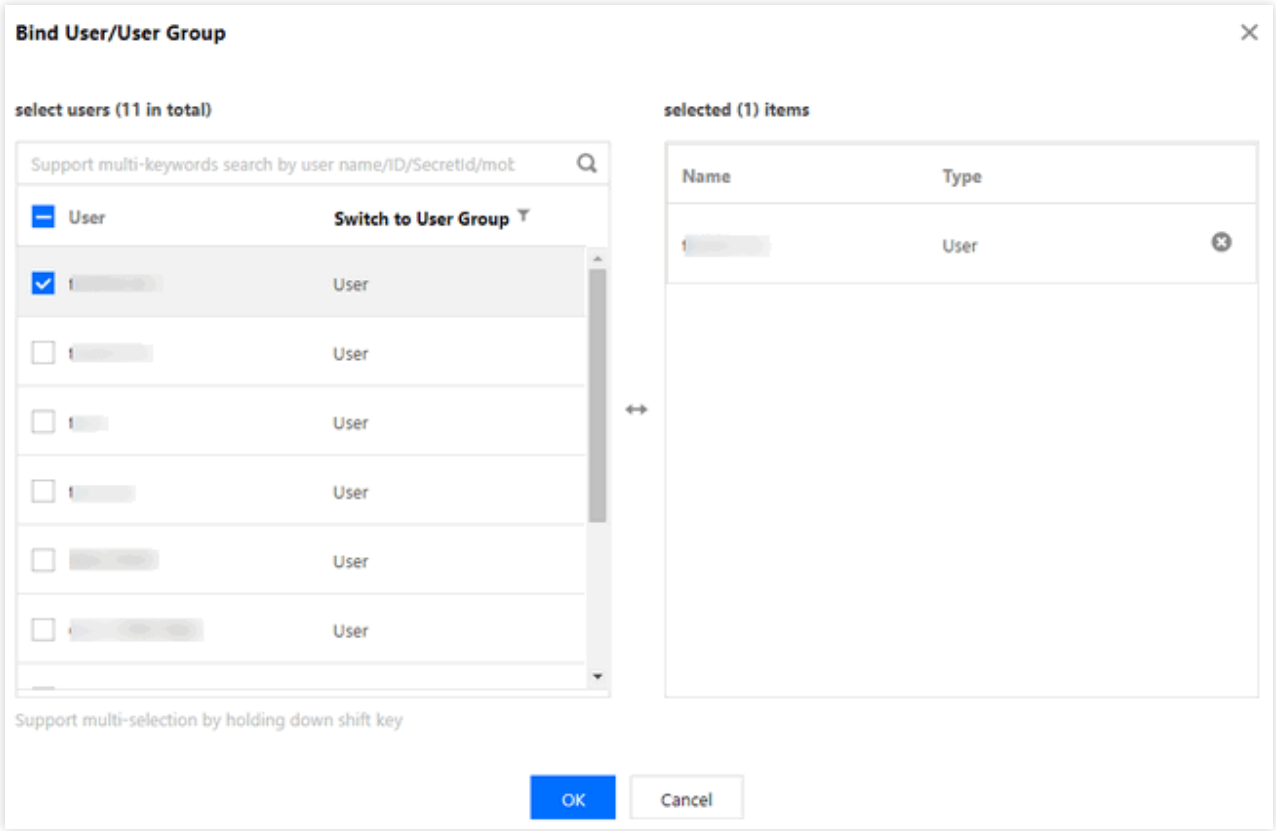


```
{
 "version": "2.0",
 "statement": [
     {
         "action": "name/ssl:*",
         "resource": "qcs::ssl:::*",
         "effect": "allow"
     }
 ]
}
```

6. 그런 다음 [정책 생성]을 클릭하여 '정책' 목록 페이지로 돌아갑니다.
7. '정책' 목록 페이지 상단에서 [사용자 지정 정책]을 선택하고 목록에서 방금 생성한 정책 행을 찾은 다음 작업 열에서 [사용자/그룹 연결]을 클릭합니다.



8. 팝업 창에서 권한을 부여할 사용자를 선택하고 [확인]을 클릭합니다.



방법2: 사전 설정 정책 연결

1. CAM 콘솔에 로그인합니다.
2. 왼쪽 사이드바에서 [사용자]>[사용자 목록]을 선택하여 '사용자 목록' 페이지로 이동합니다.

3. 권한을 부여할 서브 사용자 행의 작업 표시줄에서 [권한 부여]를 클릭합니다.
4. 팝업 창에서 QcloudSSLFullAccess(SSL 인증서(SSL) 전체 읽기/쓰기 액세스 권한) 또는 QcloudSSLReadOnlyAccess(SSL 인증서(SSL) 읽기 전용 액세스 권한)를 선택하고 [확인]을 클릭합니다.