

Einfacher Echtzeit-Server (SRS) für RTMP-Streaming

Lass uns in die Nutzung des Simple Realtime Server (SRS) eintauchen, um einen Edge-Server für RTMP-Streaming einzurichten, mit einem Fokus auf das Verständnis und die effektive Konfiguration, einschließlich der `rtmp.conf`-Datei. SRS ist ein leistungsstarker, hochleistungsfähiger Medienserver, der für Echtzeit-Streaming-Protokolle wie RTMP, HLS und HTTP-FLV entwickelt wurde. Er eignet sich besonders für Szenarien, die Skalierbarkeit erfordern, wie Content Delivery Networks (CDNs), bei denen ein Edge-Server Streams effizient an viele Clients verteilen kann, während er Inhalte von einem Origin-Server abrufen. Ich werde dich Schritt für Schritt durch den Prozess führen, das Konzept der Edge-Deployment erklären und verdeutlichen, wie man mit Konfigurationen wie `rtmp.conf` arbeitet.

Was ist SRS und Edge Deployment?

SRS ist ein Open-Source-Server, der für die Handhabung von Echtzeit-Media-Streaming entwickelt wurde, mit einem Fokus auf Einfachheit und Effizienz. Er unterstützt RTMP (Real-Time Messaging Protocol) für Echtzeit-Live-Streaming sowie andere Protokolle wie HLS und WebRTC. In SRS fungiert ein "Edge"-Server als Vermittler zwischen Clients (Zuschauern oder Veröfentlichern) und einem "Origin"-Server (wo der Stream seinen Ursprung hat). Der Edge holt sich Streams nur dann vom Origin, wenn sie von Clients angefordert werden, wodurch die Last auf dem Origin reduziert und eine skalierbare Verteilung ermöglicht wird—denke daran wie eine Caching-Schicht eines CDNs, die für Live-Streams angepasst ist.

Das Einsetzen eines Edge-Servers mit SRS ist ideal, wenn du: - Eine große Anzahl von Zuschauern bedienen möchtest, ohne den Origin zu überlasten. - Verlegern ermöglichen möchtest, Streams an den Edge zu senden, der sie dann an den Origin weiterleitet. - Die Bandbreitennutzung auf teuren Origin-Servern minimieren möchtest, indem du kostengünstigere Edge-Knoten nutzt.

Schritt-für-Schritt: Einsetzen eines Edge-Servers mit SRS für RTMP

Hier ist, wie du SRS als Edge-Server für RTMP-Streaming einrichten kannst. Ich gehe davon aus, dass du auf einem Linux-System (z.B. Ubuntu) arbeitest, da SRS für solche Umgebungen optimiert ist.

1. Installiere SRS Zuerst musst du SRS auf deinem System zum Laufen bringen: - **Lade SRS herunter:** Hole die neueste stabile Version vom offiziellen GitHub-Repository (github.com/ossrs/srs). Stand heute, dem 26. Februar 2025, würdest du das Repo normalerweise klonen: `git clone https://github.com/ossrs/srs.git` `cd srs` - **Bau SRS:** SRS verwendet einen einfachen Build-Prozess mit `./configure` und `make`: `./configure` `make` Dies kompiliert den Server in das `objs`-Verzeichnis (z.B. `objs/srs`). - **Teste die Binärdatei:** Führe sie mit der Standardkonfiguration aus, um sicherzustellen, dass sie funktioniert: `./objs/srs -c conf/srs.conf` Standardmäßig hört es auf Port 1935 für RTMP. Überprüfe die Konsolenausgabe zur Bestätigung.

2. Verstehe das Edge-Konzept In SRS arbeitet ein Edge-Server im “remote”-Modus, was bedeutet, dass er keine Streams selbst erzeugt, sondern sie vom Origin-Server abrufen, wenn ein Client sie anfordert (für die Wiedergabe) oder Streams an den Origin sendet (für die Veröffentlichung). Dieses bedarfsorientierte Abrufen macht Edge-Server effizient für die Skalierung der RTMP-Zustellung.

- **Origin-Server:** Die Quelle des Streams (z.B. wo ein Encoder wie OBS einen RTMP-Stream sendet).
- **Edge-Server:** Ein Relais, mit dem sich Clients verbinden, das vom Origin nur dann abrufen, wenn es benötigt wird.

Für dieses Beispiel nehmen wir an, dass du bereits einen Origin-Server mit SRS unter 192.168.1.100:1935 laufen hast (ersetze dies durch deine tatsächliche Origin-IP).

3. Konfiguriere den Edge-Server SRS verwendet Konfigurationsdateien, um sein Verhalten zu definieren. Die Standard-srs.conf ist ein guter Ausgangspunkt, aber für die Edge-Deployment erstellst du eine spezifische Konfiguration—nennen wir sie edge.conf. Hier ist, wie du sie einrichten kannst:

- **Erstelle edge.conf:**

```
cd conf
nano edge.conf
```

- **Füge Edge-Konfiguration hinzu:** Hier ist eine minimale edge.conf für die RTMP-Edge-Deployment:

```
listen          1935;
max_connections 1000;
srs_log_tank     file;
srs_log_file     ./objs/edge.log;
vhost __defaultVhost__ {
    cluster {
        mode      remote;
        origin     192.168.1.100:1935;
    }
}
```

- listen 1935: Der Edge hört auf RTMP-Verbindungen auf Port 1935.
- max_connections 1000: Begrenzung der gleichzeitigen Verbindungen (anpassen basierend auf der Kapazität deines Servers).
- srs_log_file: Protokollierung in eine Datei für das Debugging.
- vhost __defaultVhost__: Die Standard-Virtual-Host-Konfiguration.
- cluster { mode remote; origin 192.168.1.100:1935; }: Setzt diesen Server als Edge (mode remote) und zeigt auf den Origin-Server.

- **Speichern und Beenden:** Strg+O, Enter, Strg+X in nano.

4. Starte den Edge-Server Führe SRS mit deiner Edge-Konfiguration aus:

```
./objs/srs -c conf/edge.conf
```

Überprüfe die Protokolle (./objs/edge.log), um zu bestätigen, dass es läuft und mit dem Origin verbunden ist.

5. Teste die Einrichtung

- **Veröffentliche einen Stream:** Verwende ein Tool wie OBS oder FFmpeg, um einen RTMP-Stream an den Origin-Server zu senden:

```
ffmpeg -re -i input.mp4 -c copy -f flv rtmp://192.168.1.100/live/livestream
```

Hier ist `live` der App-Name und `livestream` der Stream-Schlüssel.

- **Spiele vom Edge ab:** Verwende VLC oder einen anderen RTMP-Client, um den Stream vom Edge abzuspielen:

```
rtmp://<edge-server-ip>/live/livestream
```

Ersetze `<edge-server-ip>` durch die IP deines Edge-Servers (z.B. 192.168.1.101). Der Edge holt sich den Stream vom Origin und dient ihn dir.

6. Erkunde rtmp.conf SRS liefert standardmäßig keine `rtmp.conf`-Datei, aber du könntest in Tutorials oder benutzerdefinierten Setups darauf stoßen. Es handelt sich im Wesentlichen um eine Namenskonvention für eine RTMP-spezifische Konfigurationsdatei. Zum Beispiel bietet die SRS-Dokumentation (ossrs.net) eine Beispiel-`rtmp.conf` für Echtzeit-RTMP-Streaming:

```
listen                1935;
max_connections       1000;
vhost __defaultVhost__ {
    tcp_nodelay        on;
    min_latency         on;
    play {
        gop_cache      off;
        queue_length 10;
    }
    publish {
        mr              off;
    }
}
```

- **Zweck:** Diese Konfiguration optimiert für Echtzeit-RTMP-Streaming auf einem Origin-Server, nicht auf einem Edge. Für die Edge-Deployment würdest du sie anpassen, indem du den `cluster`-Block aus Schritt 3 hinzufügst.
- **Wichtige Einstellungen:**
 - `tcp_nodelay on`: Verringert die Latenz, indem der Nagle-Algorithmus deaktiviert wird.
 - `min_latency on`: Priorisiert niedrige Latenz gegenüber Puffern.
 - `gop_cache off`: Deaktiviert das Caching von Group of Pictures für Echtzeit-Wiedergabe.
 - `mr off`: Deaktiviert "merge read", um Verzögerungen beim Veröffentlichen zu vermeiden.

Für einen Edge würdest du dies mit den `cluster`-Einstellungen kombinieren, anstatt es eigenständig zu verwenden.

Mehr Erklärungen: Edge-Mechanik und RTMP

- **Wie Edge funktioniert:** Wenn ein Client `rtmp://<edge-ip>/live/livestream` anfordert, überprüft der Edge, ob er den Stream hat. Wenn nicht, holt er ihn sich vom Origin (`192.168.1.100:1935`) und speichert ihn lokal, um ihn anderen Clients zu dienen. Wenn ein Verleger an den Edge sendet, leitet er den Stream an den Origin weiter.
- **RTMP-Spezifika:** RTMP ist ein Protokoll mit niedriger Latenz, das sich ideal für Live-Streaming eignet. SRS verarbeitet RTMP effizient, unterstützt Funktionen wie absolute Zeitcodierung (ATC) für die Synchronisation über Server hinweg, obwohl es standardmäßig im Edge-Modus deaktiviert ist, es sei denn, es wird angegeben.
- **Skalierbarkeit:** Füge mehrere Edges hinzu, die auf denselben Origin zeigen, um Tausende von Clients zu bewältigen. SRS unterstützt Failover, indem mehrere Origins aufgelistet werden (z.B. `origin 192.168.1.100:1935 192.168.1.200:1935;`).

Tipps und Fehlerbehebung

- **Firewall:** Stelle sicher, dass Port 1935 auf beiden Origin- und Edge-Servern geöffnet ist.
- **Protokolle:** Überprüfe `edge.log` auf Fehler wie Verbindungsfehler zum Origin.
- **Latenz:** Edge fügt minimale Latenz hinzu (typischerweise $<1s$), wenn der Origin ebenfalls niedrige Latenz hat.
- **Mehrere Edges:** Setze zusätzliche Edges mit der gleichen Konfiguration ein, passe `listen`-Ports oder IPs nach Bedarf an.

Zusammenfassung

Das Einsetzen eines SRS-Edge-Servers für RTMP ist einfach, sobald du das Verhältnis zwischen Origin und Edge verstehst. Die `edge.conf` richtet den Edge so ein, dass er Streams dynamisch abruft oder sendet, während eine `rtmp.conf`-ähnliche Konfiguration die RTMP-Leistung weiter optimieren könnte, wenn nötig.

Mit dieser Einrichtung bist du bereit, Live-Streaming effizient zu skalieren—ob für eine Handvoll Zuschauer oder ein globales Publikum. Möchtest du es weiter anpassen oder HLS neben RTMP integrieren? Lass es mich wissen!