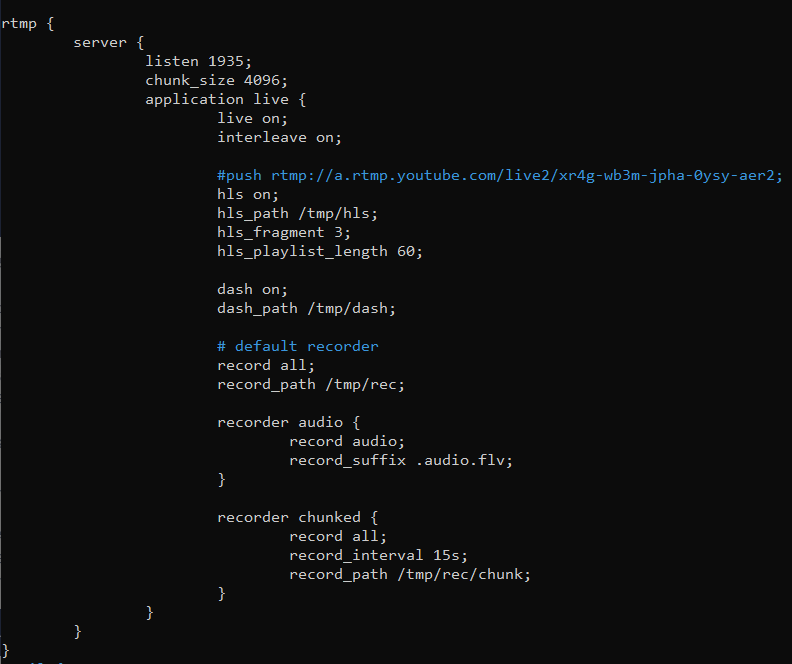
**NGINX AS STREAMING AND STORAGE**

1. **TEST System information:**Using virtual box to create a Virtual machine with these information:
   1. Ubuntu 64bit, version 21.10
   2. CPU: Intel(R) Core(TM) i5-9400 CPU @ 2.90GHz
   3. Memory: 2048MB
   4. Storage: 10GB (from a physical SSD)
   5. Network Adapter: Intel PRO/1000 MT Desktop (Bridged Adapter)
2. **Base Installation (step by step):**
   1. Get newest update: *sudo apt-get update*
   2. Install nginx: *sudo apt install nginx*
   3. Install Nginx RTMP module: *sudo apt install libnginx-mod-rtmp*
   4. Update *nginx.conf* to setup RTMP (default path: */etc/nginx/nginx.conf*)   
        
        
        
      RTMP using default port ***1935***. We setup an application named “*live*” with these main settings:
      1. *live on*: enable live stream
      2. *hls on*: enable hls for modern streams for browsers (optional). These option will save the stream in files as part of the stream to use for access from http/https on browser
      3. *record all*: enable the record feature. The file will be record automatically when a stream is active and saved to “*record\_path*”. The record path folder need to have the permission for writing file.   
         We can set the pattern for file name here with prefix option.
      4. Please go here for more details and for the others settings: <https://github.com/arut/nginx-rtmp-module/wiki/Directives>
   5. Start/reload nginx service: sudo systemctl reload nginx
   6. Install ffmpeg: sudo apt install ffmpeg
   7. Prepare bash file for playing a stream source: create stream.sh file  
      *ffmpeg -re -i rtsp://admin:HSx8fXqmSh9gCmmS@192.168.110.59:554/onvif1 -vcodec copy -loop -1 -c:a aac -b:a 160k -ar 44100 -strict -2 -f flv rtmp://192.168.110.52/live/bbb*By using *ffmpeg* to run a stream resource:
      1. Input (-i option): *rtsp://admin:HSx8fXqmSh9gCmmS@192.168.110.59:554/onvif1*
      2. Output: using rtmp application named “live” which is setup above and the name of the stream as “bbb”. We will have the full stream URI as output like this: *rtmp://192.168.110.52/live/bbb*This name will be use as name for the record file (will be added the set prefix if it has on the record setting).
      3. Other options description can be viewed here: <https://ffmpeg.org/ffmpeg.html#toc-Main-options>
   8. Run the bash file for start streaming: sudo ./stream.sh   
      With this, the stream will be run and the recorded file will be saved to the “record\_path” (/tmp/rec).   
      We can create each bash file for each stream and run them at the same time. I use 1 ssh session for 1 stream. We may have other better option if we embed them into an application code (need more research).
3. **Performance:**With the current TEST environment. I can 5 streams using same resource at the same time without any issue.   
   I still haven’t applied any stress test. Will need to research more to setup and run a stress test.
4. **Further analyze:** 
   1. Those above only a simple system for using nginx server as a livestream proxy and storage. We need to apply load balancing so we can have multiple worker for streaming and recording.
   2. For using a server as a video storage, we may need a complete solution for storing.
   3. Security is a must.   
      - To run a bash and save file to folder from our current meeting app, we should apply a token system or kind of authentication.   
      - Other option is we can build a standalone system for recording, but this system can have the permission to call API for getting the resource stream (RTMP, RTSP, etc…). I tested with calling RTMP from youtube, but it doesn’t work. RTMP from youtube only can use for pushing a stream to live youtube stream.
   4. Pexip:   
      - Documentation from Pexip for streaming only guide us how to push a stream to other 3rd live stream service (youtube, livecast, facebook, twitch, etc…).  
      - Pexip provides us some API which are potential source of the stream URI like: dial, dialOut, RTMP (See detail here: <https://docs.pexip.com/api_client/api_rest.htm>). But I haven’t tested them yet. Need API access from pexip to test.